.

FERRAMENTAS DE BUSCA MUSICAL BASEADA EM RECONHECIMENTO DE MELODIAS: ESTUDO, ANÁLISE E IMPLEMENTAÇÃO

Trabalho de Conclusão de Curso

Engenharia da Computação

Auno: Edinaldo José de Carvalho Araújo Santos

Orientador: Prof. Bruno José Torres Fernandes

UNIVERSIDADE
DE PERNAMBUCO

Edinaldo José de Carvalho Araújo Santos

FERRAMENTAS DE BUSCA MUSICAL BASEADA EM RECONHECIMENTO DE MELODIAS: ESTUDO, ANÁLISE E IMPLEMENTAÇÃO

Monografia apresentada como requisito parcial para obtenção do diplona de Bacharel em Engenharia da Computação pela Escola Politécnica de Pernambuco - Universidade de Pernambuco.

Orientador:
Bruno José Torres Fernandes

Departamento de Sistemas e Computação
Escola Politécnica de Pernambuco
Universidade de Pernambuco

Recife - PE, Brasil

31 de maio de 2011

Resumo

A área de recuperação da informação musical lida com a proposta de métodos e abordagens mais naturais para gerenciamento de objetos que expressam música, em áudio ou símbolos. Para aplicações de reconhecimento de melodias, apesar da complexidade da informação musical, diversas abordagens foram propostas nos últimos anos. Um dos sistemas mais conhecidos, o *Midomi*, apresenta uma base de dados criada por usuários do mundo todo. Talvez pela grande quantidade de melodias na base de dados, ou pela estratégia de reconhecimento usada, o sistema apresentou resultados animadores em testes preliminares. Apesar dos recentes progressos na área, como constatado pelo MIREX — Musical Information Retrieval Evaluation Exchange — um primeiro contato com as técnicas e algoritmos utilizados é sempre difícil, mesmo para aqueles com alguma experiência na área de reconhecimento de padrões. A proposta deste trabalho é reunir técnicas e métodos aplicados nesse contexto e implementar a primeira versão de um *framework* para desenvolvimento de ferramentas que suportem busca musical baseada em conteúdo.

Abstract

The research area of content-based musical information retrieval is concerned with more natural methods and paradigms to manage objects that represent music, in audio or symbolic forms. In respect to melody recognition, in spite of the complexity of musical information itself, a lot of approaches have been proposed in the last years. One of the most known system, Midomi, presents a large database generated by users all over the world. In a preliminary test, it shows promising results that could be explained by the great amount of melodies or the recognition strategy used. Apart from the achievements in the field, as indicated by MIREX — Musical Information Retrieval Evaluation Exchange — a first contact with the related techniques and algorithms is always difficult, even for a researcher experienced at pattern match problems. This work purpose is to put together produced knowledge about the subject and implement a initial version of a framework for content-based musical information retrieval tools development.

Dedicatória

Este trabalho é dedicado a minha irmã, pelo apoio e incentivo.

Agradecimentos

Agradeço a minha família pelo apoio. A Bruno, pela orientação. E aos *cantores*, em especial a Eliane, por não cobrarem cachê.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de símbolos

$ADSR$	Attack - Decay - Sustain - Release
$CBMIR$	Content-based Multimedia Information Retrieval
DFT	Discrete Fourier Transform
DTW	Dynamic Time Warping
EM	Expectation-Maximization
FFT	Fast Fourier Transform
HMM	Hidden Markov Model
HPS	Harmonic Product Spectrum
$ISMIR$	International Society for Musical Information Retrieval
$MIREX$	Musical Information Retrieval Evaluation Exchange
QBH	Query By Humming
RMS	Root Mean Square
ZCR	Zero-crossing Rate

1 Introdução

"Take care of the sense
and the sounds will take
care of themselves."
Lewis Carroll

1.1 Motivação

Recuperação de informação musical baseada em conteúdo é uma área complexa e recente. Nem todos os aspectos da informação musical são bem compreendidos e, mesmo quando a incerteza inerente à forma com que usuários distintos expressam a mesma informação melódica consegue ser modelada, características acústicas, como ruídos e harmônicas, assim como efeitos indesejados da amostragem de sinais, como o *aliasing*, exigem de uma abordagem de *reconhecimento de melodias* o uso correto de diversas técnicas de processamento de sinais.

Contudo, apesar de falta de padrões para avaliar os sistemas propostos, a aplicação de reconhecimento de melodias, motivada principalmente pela necessidade de gerenciar a enorme quantidade de informação musical transferida na internet atualmente, tornou-se um dos principais focos da área de recuperação musical baseada em conteúdo, talvez da recuperação de informação multimídia de uma forma geral.

Em pouco tempo, os resultados desses sistemas foram superados em grande velocidade. E hoje, de acordo com a avaliação mais recente do MIREX - *Musical Information Retrieval Evaluation Exchange* [1], atinge níveis de acerto próximos dos 95%.

Infelizmente, o conhecimento para implementar tais sistemas continua sendo bastante restrito, e a grande quantidade de abordagens existentes dificultam o pesquisador iniciante na área.

1.2 Objetivos

1.2.1 Objetivos principais

O principal objetivo do trabalho consiste em estudar as técnicas atuais para reconhecimento de melodias, analisando as dificuldades de implementação e as deficiências de cada técnica.

Além disso, apresentaremos uma versão inicial de um *framework* desenvolvido para suportar reconhecimento de melodias através das principais técnicas estudadas.

É do interesse do trabalho qualificar o desempenho das implementações apresentadas, mesmo que apresentem, de antemão, restrições com relação ao estado da arte.

Uma avaliação empírica dos sistemas atuais de reconhecimento de melodias disponíveis na internet é feita. Uma avaliação mais formal, no entanto, exigiria uma base de dados maior e mais variada do que a apresentada neste projeto.

1.2.2 Objetivos secundários

Como objetivos secundários deste trabalho, podemos destacar a disseminação local do tema de reconhecimento de melodias, principalmente na instituição em que o projeto fora desenvolvido.

O *framework* apresentado será disponibilizado em repositório público. Espera-se com isso estimular colaborações com o seu desenvolvimento.

1.3 Estrutura da monografia

O primeiro capítulo desta monografia apresenta uma visão geral da área de recuperação musical baseada em conteúdo, destacando brevemente as principais aplicações, bem como os principais desafios encontrados pelas correntes pesquisas na área. Ainda nesse capítulo, o formato MPEG-7 — padrão para codificação de áudio e vídeo criado especificamente para suportar diversas aplicações de recuperação baseada em conteúdo multimídia — é apresentado.

O segundo capítulo começa com uma visão geral de sistemas para recuperação musical baseada em reconhecimento de melodias. À medida que o capítulo se desenvolve, são construídos os fundamentos sobre os quais diversos sistemas de QBH — *query by*

humming — são implementados, módulo a módulo. Dentre as ferramentas matemáticas apresentadas, destacam-se o *dynamic time warping* — DTW — e o modelo oculto de Markov, HMM.

Em seguida, no capítulo 3, é descrita a implementação de uma ferramenta de QBH na linguagem de programação Java. A ferramenta faz uso das técnicas apresentadas no capítulo anterior e pode ser baixada através do repositório svn do projeto, qbhtool, no *google code* [1].

Uma breve avaliação da ferramenta é apresentada no capítulo 4, assim como a avaliação de dois dos sistemas mais conhecidos de QBH atuais: o Musipedia [2] e Midomi [3].

No último capítulo deste trabalho, uma perspectiva para o fututo da aplicação de reconhecimento de melodias é desenvolvida, assim como os planos para melhorar a ferramenta que integra o projeto.

Como apêndices, encontram-se ainda a descrição da implementação do algoritmo FFT utilizada, uma rápida explicação sobre a frequência (*picth*) das notas musicais, e a lista de músicas apresentadas aos voluntários dos quais as gravações compuseram a base de dados.

[1] http://code.google.com/

2 Recuperação de informação musical baseada em conteúdo

A informação musical tornou-se um dos principais recursos compartilhados na internet nos dias atuais (ver, por exemplo, [4]).

Em sua abordagem tradicional, a gestão da informação musical depende exclusivamente da associação de metainformações textuais — como autor e título da obra, gênero, letra, por exemplo — aos objetos que codificam a informação musical em si, como fluxos de áudio digital ou uma descrição simbólica da música (como uma partitura digital).

Em alguns sistemas, metainformações associadas por um usuário a determinado objeto multimídia (com um código identificador) torna-se disponível para qualquer outro usuário que apresente o mesmo objeto. Por exemplo, o MusicBrainz [1] e o freedb [2] são sistemas gratuitos que permitem o compartilhamento de metadados associados a faixas de um disco compacto de áudio.

No entanto, as primeiras abordagens de busca musical baseada em conteúdo exploravam recursos de *impressão digital* — código gerado a partir de características acústicas para identificar um registro — como códigos *TRM* [3], usado, por exemplo, para eliminar cópias duplicadas de músicas em uma base de dados. Um programa que usa impressões digitais para identificação musical é o Shazam[4].

Os trabalhos na área de recuperação de informação musical baseada em conteúdo são bem mais recentes do que trabalhos semelhantes direcionados à informação visual. Muitos são experimentais — apresentam baixo desempenho, ou os resultados são fortemente dependentes das condições em que os testes são realizados.

Uma questão que dificultou bastante os avanços foi a inexistência de procedimentos

[1] http://musicbrainz.org/
[2] http://freedb.org/
[3] http://www.relatable.com/tech/trm.html
[4] http://www.shazam.com/

(e base de dados) padrões para a avaliação das diferentes técnicas e algoritmos propostos pela comunidade. Nesse sentido, em 2005, foi organizado o primeiro encontro entre pesquisadores da área com o objetivo de estabelecer métodos para a avaliação e comparação das aplicações atuais de recuperação musical baseada em conteúdo. Esse encontro ficou conhecido como MIREX - Musical Information Retrieval Evaluation Exchange, e suas versões anuais acontecem na Conferência da Sociedade Internacional para Recuperação de Informação Musical - ISMIR.

Um outro grande projeto voltado para a comunidade científica interessada na área é o *Network Environment for Music Analysis*, ou NEMA. O NEMA consiste de um arcabouço (*framework*) de recursos aberto e extensível, baseado em *webservice*, para facilitar a integração entre dados musicais e ferramentas analíticas e de avaliação. Informações sobre o projeto, assim como informações sobre o ISMIR em geral, podem ser encontradas na página do *Laboratório de Avaliação de Sistemas de Recuperação de Informação Musical* - IMIRSEL[5].

2.1 Características musicais de alto nível

A dificuldade da extração das características de alto nível da música reflete a complexidade da informação musical. Existe uma associação direta entre essas características e as Tarefas do interesse da recuperação de informação musical baseada em conteúdo. Na Tabela 1, essas características estão listadas e associadas as respectivas tarefas de acordo com [5].

É importante ressaltar que, dependendo da aplicação, a informação de alto nível nem sempre garante melhor desempenho em relação a uma abordagem que lida diretamente com as características de baixo nível. É o que acontece, por exemplo, com reconhecimento de melodias em aplicações de *query by humming*, segundo a avaliação mais recente do MIREX.

Como podemos observar na Tabela 1, as características de alto nível são aquelas de maior interesse para usuários em geral de sistemas de QBH—colecionadores de música digital, músicos profissionais e amadores, musicólogos, por exemplo, pois está relacionado ao conhecimento intuitivo ou aprofundado de como uma peça musical é criada.

[5] www.music-ir.org/evaluation/

Descrição	Fonte	Tarefas
Timbre	Áudio	Reconhecimento de instrumentos Reconhecimento de vozes percusivas com frequência percebida
Melodia / Baixaria	Áudio / Simbólica	Extração da linha melódica Extração da linha do baixo
Ritmo	Áudio	Detecção de limites iniciais e finais Identificação métrica Alinhamento métrico (barras) Detecção da batida Detecção do tempo Tempo médio
Frequência percebida (*pitch*)	Áudio	Frequências fundamentais simples Frequências fundamentais múltiplas
Harmonia	Áudio / Simbólica	Extração de rótulo de acordes Extração da linha do baixo
Tom	Áudio Simbólico	Modulação Vocabulário tonal
Estrutura	Áudio Simbólico	Extração de verso e refrão Extração de repetições
Letras	Áudio	Reconhecimento de canto e letra Reconhecimento de palavras
Música oriental	Áudio	Sistemas de afinação micro-tonal Identificação de elementos incomuns da música ocidental

Tabela 1: Características de alto nível da informação musical

2.2 Especificidade das tarefas de recuperação da informação musical

As tarefas de recuperação da informação musical têm como objetivo recuperar, a partir de metadados textuais ou do próprio conteúdo musical, músicas ou informações associadas a estas.

As tarefas podem ser classificadas de acordo com o tipo do objeto recuperado (de áudio ou simbólico).

A especificidade das tarefas é outro importante parâmetro que pode ser utilizado para classificá-las. Basicamente, ela nos diz a quantidade de objetos que uma consulta tratada por aquela tarefa pode casar.

O nível de especificidade também determina o nível das características musicais codificadas na consulta. Por exemplo, características de alto nível são do interesse das tarefas de média especificidade. As características de baixo-nível, no entanto, pode interessar a tarefas de qualquer nível.

A Tabela 2 apresenta exemplos de tarefas da recuperação de informação musical associadas ao seu nível de especificidade.

2.3 O Padrão MPEG-7

O *Moving Picture Expert Group*–MPEG, grupo formado pela Organização Internacional para Padronização, ou ISO, e responsável pela definição de padrões de codificação de áudio e vídeo, desenvolve desde 1996 o primeiro padrão voltado para a descrição de conteúdo multimídia otimizada para busca em grande bases de dados. Os padrões anteriores do grupo, o MPEG-1, MPEG-2 e MPEG-4, focaram principalmente na compressão de áudio e vídeo. Eventualmente, o MPEG-4 foi estendido com suporte a conteúdo 3D e gestão de direitos digitais - DRM.

O *MPEG-1 Audio Layer III*, posteriormente *MPEG-2 Audio Layer III*, desenvolvido com a participação de engenheiros de grupos como a *Fraunhofer Society for the Advancement of Applied Research*, entre outros integrantes do *MPEG Audio Subgroup* foi o principal formato de arquivos de áudio transferidos na internet nos últimos anos. A busca por registros de áudio nesse formato baseia-se no casamento de metainformações textuais (armazenadas no componente ID3 do arquivo), como autor, título, gênero, ano e álbum.

Tarefa	Especificidade	Descrição
Identificação Musical	Alta	Identificação discos e faixas; identificação por impressões digitais
Detecção de plágios	Alta	Identifica apropriações ilegais de propriedades intelectuais
Monitoramento de *copyright*	Alta	Monitora a disseminação musical por infringimentos de *copyright*
Versões	Alta/Média	Identifica *remix*, versões ao vivo/estúdio, *cover* (normalização da base de dados)
Melodia	Alta/Média	Encontra obras que contém um fragmento melódico.
Intérprete	Média	Encontra músicas por artista
Semelhaça de sons	Média	Encontra músicas semelhantes
Alinhamento musical	Média	Mapeia gravações distintas, abstraindo escala de tempo e repetições
Compositor	Média	Encontra músicas por compositor
Recomendação	Média/Baixa	Encontra música que se encaixa ao perfil musical do usuário
Humor	Baixa	Encontra música a partir de conceitos emocionais, como alegria, nostalgia, etc.
Estilo/Gênero	Baixa	Encontra por categorias genéricas
Instrumento	Média	Procura músicas que apresentam sons de intrumentos específicos
Música-Fala	Média	Segmentação em transmissões de rádio; catálogo de arquivos musicias

Tabela 2: Tarefas de recuperação de conteúdo musical e respectivas especificidades

Geralmente essas metainformações, como visto no capítulo 1, são associadas aos registros colaborativamente.

O padrão MPEG-7, no entanto, apresenta ferramentas de descrição que podem ser usadas para representar diversas características de objetos multimídia em geral. Dentre as que dizem respeito a registros musicais apenas, podemos destacar os descritores de melodia e de expressões rítmicas e tímbricas. Vale ressaltar, como observado por [6], que descritores para características de nível de abstração mais baixo, como cor, textura, movimento, tom musical, etc., podem ser construídos automaticamente. Isso não acontece com características de mais alto nível, como a subjetividade do conteúdo, ou interações entre elementos de uma cena, por exemplo. Nesses casos, metainformações textuais devem ser inseridas manualmente pelos usuários.

O capítulo 4 desta monografia apresenta uma abordagem para extração automática de características da música para reconhecimento de melodias. Em [7], as diversas ferramentas do padrão são descritas e um sistema de recuperação de informação multimídia baseada em conteúdo é apresentado.

3 *Sistemas de recuperação musical baseada em reconhecimento de melodias*

3.1 Reconhecimento de melodias: *query by humming*

Em 1995, Ghias [8] apresentou um sistema capaz de reconhecer melodias de uma pequena base de dados construída a partir de músicas MIDI - *Musical Instrumental Digital Interface*, ou *Interface Digital de Instrumento Musical*. Ghias aplicou o conceito de *contorno melódico*, a forma natural como nós percebemos a melodia. Para comparar diferentes *contornos melódicos*, adotou o algoritmo do *casamento aproximado para cadeias de caracteres* [9]. A Figura 1 mostra a arquitetura de seu sistema.

Desde o referido trabalho, a tarefa de busca musical a partir de um trecho da melodia *trauteada* ou *cantarolada* pelo usuário passou a ser conhecida na literatura como *query by humming*, ou QBH. Comumente, os termos *singing* e *humming* aparecem com distinção, como o ato de cantar com ou sem a letra da música, respectivamente. Aqui, serão usados indiscriminadamente, com ressalvas apenas quando uma técnica favorecer naturalmente uma ou outra abordagem. Em alguns casos, os sistemas de reconhecimento de melodia podem identificar melodias *assobiadas* ou até mesmo *tocadas* por diferentes instrumentos musicais.

Existem dois principais interesses em QBH. Primeiramente, permitir ao usuário identificar uma música da qual não conheça (ou lembre de) qualquer *metadado* associado, mas seja capaz de reproduzir algumas notas da melodia, mesmo sem a respectiva letra. Não obstante, se parte da letra da música for conhecida, muitas vezes a busca baseada apenas nessa informação textual digitada pelo usuário pode falhar, se ela apresentar erros estruturais, como palavras trocadas por sinônimos ou termos com quais rimam, por exemplo. Nesse caso, o reconhecimento de melodias ainda pode obter sucesso.

O *Midomi*[3] é uma rede social desenvolvida para explorar esse recurso. Nele, a base de dados de melodias é construída pelos próprios usuários do sistema, que gravam músicas em seus perfis. Nesse caso, a base de dados está em constante crescimento. Além disso, não depende de informações simbólicas da música, como arquivos MIDI, sujeitas a direitos autorais.

Outro grande interesse nessa ferramenta consiste em dispor de uma maneira mais natural para consultar coleções de músicas digitais armazenadas principalmente em dispositivos portáteis, como celulares e *tablets*, que, por concepção, não apresentam mecanismos tradicionais de entrada de dados dos computadores pessoais (*desktop*). Por exemplo, cantando parte de uma melodia, o usuário pode selecionar para reprodução a faixa associada, ou mesmo músicas do álbum ou artista relacionado, se preferir.

Vale ressaltar que, ao contrário da aplicação de reconhecimento musical, que reconhece gravações específicas através de impressões digitais, QBH é uma aplicação de menor especificidade, ou seja, reconhece uma música se a melodia desta contiver as notas cantadas—ou melhor, uma sequência semelhante de notas. Dessa forma, podemos aplicá-la também em uma base de dados *universal* para detectar plágios ou incidências musicais, embora os sistemas atuais não sejam tão escaláveis para base de dados maiores do que dezenas de milhares de registros.

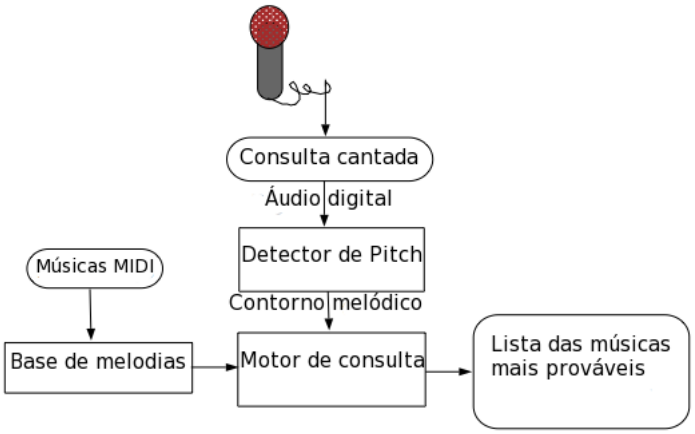

Figura 1: Sistema de QBH de GHIAS

A base de dados de um sistema de QBH é, frequentemente, construída a partir da transposição de músicas MIDI (apenas a faixa da melodia) para o formato de representação musical adotado pelo sistema. Para casos em que a técnica de reconhecimento baseia-se em propriedades estatísticas associadas à incerteza do canto, é necessário também estimar os parâmetros do modelo. Para estimá-los, no entanto, precisamos de gravações da mesma

melodia por diferentes usuários. Se dispormos de gravações suficientes, possivelmente, a versão MIDI é dispensável.

Capturar o sinal de áudio que codifica uma melodia cantada por um usuário, seja para construir a base de dados, seja para consultar o sistema por melodias conhecidas, requer atenção quanto ao nível de ruído do ambiente, à taxa de amostragem (geralmente, uma taxa de 22kHz é suficiente, mas dependendo da técnica de detecção de *pitch* utilizada, usar amostragens mais altas pode evitar problemas de baixa resolução e *aliasing*), e à expressão fonética permitida. Para métodos de segmentação baseados principalmente na energia, por exemplo, o canto só poderá ser segmentado com eficiência restringindo-se aos sons de fonemas específicos, como será visto em breve.

O sinal amostrado é, então, dividido em segmentos de duração fixa, geralmente entre 10 e 50ms. A determinação do *contorno de pitch* a partir da estimativa da frequência fundamental segmento por segmento do sinal é o próximo passo. Após a determinação desse contorno, o sinal poderá ser segmentado em notas ou, do contrário, será dividido em quadros de tamanho fixo e a comparação feita nesse nível (nesse caso, a representação musical pode ser apresentada em funcão do tempo, uma vez que cada quadro corresponde a uma unidade de tempo).

Como proposto na abordagem de McNab et al. [10], trabalho paralelo ao de Ghias, heurísticas ou técnicas de segmentação podem ser aplicadas para extrair notas musicais. Dessa forma, é possível elevar o nível de abstração da representação melódica utilizada. O *contorno de pitch*, nesse caso, é transformado em uma sequência de notas $N = n_i, n_{i+1}, \cdots, n_N$ do sinal. Por sua vez, a sequência de notas poderá ser convertida em uma sequência de *intervalos tonais*, para refletir nossa percepção natural da melodia [11]: o *contorno melódico*

Portanto, em casos em que a segmentação é aplicada, existe para cada símbolo da *sequência de notas* um intervalo de tempo d correspondente à duração da nota musical. Esses valores de tempo constituem informações rítmicas a respeito da música e foram desconsiderados por muitos sistemas inicialmente. Atualmente, no entanto, existem abordagens de recuperação musical baseada apenas nela [12].

De fato, para algumas músicas, a expressão rítmica é mais determinante do que a própria sequência de notas sem informação a respeito do tempo associada.

As técnicas de reconhecimento mais usadas com melodias são o DTW—dynamic time warping; e o HMM—modelo oculto de Markov. Vale ressaltar que o DTW, mesmo sendo

uma técnica aplicada a nível de quadro, pode apresentar resultados melhores do que sistemas que lidam com representação melódica de alto nível, conforme constatado por [13].

3.2 Estimativa da frequência das notas

Um sinal acústico discreto (amostrado) produzido por uma fonte sonora pode ser genericamente modelado por uma série de funções senoidais complexas com frequências múltiplas de uma frequência fundamental, ω_k, assim expressa:

$$x_k(n) = \sum_{l=1}^{L_k} a_{k,l} e^{j\omega_k l n} + e_k(n), \tag{3.1}$$

sendo $a_{k,l} = A_{k,l} e^{j\phi_{k,l}}$ a amplitude complexa da *l-ésima* harmônica da fonte k e $e_k(n)$ o ruído associado.

O sinal representado dessa forma é composto pela parte periódica (associada a um único *pitch* ou *frequência percebida*) e parte não periódica (ruído). Exemplo de sinais que obedecem esse modelo (de único pitch) são o sinal da fala; ECG - eletrocardiograma (exceto para mulheres grávidas); sons de instrumento de sopro como o trompete; e notas tocadas em instrumentos musicais, como violão, violino, etc., isoladamente (sem sobreposição).

A frequência fundamental do sinal é expressa por

$$\omega_k = \frac{2\pi}{\tau_k}, \tag{3.2}$$

onde τ_k corresponde ao *período de pitch*.

O conceito de frequência fundamental é muitas vezes usado na literatura como sinônimo de *pitch*. No entanto, é importante ressaltar que não existe sempre correspondência entre frequência fundamental e a frequência percebida. O *pitch* é atributo dos tons musicais, como o timbre, a intensidade e a duração, mas sua percepção, muitas vezes, está associada a fenômenos *psicoacústicos* que não são explicados pela física, como descreve Plack [14]. Um exemplo disso é efeito da fundamental ausente.

A frequência física do sinal, f_k (Hertz), está relacionada à frequência fundamental da seguinte maneira:

$$\omega_k = 2\pi \frac{f_k}{f_c}, \tag{3.3}$$

onde f_c corresponde a frequência de amostragem do sinal (o índice c indica valor complexo).

É possível estimar a frequência fundamental do sinal tanto no domínio do tempo quanto no domínio da frequência, como será mostrado a seguir através de duas técnicas simples para único *pitch*. Rabiner [15] apresenta um estudo comparativo de diversos algoritmos de detecção de *pitch* — PDA.

Além dos algoritmos de detecção da frequência fundamental apresentados, vale citar ainda a função de diferença da magnitute média, ou AMDF, o método da probabilidade máxima, considerado um pouco lento, e a análise cepstral.

O *cepstrum* é um anagrama para *spectrum*. O conceito do *cepstrum* foi criado para refletir melhor a relação de nossa percepção com relação a frequência dos sons, que, naturalmente, é associado a uma escala logarítmica.

O Apêndice A apresenta mais detalhes sobre o *pitch* associado às notas musicais em diferentes oitavas.

3.2.1 Funções *janela*

Considere, independentemente do algoritmo PDA utilizado, que o sinal amostrado é, inicialmente, multiplicado por uma função conhecida como *função janela* com o objetivo de isolar um pequeno segmento do sinal para ser analisado.

Para exemplificar, considere a função janela retangular definida da seguinte maneira:

$$r(j) = \begin{cases} 1 & offset \leq j < offset + w \\ 0 & \text{caso contrário} \end{cases}, \tag{3.4}$$

onde *offset* é um variável que nos permitir mover a função retangular sobre qualquer região do sinal original, e w é a largura da janela.

Multiplicar o sinal pela função *janela retangular* para $offset = 0$ resulta em um novo sinal que conserva as w primeiras amostras do sinal original anulando todas as demais. Para isolarmos as próximas w amostras do sinal, simplesmente incrementamos $offset$ pelo valor w e multiplicamos novamente o sinal original pela *função janela*.

Alternativamente, se definirmos o incremento do *offset* para um valor menor do que a largura w da janela, criaremos sobreposição entre janelas subsequentes. Essa é uma forma de garantir que as mudanças de propriedades de uma janela para a seguinte seja suavizada.

Sempre que nos referimos a janela de um dado sinal de áudio, podemos assumir que estamos nos referindo a uma função $h(t)$, tal que

$$h(t) = f(t)w(t), \tag{3.5}$$

onde f(t) é a função original e w(t) a função janela escolhida para analisar o sinal em um determinado intervalo.

Ademais, estabeleceremos aqui que sempre que o tipo de janela for omitido, ficará implícito a janela retangular.

Ao se trabalhar no domínio da frequência, devemos multiplicar o sinal por uma janela de ponderação suavizada nos extremos. Um exemplo comum é a janela Hanning, cuja função está representada na equação 3.6.

$$A(x) = cos^2(\frac{x\pi}{2a}) \tag{3.6}$$

A janela do sinal conserva as propriedades acústicas do sinal original no intervalo determinado, porém suaviza as descontinuidades existentes nos extremos, facilitando o cálculo da transformada.

3.2.2 A autocorrelação

A autocorrelação é uma das técnicas mais antigas para estimar a frequência de sinais aproximadamente periódicos. Um dos algoritmos baseados nessa técnica mais citados nos trabalhos de reconhecimento de melodias foi proposto por Gold e Rabiner. A definição da autocorrelação é apresentada na equação 3.7. No entanto, uma definição alternativa, mas que segue a ideia geral da autocorrelação, é descrita em seguida.

A operação tradicional de *autocorrelação* aplicada sobre uma janela do sinal, h(t), é definida como:

$$R(s) = \sum_{n=1}^{N} h(n)h(s+n), \tag{3.7}$$

onde s corresponde a um deslocamento aplicado ao sinal, $h(n)$ é o valor do sinal para a n-ésima amostra da janela, e N o tamanho da janela em número de amostras.

A autocorrelação deve ser máxima quando o deslocamento corresponder a frequência (em número de amostras) do sinal. A técnica da autocorrelação consiste em maximizar a função para o menor valor possível de s (domínio) dentro de uma faixa, evitando assim a ocorrência de harmônicas.

Sabe-se que um sinal aproximadamente periódico é semelhante ao próprio sinal defasado por um intervalo de tempo correspondente a um ciclo. Se calcularmos a diferença absoluta ponto a ponto entre o sinal e sua versão defasada em s amostras, e depois somarmos as diferenças, encontraremos uma função que, para efeito da detecção da frequência fundamental, serve ao mesmo propósito da função de autocorrelação definida na equação 3.7, exceto que agora desejamos minimizar a função, e não mais maximizá-la. De agora em diante, usaremos esta última definição para a função de autocorrelação.

A derivada da função de autocorrelação nos informa, em toda mudança de sinal negativo para positivo, a possível frequência fundamental do sinal (dada pelo mínimo, s_{min}, da função de autocorrelação).

As Figuras 2 e 3 ilustram a técnica da autocorrelação.

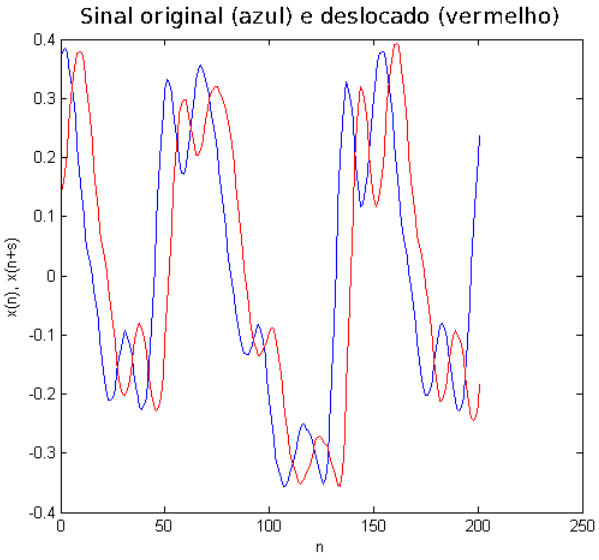

Figura 2: Um sinal de áudio e sua versão defasada

É importante ressaltar, porém, que o valor de s encontrado corresponde à frequência

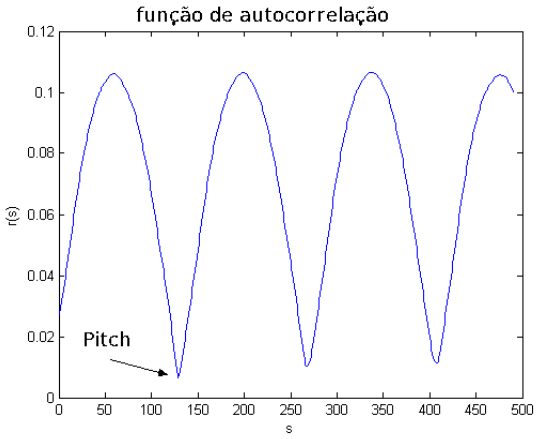

Figura 3: A função de autocorrelação

em número de amostras. A relação expressa na equação 3.3 nos permite calcular a frequência física correspondente.

Apesar de relativamente robusta, a autocorrelação é muito sensível a taxa de amostragem. Tecnicamente, quanto menor essa taxa, menor a resolução dos valores de frequência.

É preciso também tomar cuidado com o tamanho da janela. É imprescindível que o seu período (número de amostras) seja pelo menos igual a duas vezes o menor período que se deseja detectar.

Se a taxa de amostragem ou o tamanho da janela não forem grandes o suficiente, observaremos, com mais frequência, o efeito do *aliasing*, que se manifesta através de distorções e ruídos e incapacidade de reprodução adequada do sinal amostrado. Quando um sinal é amostrado a uma taxa inferior ao dobro da maior frequência que desejamentos representar no sinal (segundo o Teorema de Nyquist), não é possível reconstituí-lo com perda aceitável de informação.

Além disso, algumas considerações precisam ser feitas para que a autocorrelação apresente desempenho próximo ao exigido por uma aplicação de tempo real.

Primeiro, é preciso definir o menor número de pontos (valores de s) para calcular a função de autocorrelação dentro do intervalo de frequência desejado. Por exemplo, para uma aplicação envolvendo apenas a voz, uma faixa de frequência entre 50Hz e 22kHz são razoáveis e, portanto, podemos considerar valores de s entre 1 e 600 apenas.

Se calcularmos a derivada ainda durante o cálculo da função de autocorrelação, podemos encontrar o mínimo para uma janela em particular antes mesmo de calcular a função para todos os valores de s.

Por último, procurar o mínimo da autocorrelação considerando a frequência observada na janela anterior resulta também em ganho de desempenho. Podemos assumir, caso a taxa de amostragem seja alta o suficiente e a janela grande, que não haverá mudanças drásticas na frequência do sinal.

3.2.3 O espectro do produto harmônico

Para determinar a frequência fundamental de uma nota musical no domínio da frequência, podemos recorrer a um algoritmo prático conhecido como o *espectro do produto harmônico*, ou HPS. Esse algoritmo parte do princípio que o espectro da janela de um sinal é formado por picos de energia situados na fundamental e na série harmônica da nota.

O algoritmo consiste em multiplicar o espectro do sinal, obtido a partir do algoritmo da Trasformada Rápida de Fourier - FFT, por versões comprimidas do próprio espectro. A compressão é realizada reduzindo a amostragem do espectro original por fatores inteiros.

A medida que comprimimos o espectro por um fator inteiro a, a frequência harmônica dada por $a\omega_k$ deve se alinhar à frequência fundamental ω_k. Dessa forma, após um pequeno número de compressões, teremos um alinhamento na posição da frequência fundamental das harmônicas correspondentes.

Ao multiplicarmos todas as versões do espectro (original e comprimidas), o resultado é uma função na qual o maior pico está posicionado na frequência fundamental.

A Figura 4 ilustra a técnica.

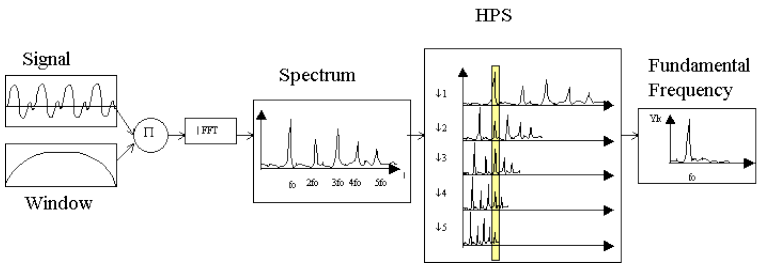

Figura 4: Espectro do produto harmônico

Apesar da praticidade do algoritmo, sua precisão depende da resolução da função do espectro e, consequentemente, do tamanho da janela. Se esta crescer demais, o tempo de processamento pode tornar-se incômodo.

3.3 O contorno melódico e o código de Parson

Na abordagem de Ghias, após a determinação do *pitch* de cada segmento do sinal (ver seção 3.2), as notas são reconstruídas (embora ele não tenha sido claro quanto à abordagem utilizada) e o contorno melódico, que corresponde à função da variação da frequência do sinal no tempo, calculado.

Em seguida, o código de Parson, criado por Daynes Parson, é usado para codificar o contorno. O código é assim definido:

$$p(i) = \begin{cases} U & \text{se pitch(i)} > \text{pitch(i-1)} \\ D & \text{se pitch(i)} < \text{pitch(i-1)} \\ R & \text{se pitch(i)} = \text{pitch(i-1)} \\ * & \text{se i} = 0 \end{cases} \tag{3.8}$$

onde o *pitch*(i) denota a frequência fundamental da i-ésima nota do sinal.

O código de Parson não leva em consideração os atributos rítmicos da nota. Sua única vantagem em relação a outras representações invariantes à entonação é a simplicidade, que implica em menor tempo de processamento da consulta.

É possível, no entanto, codificar a variação de frequência da linha melódica quantitativamente, usando o intervalo de semitons, por exemplo, ao invés do código de Parson. Todos os trabalhos recentes optam por uma representação quantitativa para a melodia. Outro exemplo de representação da melodia é descrito durante a apresentação do HMM — modelo oculto de Markov — na seção 3.5.

3.4 A extração das notas musicais

Para a maioria das aplicações de processamento digital de áudio no domínio da música, um dos problemas mais importantes consiste em detectar, no sinal digital, eventos que delimitam o itervalo de tempo durante o qual cada nota musical pode ser percebida. Essa informação pode ser usada para refinar a informação do contorno melódico, isolando as regiões de silêncio do sinal musical e agrupando os segmentos nos quais o *pitch* permance constante e a variação de energia estável (sustentação da nota).

Em processamento de sinais digitais de imagem, mais especificamente na disciplina de visão computacional, origem dos primeiros algoritmos de segmentação, o objetivo desse

procedimento consiste em extrair da imagem objetos e regiões de representação mais simples, mas capazes de transmitir características importantes a respeito dela.

Em processamento de sinais musicais, dois conceitos distintos embora semelhantes são importantes para descrever as diferentes abordagens para a solução desse problema: *ataques* e *transientes*.

Observando a representação da forma de onda do som de uma nota isolada produzida por três instrumentos distintos e o envelope de amplitude para essas notas, Figuras 5 e 6, respectivamente, pode-se notar, principalmente na imagem do envelope, a existência de quatro fases distintas.

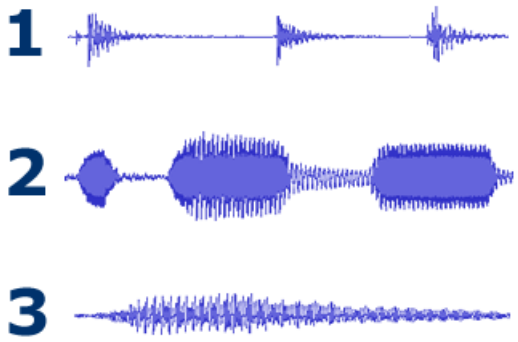

Figura 5: Formas de onda de notas isoladas

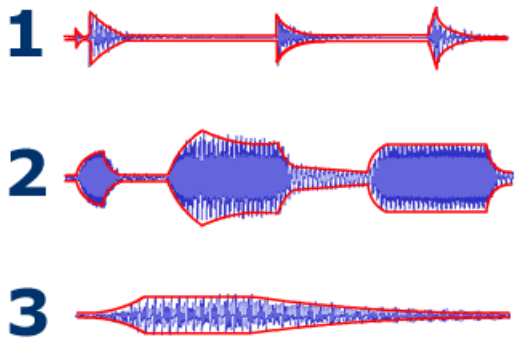

Figura 6: ADSR das notas

O envelope de amplitude para uma nota musical isolada é conhecido na literatura por ADSR — ou ataque, decaimento, sustentação e liberação, do inglês: *attack, decay, sustain, release* — em referência às quatro fases observadas em um tom emitido por algum instrumento musical acústico, incluindo o nosso próprio trato vocal.

O período transiente abrange desde o início da excitação até o instante em que os componentes de frequência impuros, i.e. não associados à nota em questão, perdem energia

e são subjulgados pela frequência fundamental associada ao tom (ou uma combinação de suas harmônicas). Isso geralmente ocorre na fase de sustentação.

Durante o período de sustentação, portanto, a frequência da nota pode ser mais facilmente detectada, através de alguma das técnicas apresentadas na seção 3.2.

Existem diversas abordagens para a segmentação das notas, principalmente ao lidar com intervalos harmônicos, nos quais duas ou mais notas se sobrepõem durante um intervalo de tempo. Um tutorial extenso sobre o assunto consta em [16].

Neste trabalho, no entanto, apresentaremos apenas a abordagem de segmentação baseada na característica peculiar da energia na fase de *ataque* da nota. Em relação a uma abordagem focada na frequência do período transiente, a principal desvantagem consiste na dificuldade de identificar notas cantadas com fonemas de fraco som consonantal. Ainda assim, a segmentação baseada em energia é mais robusta para melodias que apresentam rápidas sequências de notas na mesma frequência percebida. Por essa razão, é mais utilizada.

3.4.1 Abordagem da energia

Haus e Pollastri [17] apresentaram uma abordagem de segmentação de notas musicais partindo do princípio que, assim como nos sinais da fala comum, o som detectado em sinais musicais do canto podiam ser classificados entre *com voz* e *sem voz* (na verdade, a classificação em que eles se basearam apresentava três classes).

Grosseiramente, as classes podem ser associadas aos sons de vogais e consoantes, respectivamente. Das primeiras, é fácil extrair informação a respeito da frequência das notas, uma vez que as condições de produção do som favorecem a *periodicidade aproximada* do sinal. No entanto, nas segundas, o nível de ruído presente no sinal é relevante para extrair informações a respeito dos limites (temporais) das notas e, consequentemente, rítmicas.

Foram definidos também três níveis de abstração para as operações com o sinal musical. No nível mais baixo, de *quadro*, eram calculados o espectro do sinal, a taxa de cruzamento no valor zero (ZCR), a energia RMS, e possíveis erros de oitavas.

No nível seguinte, de *bloco*, um limiar para o som ambiente é determinado, segmentos são classificados entre as classes com ou sem voz e isolados do restante do sinal (segmentos de *silêncio*). Por último, o *pitch* de cada segmento é estimado (ver seção 3.2).

No nível de evento, mais alta abstração, os eventos de início e fim de notas são detectados e as notas são determinadas (tanto em frequência quanto em duração).

Na fase de pré-processamento, é estabelecido um limiar para a discriminação sinal/ruído em 15% do valor da energia RMS dos primeiros 60ms do sinal. Se esse valor for superior a -30dB, o som ambiente é considerado ruidoso demais e o sinal não pode ser processado. Caso contrário, o sinal é dividido em segmentos de 10ms e uma classificação inicial entre som e silêncio é feita comparando o RMS de cada segmento com o limiar calculado anteriormente.

Em seguida, várias heurísticas para classificar os segmentos entre as classes *com voz* e *sem voz* podem ser consideradas. Entre elas, analisar a derivada do sinal, normalizada para o maior valor, na qual valores altos indicam mudanças bruscas de amplitude (características da região de ataque da nota). Outra informação que pode ser analisada trata-se da taxa de cruzamento no valor zero dentro de cada segmento de 10ms do sinal. Valores tão altos quanto 49 são típicos de regiões *sem voz*, enquanto que valores em torno de 14 são típicos de regiões *com voz*.

Considerar outras heurísticas pode melhorar o desempenho da segmentação, a custo de tempo de execução. No projeto apresentado no capítulo 4, apenas essas duas foram usadas.

3.5 O reconhecimento de melodias

Os algoritmos de reconhecimento de padrão representam o componente mais importante e complexo de um sistema de QBH. Para ser aplicada nesse contexto, uma técnica precisa ser robusta o suficiente para identificar melodias independentemente de erros nas notas individuais (em tempo e/ou frequência) ou na afinação, considerando que o nível de treinamento musical dos potenciais usuários é variado.

O sistema deve reconhecer a melodia independentemente da transposição de tom na qual foi apresentada ao sistema. Dependendo da aplicação, possui restrição de tempo máximo para executar uma consulta. Na pior das hipóteses, esse tempo não deve ultrapassar alguns poucos segundos.

Escolher a representação da melodia deve levar em consideração alguns aspectos importantes. Um deles é se a invariância em relação a transposição tonal deve ser codificada na própria representação ou se a técnica de reconhecimento deve tratar o problema.

Uma das primeiras técnicas usada para reconhecer melodias baseia-se no casamento aproximado de cadeias de caracteres. A técnica usada por Ghias, compara duas melodias em código de Parson relativamente bem, desde que ambas sejam cantadas em escala de tempo compatíveis, que é uma suposição pouco razoável.

Nos sistemas mais recentes, no entanto, duas técnicas são dominantes. O DTW é a principal técnica que lida diretamente com a forma de onda, ou seja, não depende da abstração da nota musical. Foi concebida para alinhar duas séries temporais (no nosso caso, melodias) modificando a escala de tempo de qualquer uma delas para garantir a melhor similaridade.

Por sua vez, o modelo oculto de Markov, HMM, tradicionalmente usado em aplicações de reconhecimento da fala, foi depois proposto por Shiffrin [18] como uma abordagem alternativa para a tarefa de reconhecimento de melodias. O HMM é construído para capturar propriedades estatísticas da informação contida na sequência de notas que compõem a melodia. Portanto, é uma abordagem de mais alto nível que depende bastante do desempenho da segmentação.

Uma comparação da eficiência de três técnicas de reconhecimento de melodias baseadas em programação dinâmica, entre elas o HMM e o DTW, pode ser encontrada em [19].

3.5.1 Dynamic Time Warping: invariância à escala de tempo

O Dynamic Time Warping, ou DTW, tem como principal objetivo prover uma abordagem para comparar características de séries temporais que são conservadas quando a escala de tempo de qualquer uma delas é alterada.

O DTW flexiona o eixo temporal (ou o eixo que representa os valores do domínio) de uma ou de ambas as funções para encontrar o melhor alinhamento entre elas. Quando aplicado entre duas melodias, o sistema abstrai a possível diferença de tempo entre elas, nos dando o melhor grau de semelhança.

No entanto, diferenças de amplitude — que, em nosso caso, é uma medida da frequência (ou variação de frequência) da nota — causadas pela transposição da melodia em diferentes tons devem ser tratadas independentemente (a técnica só garante a otimização no eixo do tempo).

Considere duas sequências temporais representadas por $T = t_1, t_2, t_3, \ldots, t_N$ e $R =$

$r_1, r_2, r_3, \ldots, r_M$, ou $t(i)$ e $r(j)$, respectivamente.

O algoritmo do DTW determina que a tabela de mapeamento entre T e R, $D_{i,J}$, pode ser obtida pela seguinte função:

$$d_{i,j} = d(i,j) + min \begin{cases} d_{i-1,j} \\ d_{i-1,j-1}, \\ D_{i,j-1} \end{cases} \tag{3.9}$$

onde $d(i,j)$ define o custo associado ao nó $t(i)$ e $d(j)$ e pode ser calculado por

$$d(i,j) = |t(i) - r(j)|. \tag{3.10}$$

O melhor caminho dentro da tabela é definido por

$$W = w_1, w_2, \ldots, w_K, \tag{3.11}$$

onde w_k representa um par ordenado obtido de $N \times M$,

$$w_k = (i;j)_k. \tag{3.12}$$

Para determinar W, percorre-se o caminho com o menor custo global associado (soma dos células de $D_{i,j}$ percorridas).

As restrições a seguir são impostas a W. Dependendo da aplicação, outras restrições podem ser consideradas.

Condição de fronteira O caminho começa em uma diagonal da tabela e termina na oposta, $w_1 = (1,1)$ e $w_k = (m;n)$;

Continuidade Dado $w_k = (a;b)$ então $w_{k-1} = (a';b')$, onde $a - a' \leq 1$ e $b - b' \leq 1$.

Monotonicidade Dado $w_k = (a;b)$ então $w_{k-1} = (a';b')$, onde $a - a' \geq 0$ e $b - b' \geq 0$.

3.5.2 O modelo oculto de Markov: abordagem estatística

Em seu clássico artigo/tutorial sobre o modelo oculto de Markov e suas aplicações em reconhecimento da fala, Rabiner [20] afirma que devemos assumir que o sinal candidato para representação pelo HMM "pode ser bem caracterizado por um processo paramétrico aleatório, e os parâmetros do processo estocástico podem ser determinados de uma

maneira precisa e bem-definida".

O sinal musical compartilha muitas características com o sinal da fala. Shiffrin foi um dos primeiros a propor uma abordagem de reconhecimento de melodias baseada no HMM.

3.5.3 Cadeias de Markov

Cadeias (ou modelos) de Markov são ferramentas estatísticas usadas para representar sistemas cuja evolução no tempo, entre os instantes $t_i = a$ e $t_f = b$, é descrita por uma sequência de estados $Q = q_a, q_{a+1}, ..., q_{b-1}, q_b$, onde $q_i \in S$ e $S = s_1, s_2, ..., s_N$ é a sequência de N estados possíveis para o sistema, ou o *espaço amostral* da variável Q.

A probabilidade do próximo estado do sistema corresponder a um estado s_j dado que o sistema se encontre em um estado s_i, definida como $a_{i,j}$, é assim expressa:

$$a_{i,j} = P[q_{t+1} = S_j | q_t = S_i], \tag{3.13}$$

onde i e j são valores quaisquer entre 1 e N.

Considere também as seguintes propriedades:

$$P[q_t = S_j | q_{t-1} = S_i, q_{t-2} = S_k, ...] = P[q_t = S_j | q_{t-1} = S_i] \qquad \text{e} \tag{3.14}$$

$$\sum_{j=1}^{N} a_{i,j} = 1 \qquad [\forall j \in X | X = \{1, 2, 3, ..., N\}], \tag{3.15}$$

onde $a_{i,j} \geq 0$..

É possível definir também a *distribuição de probabilidades do estado inicial*, π, que determina, para cada estado s_i, a probabilidade π_i do sistema iniciar no respectivo estado.

Considere o seguinte exemplo simples:

Imagine uma situação hipotética em que o tempo (no contexto climático) assume, para cada unidade de tempo correspondente a um dia, um dos estados entre o conjunto finito de estados $S = \{Chuvoso, Nublado, Ensolarado\}$. E que a probabilidade de termos um dia de tempo j dado que, no dia anterior, o tempo i tenha ocorrido é dada pela *distribuição de probabilidades da transição de estados*, representada como segue

$$A = \{a_{i,j}\} = \begin{pmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,N} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,N} \\ \vdots & \vdots & \ddots & \vdots \\ a_{N,1} & a_{N,2} & \cdots & a_{N,N} \end{pmatrix}, \tag{3.16}$$

de onde assumiremos, para o exemplo dado (adaptado de [20]), os seguintes valores

$$A = \{a_{i,j}\} = \begin{pmatrix} a_{Chuv,Chuv} & a_{Chuv,Nub} & a_{Chuv,Sol} \\ a_{Nub,Chu} & a_{Nub,Nub} & a_{Nub,Sol} \\ a_{Sol,Chuv} & a_{Sol,Nub} & a_{Sol,Sol} \end{pmatrix} = \begin{pmatrix} 0.4 & 0.3 & 0.3 \\ 0.2 & 0.6 & 0.2 \\ 0.1 & 0.1 & 0.8 \end{pmatrix}. \tag{3.17}$$

A partir desses dados, podemos extrair informações como a probabilidade do sistema permanecer d dias seguidos no estato s_i

$$p_i(d) = P(q_2 = s_i, q_3 = s_i, ..., q_{d-1} = s_i, q_d = s_{j \neq i} | q_1 = s_i) = (a_{i,j}^{d-1}(1 - a_{i,j}), \tag{3.18}$$

ou a quantidade média de dias que o sistema permance em um único estado s_i

$$d_i = \sum_{d=1}^{\infty} dp_i(d) = \frac{1}{1 - a_{ij}} \tag{3.19}$$

Isso resume a ideia geral de um modelo de Markov.

Para estender a aplicabilidade do modelo para diversas aplicações de interesse, no entanto, precisamos incluir um novo conceito que representa a incerteza inerente aos estados de sistemas mais complexos.

Shiffrin aponta que outros trabalhos anteriores ao seu aplicaram o modelo de Markov na recuperação musical baseada em conteúdo. No entanto, esse modelo não é capaz de lidar com erros de execução da melodia como o modelo descrito a seguir.

3.5.4 O modelo oculto de Markov

No modelo oculto de Markov, um novo processo estocástico é adicionado. Esse novo processo esconde o anterior — que determina a transição de estados — ocultando do ambiente a sequência de estados pela qual um determinado sistema passou durante um dado intervalo de tempo.

Nesse caso, o sistema emite um sinal de símbolos observáveis, que podem ser, por exemplo, vetores extraídos do sinal da fala caracterizando um fonema, ou uma descrição da transição de notas, como proposto por Shiffrin.

Uma nova distribuição de probabilidades deve ser considerada. A *distribuição de probabilidade da observação de símbolos*, definida como:

$$b_{S_i}(v_k) = P[o_t = v_k | q_t = S_i] = P[v_k \text{ em } t | q_t = S_i], \begin{matrix} 1 \leq & i & \leq N \\ 1 \leq & k & \leq M, \end{matrix} \tag{3.20}$$

representa a probabilidade de observação de um símbolo v_k do conjunto de M símbolos $V = \{v_1, v_2, v_3, ..., v_M\}$ no estato s_i.

A partir desses dados, é necessario resolver as seguintes questões:

1. Dada uma sequência de símbolos, O, observada num determinado intervalo de tempo, como computar eficientemente sua probabilidade da observação dado um modelo λ, ou seja, qual valor de $P(O|\lambda)$?
2. Dado um modelo e uma sequência de símbolos observados, como determinar a sequência de estados que melhor *explica* a sequência (mais provável, por exemplo)?
3. Como ajustar os parâmetros de um modelo para maximizar a probabilidade descrita na questão 1?

Abordagens para a solução desses *três problemas clássicos* do HMM, incluindo os algoritmos do backward-forward e da decodificação dos estados ocultos de *Viterbi*, podem ser encontrados no tutorial de Rabiner.

3.5.5 Melodias como o modelo oculto de Markov

Para a representação das notas Shiffrin fez as seguintes definições:

1. A transição entre as notas n e $n+1$ é descrita pela tupla <deltaPitch, IOIratio>
2. $deltaPitch_n$ é a diferença de *pitch* entre as notas n e $n+1$;
3. O intervalo entre instantes iniciais de notas, IOI_n, é a diferença entre os instantes iniciais das notas n e $n+1$; e

4. $IOIratio_n$ é $\frac{IOIn}{IOI_{n+1}}$. Para a última transição da sequência, $\frac{IOI_n}{durao_{n+1}}$.

A *sequência de transição das notas*, como definidas por Shiffrin, correspondem a sequência de símbolos observada no HMM.

O modelo é construído, inicialmente, a partir de uma versão MIDI da música e um estado é construído para cada tupla $(detalPitch_n, IOIratio_n)$ dos símbolos observados. Um exemplo de transcrição musical acompanhada do modelo de Markov construído para representá-la aparece na Figura 8.

As probabilidades de transição entre estados são obtidas diretamente da distribuição observada na *sequência de transições* da melodia.

Para estimar a *distribuição de probabilidade de observação*, no entanto, é necessário dispor de um conjunto de observações (gravações de usuários para a melodia) para treino.

Shiffrin discretizou a diferença de *pitch* em 25 intervalos — 2 intervalos para cada semiton de uma oitava — e a razão entre intervalos de instantes iniciais de notas em 27 intervalos tomados em escala logarítmica, o que limitou o alfabeto de símbolos observáveis para $25 * 27 = 675$ símbolos. A escolha desses intervalos foi baseada em observações empíricas.

A partir do conjunto de treino, é possivel treinar o modelo oculto discreto de Markov usando, por exemplo, o procedimento descrito por Rabiner.

3.5.6 Outras abordagens

O algoritmo de *escalonamento linear*, LS, criado por Jang et al. [21], superou o algoritmo de Viterbi em seu sistema de QBH baseado em HMM. Foram os primeiros a abandonar a abordagem *bottom-up* adotada pelos algoritmos tradicionais baseados em programação dinâmica, como o próprio Viterbi e o DTW.

Depois dele, Wu [22] propôs uma nova técnica baseada no LS, o *alinhamento recursivo*, que apresenta uma estratégia de reconhecimento dividida em níveis. Primeiro, tenta casar a melodia no contorno melódico como um todo. Em seguida, prossegue em sucessivas etapas fazendo ajustes locais.

Mais recentemente, outra abordagem [23] de baixo nível (sem segmentação de notas) apresentou bons resultados na extração e comparação do *contorno de pitch* de melodias. Através de coeficientes da *transformada wavelet*, foi proposta uma nova representação

para melodias invariante a características pessoais de entonação e rítmo.

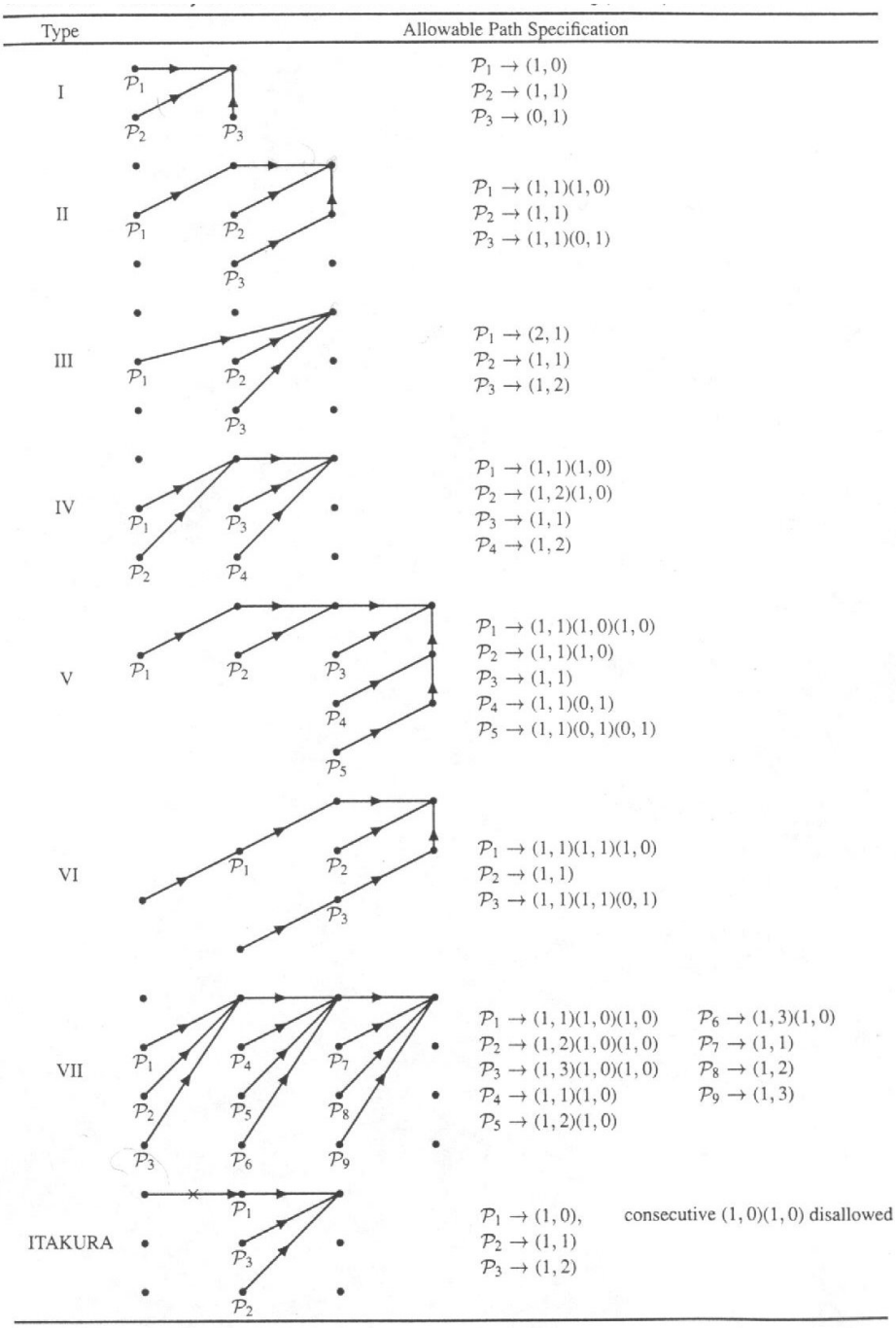

Figura 7: Caminhos do *dynamic time warping*

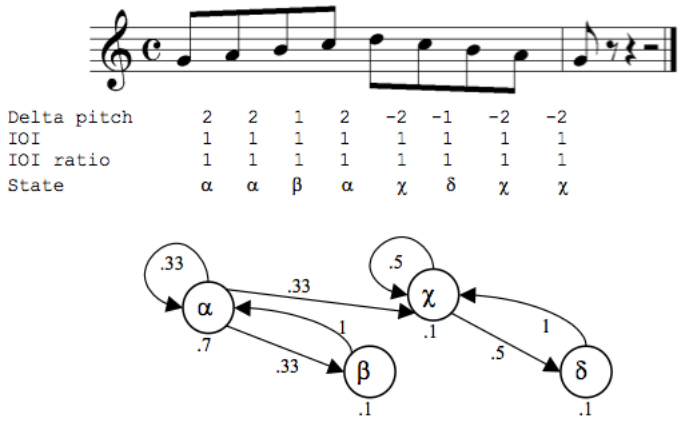

Figura 8: Um exemplo de representação da melodia usando HMM (Shiffrin)

4 O projeto de QBH: framework e base de dados

4.1 Um framework de QBH na linguagem Java

Com base nos estudos sobre o tema, uma ferramenta de reconhecimento de melodias foi construída usando a linguagem de programação Java.

Embora já existam *frameworks* para trabalhar com *recuperação de informação musical baseada em conteúdo*, como o MARSYAS (marsyas.info), a implementação foi um recurso usado para melhor compreender as técnicas envolvidas e as dificuldades comuns de implementação. Todas as técnicas apresentadas foram implementadas.

O sistema adota uma arquitetura modular. Dessa forma, é possível adicionar novos recursos ao sistema, como uma nova estratégia de segmentação ou um novo PDA, de forma transparente.

Os principais módulos do sistema são:

Interface de captura e registro de áudio (AudioCaptureTool) é responsável por capturar o sinal acústico a partir do microfone do sistema. Esse módulo utiliza um módulo auxiliar, *AudioIO*, que converte o formato padrão de representação do sinal, de *array* de *bytes* com sinal, para *array* de valores em ponto flutuante entre 0 e 1, além de realizar a conversão inversa e permitir que arquivos de áudio sejam salvos e lidos do disco rígido. As classes *SoundPlayer* e *SoundRecorder* apresentam, respectivamente, funções com chamadas assíncronas para manipular a reprodução e gravação do som. Nessa última, é possível alterar facilmente as configurações do formato utilizado, como a taxa de amostragem.

Gerador de gráficos (Plotting) pode gerar diversos gráficos a partir do sinal de entrada, como sua forma de onda com ou sem marca da segmentação sobreposta (Figura 9), o contorno melódico (Figura 10), ou o espectro de uma determinada porção

do sinal (Figura 11). Alguns dos controles que podem ser usados para manipular gráficos são mostrados na Figura 12.

Estratégia de segmentação (SegmentationStrategy) é o módulo responsável por identificar os limites (início e fim) das notas no sinal de áudio. A estratégia de segmentação se comunica com o *Identificador de Pitch* e o *Profilador do Sinal*. Dessa forma, ele pode obter do primeiro informações a respeito da frequência fundamental observada em qualquer segmento do sinal; e do segundo informações associadas a intensidade do sinal, como o nível de energia média de um quadro e a taxa sinal-ruído (SNR). O módulo de segmentação guarda registros das notas detectadas e pode retornar essa informação para qualquer módulo que solicitar.

Profilador de sinal (EnergyProfiler) computa algumas propriedades do sinal relacionadas à sua intensidade, como o valor RMS da amplitude, a taxa SNR e o grau de silêncio de cada segmento.

Identificador de Pitch (PitchTracker) representa outra classe abstrata do sistema. Não implementa a estratégia que detecta o *pitch* do sinal (como a autocorrelação ou o espectro do produto harmônico, por exemplo), apenas define um método abstrato responsável pela tarefa. A interface do método abstrato, *estimatePitch*, retorna a frequência estimada para uma dada janela do sinal. O tipo da janela, ie. retangular, *hanning*, *hamming*, etc., é representado por uma *enumeração*: *WindowFunction*. Nesse módulo, também são implementadas estratégias para discretizar a frequência calculada e para suavizar o contorno de frequência do sinal. retornar as notas musicais identificadas pelo nome e oitava, por exemplo *C4*— Dó na quarta oitava, além de estratégias para estimar e/ou transpor o tom de uma melodia.

Identificador de melodias (SongMatcher) é responsável por construir uma representação para a música a partir das informações extraídas da segmentação, ou do contorno de frequência. Existem duas implementações dessa interface: *StatisticalSongMatcher*, que implementa o modelo semelhante ao de *Shiffrin*, porém com o *deltaPitch* assumindo valores entre *-12* a *12* para contemplar intervalo descendentes; e o PitchContourMatcher, que aplica o DTW sobre o contorno de frequência de duas melodias para estimar a semelhança entre elas.

Além disso, foram implementados algoritmos para aplicar filtros de frequência, *center-clipping*, normalização e reamostragem do sinal.

A implementação da FFT é apresentada no apêndice.

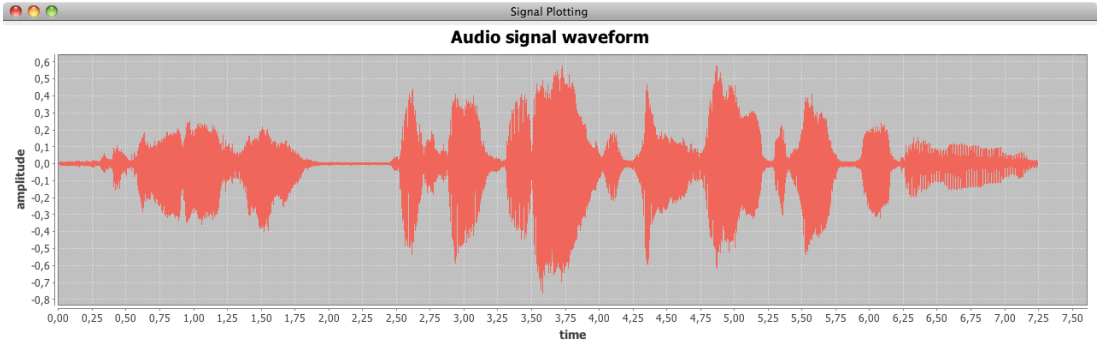

Figura 9: Gráfico gerado pelo programa: forma de onda

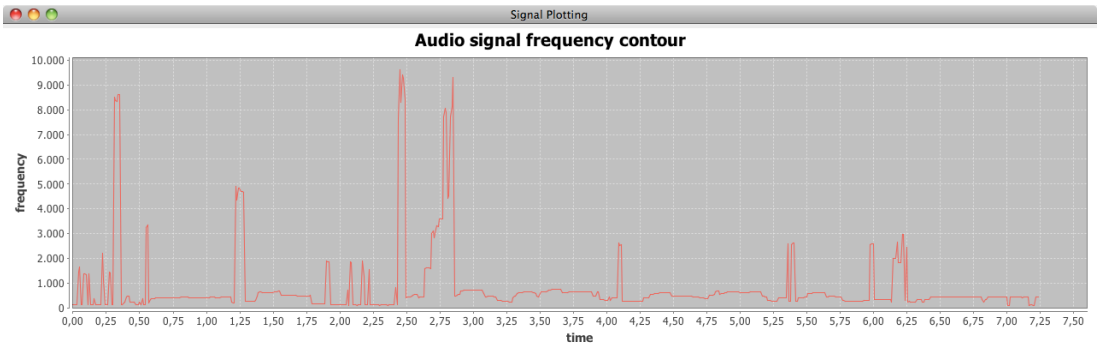

Figura 10: Gráfico gerado pelo programa: contorno de frequência

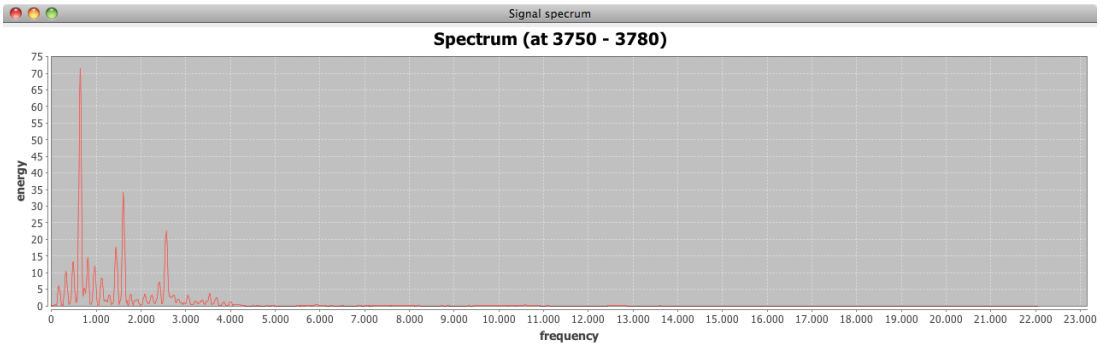

Figura 11: Gráfico gerado pelo programa: espectro de uma janela

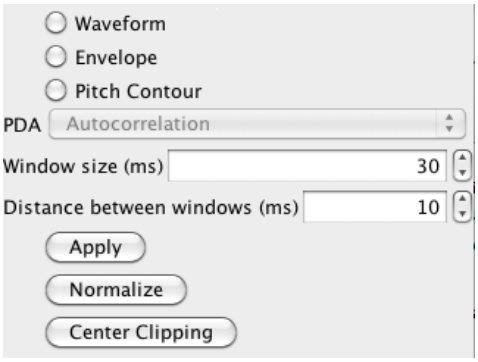

Figura 12: Painel de controles do gerador de gráficos

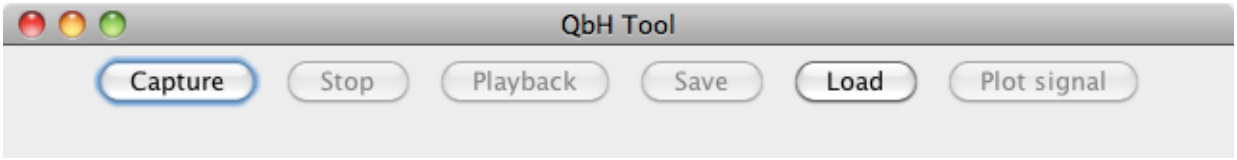

Figura 13: Tela principal do programa de reconhecimento

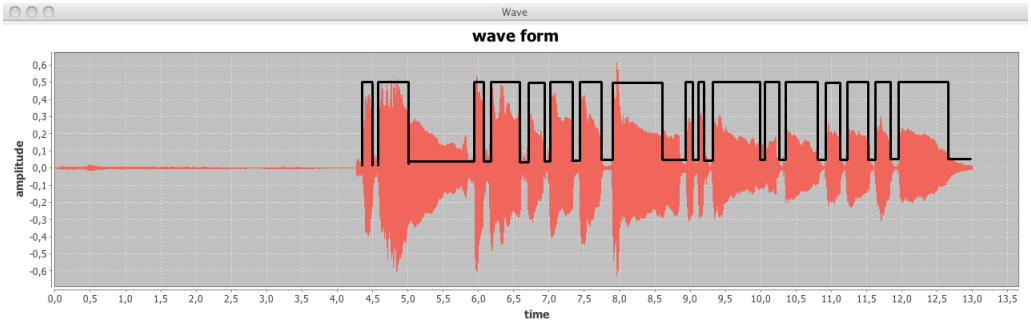

Figura 14: O resultado da detecção das notas

A interface gráfica do usuário é simples. A tela inicial, exibida na Figura 13, apresenta as opções de capturar áudio do microfone, carregar um arquivo do disco, reproduzir uma gravação capturada ou carregada, ou salvar uma gravação.

O áudio é capturado usando formato PCM, a 44100 amostras por segundo, 16 bits por amostra e canal único. Um valor alto foi escolhido para a amostragem para evitar os problemas de aliasing com as técnicas de detecção de *pitch*.

A opção de salvar permite que o usuário salve a melodia cantada na pasta de sua escolha. Ao salvar a gravação, porém, será salvo no diretório do programa arquivos com a descrição da melodia (atualmente um arquivo com extensão .mel e outro com extensão .ly — do lilypond (www.lilypond.org)) e uma entrada na base de dados do programa será criada, associando o arquivo de áudio gravado aos arquivos de descrição.

Ao abrir um arquivo do tipo *descrição melódica* — identificados pela extensão *.mel* — o programa exibe uma outra janela mostrando um gráfico com a forma de onda do áudio (carregado automaticamente) e funções retangulares delimitando as notas detectadas, como mostrado na Figura 14.

O formato de arquivo *mel* é composto por duas linhas de informação. Na primeira, existe um código Parson para o contorno melódico do sinal. Na segunda, a sequência de notas detectadas na melodia são listadas separadas por vírgula. Cada nota é representada pela seguinte tupla:

(pitch, onset, offset)

onde *pitch* é a frequência física (em Hertz) detectada, *onset* é a primeira amostra da nota, e *offset* a última amostra.

4.1.1 O pré-processamento do sinal e outras considerações

Ao capturar uma melodia através do sistema, é recomendável que o usuário permaneça em silêncio durante os 5 a 10 primeiros segundos da gravação para que o sistema possa computar o perfil do som ambiente.

Após a captura do sinal de áudio, o sistema calcula a taxa de sinal-ruído da gravação como descrito na seção de segmentação no capítulo anterior. Caso esta seja alta demais, o sistema informa ao usuário que a gravação não pode ser utilizada por causa da interferência do som ambiente. Nesse caso, o usuário poderá ainda salvar a gravação, mas o sistema não extrai qualquer informação a respeito da melodia, nem registra o arquivo na base de dados.

Atualmente, antes de salvar a descrição da melodia na base de dados, o sistema transpõe seu tom para que a média tonal da música seja próxima a média de tons das músicas na base de dados. Se ainda não houverem músicas na base de dados, o sistema simplesmente computa a média tonal da melodia atual, e salva na base dados junto com a música. Essa informação será usada para transpor o tom da próxima música apresentada, na tentativa de eliminar dessa forma a diferença de tom entre versões da mesma melodia, possivelmente cantadas por usuários diferentes.

A média de tons da música é calculada usando simplesmente a média aritmética das frequências de todas as notas detectadas:

$$M_{TOM} = \frac{\sum_{i=1}^{N} pitch(i)}{N}, \tag{4.1}$$

onde N é o número de notas da melodia e $pitch$(i) a frequência fundamental da *i-ésima* nota.

Para criar um modelo oculto de Markov a partir de uma arquivo MIDI, o sistema usa o módulo perl MIDI Splitter (http://apps.linuxaudio.org/apps/all/midi_splitter)

4.1.2 A apresentação dos resultados

O resultado da execução do algoritmo de reconhecimento não é apresentado ao usuário pela interface gráfica. Atualmente, ao salvar a gravação, é criado no diretório do programa um arquivo CSV com uma tabela NxN, em que N é o número de melodias armazenadas na base de dados. Nessa tabela, para cada melodia i, o resultado do algoritmo DTW entre i e as demais melodias representadas por j, tal que $j \neq i$, é salva na célula da linha i e coluna j. A cada melodia salva, o programa carrega a tabela na memória, e a recria adicionando as informações para a nova melodia.

Se o usuário desejar conferir a transcrição de uma melodia salva no sistema, poderá compilar o arquivo .ly salvo no diretório de dados do programa usando o lylipond. Este criará um arquivo PDF com a partitura das notas detectadas e um arquivo MIDI que poderá ser reproduzido em um tocador com sintetizador de instrumentos.

Na API do *framework* desenvolvido também existe um método para reproduzir através da saída de áudio padrão a melodia transcrita, embora esse recurso apresente baixa qualidade de som.

4.2 A base de dados do sistema

Para avaliar os algorimos implementados, uma pequena base de dados foi criada. Seis voluntários gravaram em torno de 15 músicas cada. As músicas foram escolhidas pelos próprios usuários em uma lista de 50 músicas.

A lista de músicas, por sua vez, foi criada para contemplar diversos gêneros e origens nacionais e internacionais. Muitas delas foram verificadas na base do *Midomi*, o mais popular sistema de recuperação musical baseado em reconhecimento de melodias. Esse sistema consta com uma base musical construída por usuários de todo o mundo e em constante crescimento.

Foi também solicitados aos usuários que, se possível, gravassem duas versões de cada música: *cantada* e *cantarolada*.

A lista de músicas cantadas consta no apêndice C.

5 Avaliações e comparações dos sistemas atuais de QBH

5.1 O desempenho dos sistemas atuais

5.1.1 Midomi

O *Midomi* é uma rede social e sistema de busca musical baseda em conteúdo. A base de dados é criada pelos próprios usuários, sem nenhuma informação prévia, o que significa que o sistema funciona sem treinamento e sem a necessidade de uma versão transcrita da música (como arquivos MIDI).

Para avaliar o sistema, os voluntários cantaram quinze melodias selecionadas de uma lista de cinquenta músicas. A tabela 3 mostra o nível de acerto, para classificações em primeiro, segundo e terceiro lugar. Apenas uma música foi mostrada na quinta opção e, somadas com as músicas não encontradas contam 14%.

A base de dados do *Midomi* está crescendo rapidamente. Tanto os resultados quanto o tempo de consulta foram satisfatórios.

5.1.2 Musipedia

O nível de reconhecimento do *Musipedia* não foi avaliado, pois a base de dados desse sistema não apresenta temas em comum com a base de dados coletada.

Quanto à segmentação, o sistema exibe o resultado dessa operação, permitindo que o usuário ajuste manualmente os atributos da notas. Esse recurso é útil para refinar as

Tabela 3: Resultados da avaliação do Midomi (valores em porcentagem)

Perfil de usuário	Primeira opção	Segunda opção	Terceira opção
Cantada	75	86	86
Cantarolada	72	84	86

buscas, desde que a quantidade de erros na segmentação seja pequena, o que não aconteceu na maioria dos casos.

Apenas as músicas com notas bem separadas, cantadas com sílabas que favorecem a subida rápida de energia da região de ataque da nota e maior estabilidade na região da sustentação obtiverem sucesso na segmentação superior a 70% das notas.

Uma nota foi considerada corretamente segmentada se o *pitch* da nota da melodia corresponder ao da nota detectada, e o onset (início) da nota detectada acontecer antes da metade da duração da nota real.

A verificação da frequência, no entanto, foi feita intuitivamente, e a verificação da duração da nota foi medida manualmente direto na forma de onda. Além disso, notas duplicadas ou fantasmas foram desconsideradas.

O sistema apresenta outros recursos de busca baseada em conteúdo, como *query by tapping*, que permite ao usuário localizar registros musicais a partir de batidas na barra de espaço na qual o sistema tenta identificar um registro musical a partir do rítmo

5.2 A ferramenta proposta: comparações

Os resultados obtidos com a ferramenta implementada ficaram bem abaixo dos resultados do *Midomi*. No entanto, quanto à segmentação, principalmente com notas cantadas, o sistema mostrou-se mais eficiente do que o *Musipedia*. Erros de segmentação, no entanto, podem variar dependendo do usuário e do estilo da música e do canto. Para músicas cantaroladas usando fonemas específicos, como /ta/ e /da/, esse valor ficou entre 70 e 80%, e abaixo dos 70% para músicas cantadas com as respectivas letras. Quando a música é assobiada, o sucesso da segmentação pode, facilmente, superar 90% (em comparação à transcrição manual das notas), desde as informações do contorno de frequência sejam usadas diratamente na segmentação.

O HMM não foi testado com muito rigor. Apenas seis músicas em formato MIDI foram processadas e transformadas em modelos de Markov. Em seguida, apesar do pequeno número de repetições de uma mesma música, os modelos foram treinados, seguindo a ideia geral do algoritmo EM, *expectation-maximization*, descrita em [20].

Outros três usuários, que não participaram da geração da base de dados, consultaram as seis músicas distintas das quais consistiam a base de dados. Em média, quatro músicas foram apresentadas como a primeira opção. As outras duas músicas foram apresentadas

na segunda opção para dois usuários e, para o terceiro, como segunda e terceira opção. Em todos os casos, as músicas foram cantaroladas em trechos conhecidos da música, não necessariamente a partir do início.

Em relação ao DTW, com uma base de dados um pouco maior, composta por vinte músicas, foi observado acertos acima de 60% no primeiro lugar, 70% para o segundo, e próximo de 90% no terceiro lugar. Novamente, os mesmos três usuários consultaram o sistema. Um deles obteve êxito póximo de 100% no *top-3*. Para consultar o sistema, o usuário catava a música desejada seguindo a própria letra. É possível que o Midomi utilize técnicas de reconhecimento da fala para identificar porções relevantes da letra, auxiliando a busca. Aqui, o uso da letra é relevante apenas para manter a influência de determinados fonemas presentes na letra sobre a entonação da melodia.

Foram detectados problemas, porém, em ambas as implementações de PDA. Para autocorrelação, o índice de harmônicas presentes foi alto. Enquanto que, no HPS, erros foram causados, provavelmente, pelo efeito de *aliasing*. Usar simplesmente o máximo do espectro do sinal para estimar a frequência mostrou-se, no entanto, particularmente eficaz com assobios, produzindo contorno de frequência bem definidas e fáceis de processar.

6 Conclusões

O presente trabalho apresentou algumas questões associadas ao reconhecimento de melodias e implementou um framework com técnicas básicas para aplicações de QBH.

Atualmente, sistemas como o *Midomi* apresentam implementações do estado da arte em recuperação de informação musical baseada em conteúdo. O sistema foi avaliado, mostrando que é possível criar sistemas de reconhecimento de melodias com desempenho, em tempo de execução e precisão nos resultados, satisfatórios.

Já era esperado, no entanto, que o sistema implementado nesse trabalho ficasse aquém dos resultados dos sistemas atuais. O projeto atual será portado para uma linguagem com maior desempenho para aplicações em tempo real, como C++, e novos módulos serão implementados.

O framework será utilizado na implementação de um ferramenta de busca musical de tempo real para gerenciar coleções pessoais de músicas digitais, além de permitir que uma música não encontrada na coleção seja procurada na internet. Esse módulo *web*, por sua vez, poderá ser utilizado por sites de venda de músicas digitais.

Referências

[1] MIREX-WIKI. Disponível em: <http://www.music-ir.org/mirex/wiki/>.

[2] Musipedia. Disponível em: <http://www.musipedia.org/>.

[3] Mimodi. Disponível em: <http://www.midomi.com/>.

[4] BHATTACHARJEE, S.; GOPAL, R. D.; SANDERS, G. L. Digital music and online sharing: software piracy 2.0? *Commun. ACM*, ACM, New York, NY, USA, v. 46, p. 107–111, July 2003. ISSN 0001-0782. Disponível em: <http://doi.acm.org/10.1145/792704.792707>.

[5] CASEY, M. et al. Content-based music information retrieval: Current directions and future challenges. *Proceedings of the IEEE*, v. 96, n. 4, p. 668 –696, april 2008. ISSN 0018-9219.

[6] CHANG, S.-F.; SIKORA, T.; PURL, A. Overview of the mpeg-7 standard. *Circuits and Systems for Video Technology, IEEE Transactions on*, v. 11, n. 6, p. 688 –695, jun 2001. ISSN 1051-8215.

[7] CHELLA, M. T. Sistema para classificacao e recuperacao de conteudo multimidia baseado no padrao mpeg7. 2004.

[8] GHIAS, A. et al. Query by humming: musical information retrieval in an audio database. In: *Proceedings of the third ACM international conference on Multimedia*. New York, NY, USA: ACM, 1995. (MULTIMEDIA '95), p. 231–236. ISBN 0-89791-751-0. Disponível em: <http://doi.acm.org/10.1145/217279.215273>.

[9] BAEZA-YATES; NAVARRO, R. G. Faster approximate string matching. *Algorithmica*, Springer New York, v. 23, p. 127–158, 1999. ISSN 0178-4617. 10.1007/PL00009253. Disponível em: <http://dx.doi.org/10.1007/PL00009253>.

[10] MCNAB, R. J. et al. Towards the digital music library: tune retrieval from acoustic input. In: *Proceedings of the first ACM international conference on Digital libraries*. New York, NY, USA: ACM, 1996. (DL '96), p. 11–18. ISBN 0-89791-830-4. Disponível em: <http://doi.acm.org/10.1145/226931.226934>.

[11] HANDEL, S. *Listening: An Introduction to the Perception of Auditory Events*. MIT Press, 1993. (Bradford Books). ISBN 9780262581271. Disponível em: <http://books.google.com/books?id=-cuDPwAACAAJ>.

[12] CHANG, S.-F.; GAO, M.-Y. Query by tapping: A new paradigm for content-based music retrieval from acoustic input. In: SHUM, H.-Y.; LIAO, M.; CHANG, S.-F. (Ed.). *Advances in Multimedia Information Processing, PCM2001*. [S.l.: s.n.], 2001, (Lecture Notes in Computer Science, v. 2195). p. 590–597.

[13] ZHU, Y.; SHASHA, D. Warping indexes with envelope transforms for query by humming. In: *Proceedings of the 2003 ACM SIGMOD international conference on Management of data*. New York, NY, USA: ACM, 2003. (SIGMOD '03), p. 181–192. ISBN 1-58113-634-X. Disponível em: <http://doi.acm.org/10.1145/872757.872780>.

[14] PLACK, C.; OXENHAM, A.; FAY, R. *Pitch: neural coding and perception*. Springer, 2005. (Springer handbook of auditory research). ISBN 9780387234724. Disponível em: <http://books.google.com/books?id=n6VdlK3AQykC>.

[15] GOLD, B.; RABINER, L. Parallel processing techniques for estimating pitch periods of speech in the time domain. *The Journal of The Acoustical Society of America*, v. 26, p. 446–448, 1969.

[16] BELLO, J. et al. A tutorial on onset detection in music signals. *Speech and Audio Processing, IEEE Transactions on*, v. 13, n. 5, p. 1035 – 1047, sept. 2005. ISSN 1063-6676.

[17] HAUS, G.; POLLASTRI, E. An audio front end for query- by-humming systems. In: *International Symposium on Music Information Retrieval*. [S.l.: s.n.], 2001. p. 65–72.

[18] SHIFRIN, J. et al. Hmm-based musical query retrieval. In: *Proceedings of the 2nd ACM/IEEE-CS joint conference on Digital libraries*. New York, NY, USA: ACM, 2002. (JCDL '02), p. 295–300. ISBN 1-58113-513-0. Disponível em: <http://doi.acm.org/10.1145/544220.544291>.

[19] DANNENBERG, R. et al. The musart testbed for query-by-humming evaluation. *Computer Music Journal*, v. 28, p. 34–48, 2004.

[20] RABINER, L. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, v. 77, n. 2, p. 257 –286, feb 1989. ISSN 0018-9219.

[21] JANG, J.-S.; GAO, M.-Y. A top-down approach to melody match in pitch con. 2006.

[22] WU, X. et al. A query-by-singing system based on dynamic programming. 2000.

[23] JEON, W.; MA, C.; CHENG, Y. M. An efficient signal-matching approach to melody indexing and search using continuous pitch contours and wavelets. p. 65–72, 2009.

APÊNDICE A – As notas musicais e o pitch

A tabela 4 mostra a frequência fundamental e comprimento de onda associado a cada nota musical na primeira oitava.

A frequência de cada nota é dobrada na oitava seguinte. Por exemplo, a nota lá (A) na segunda oitava tem frequência $27.5Hz * 2 = 55Hz$. Na terceira oitava, sua frequência é $55Hz * 2 = 110Hz$, e assim sucessivamente.

A escala cromática, composta pelas doze notas de uma oitava qualquer, apresenta entre cada nota um intervalo de um semitom — menor intervalo musical (ou algo próximo do menor intervalo de *pitch* que conseguimos distinguir). Da mesma forma, a primeira nota de cada oitava encontra-se exatamente $\frac{1}{2}$ tom acima da última nota da oitava anterior.

Portanto, a frequência das notas musicais obedecem uma progressão geométrica cuja razão r é dado por

$$r = 2^{\frac{1}{12}}. \tag{A.1}$$

Nota	Frequência (Hz)	Comprimento da onda (cm)
C	16.35	2100
C#/Db	17.32	1990
D	18.35	1870
D#/Eb	19.45	1770
E	20.60	1670
F	21.83	1580
F#/Gb	23.12	1490
G	24.50	1400
G#/Ab	25.96	1320
A	27.50	1250
A#/Bb	29.14	1180
B	30.87	1110

Tabela 4: Frequência das notas musicais na primeira oitava (0)

APÊNDICE B – A Transformada Rápida de Fourier

A *transformada de Fourier* é um algoritmo eficiente para computar a *transformada discreta de Fourier* usado em diversas aplicações, incluindo multiplicação de números inteiros arbitrariamente grandes, e análise de sinais de áudio e vídeo no domínio da frequência.

A transformada de uma função discreta $x(t)$ pode ser calculada usando o algoritmo da DFT, definido como

$$X_k = \sum_{n=0}^{N-1} x_n e^{-i2\pi k \frac{n}{N}} \quad \text{k} = 0, \ldots, \text{N-1}, \tag{B.1}$$

onde N é o número de amostras.

No entanto, avaliar essa função diretamente envolve $O(N^2)$ operações, enquanto que uma implementação de algoritmo da FFT apresenta complexidade não maior do $O(n \log n)$.

Uma das implementações mais simples da FFT, conhecida como raiz-2 *Cooley-Tukey*, consiste em dividir o vetor em elementos ímpares e pares e calcular recursivamente a DFT de ambas as partes (usando o raiz-2 DIT).

A transfomada é definida como uma combinação das duas transformadas menores da seguinte maneira:

$$X[n] = \begin{cases} E[n/2] + O[n/2]w_{n/2} & \text{se n} < \text{N}/2 \\ E[n/2] - O[n/2]w_{n/2} & \text{se n} \geq \text{N}/2 \end{cases} \tag{B.2}$$

onde E e O são, respectivamente, a transformada discreta (calculada recursivamente) dos elementos pares e ímpares, e w_i é definido como

FFT.java

```
public static double[][] fft(double[][] x) {
        return fft(x, x.length);
}

private static double[][] fft(double[][] x, int d) {
        if(d == 1)
                return x;
        double[][] tempF = fft(x, d/2);
        double[][] f = new double[x.length][2];
        int s = x.length / d;
        int blocksTotal = x.length / (s*2);

        int i = 0;
        for(int b = 0; b < blocksTotal; ++b) {
                int start = b * (s * 2);
                int end = b * (s * 2) + s;

                for(int n = start; n < end; ++n) {

                        double kth = -2 * b * Math.PI / d;
                        double[] wk = new double[] { Math.cos(kth), Math.sin(kth) };
                        double a0 = tempF[n][0], a1 = tempF[n][1],
                                        b0 = tempF[n + s][0], b1 = tempF[n + s][1];
                        f[i][0] = a0 + (wk[0] * b0 - wk[1] * b1);
                        f[i][1] = a1 + (wk[0] * b1 + wk[1] * b0);
                        f[i + x.length/2][0] = a0 - (wk[0] * b0 - wk[1] * b1);
                        f[i + x.length/2][1] = a1 - (wk[0] * b1 + wk[1] * b0);
                        ++i;
                }

        }
        return f;
}
```

-U:**- FFT.java All L1 (Java/l Abbrev)----------

Figura 15: Algoritmo da FFT

$$w_c = e^{\frac{-2\pi cj}{N}}. \tag{B.3}$$

O índice $N/2$ foi usado na equação B.2 por causa da divisão do sinal original em duas partes com a metade do número de amostras.

A Figura 15 mostra uma implementação *online* desse algoritmo.

APÊNDICE C – A lista de músicas sugeridas para gravação

1. The look of love (Burt Bacharack e Hal David)
2. Yesterday (Paul McCartney)
3. Hey Jude (Paul McCartney)
4. I´ve got under my skin (Cole Porter)
5. ´s wonderful (George Gershwin e Ira Gershwin)
6. I say a little prayer for you (Diana King e Dionne Warwick)
7. Livin´la vida loca (Puerto Rican)
8. I fell good (James Brown)
9. La vie em rose (Édith Piaf e Louis Gugliemi)
10. C´est si bom (Henri Betti e André Hornez)
11. La barca (Roberto Cantoral)
12. La puerta (Luis Demetrio)
13. Besame mucho (Consuelo Velásquez)
14. Quizas, quizas, quizas (Osvaldo Farres)
15. Que sera, sera (Whatever wil be, will be) (Jay Livingston e Ray Evans)
16. When I fall in love (Victor Young e Edward Heyman)
17. Only you (And you alone) (B. Ram e A. Ram)

18. Killing me softly with his song (Charles Fox e Norman Gimbel)
19. O que tinha que ser (Tom Jobim e A. Oliveira)
20. Anos dourados (Tom Jobim e Chico Buarque)
21. Último pau de arara (Venâncio, Corumba e J. Guimarães)
22. Olha pro céu (Luiz Gonzaga e José Fernandes)
23. Explode coração (Gonzaguinha)
24. Grito de alerta (Gonzaguinha)
25. Detalhes (Roberto Carlos e Erasmo Carlos)
26. Negue (Adelino Moreira e Enzo de Almeida Passos)
27. Encontro das águas (Jota Maranhão e Jorge Vercillo)
28. Alguém me disse (Evaldo Gouveia e Jair Amorim)
29. Universo do teu corpo (Taiguara)
30. Matriz ou filial (Lúcio Cardim)
31. Marina (Dorival Caymmi)
32. A vizinha do lado (Dorival Caymmi)
33. Leva meu samba (Ataulfo Alves)
34. Por causa de você (Tom Jobim e Dolores Duran)
35. Volta (Lupicínio Rodrigues)
36. A violeira (Tom Jobim e Chico Buarque)
37. Qui nem jiló (Luiz Gonzaga e Humberto Teixeira)
38. A morte do vaqueiro (Luiz Gonzaga e Nelson Barbalho)
39. Festa do interior (Abel Silva)
40. Enredo do meu samba (Jorge Aragão)
41. Atire a primeira pedra (Ataulfo Alves e Mário Lago)

42. Coração em desalinho (Mauro Diniz e Ratinho)

43. Chega de saudade (Tom Jobim e Vinícius de Moraes)

44. Naquela mesa (Sérgio Bittencourt)

45. Roda viva (Chico Buarque)

46. Somos todos iguais essa noite (Ivan Lins)