2
Trabalhos Relacionados

Este capítulo faz uma categorização dos algoritmos de ajuste elástico para áudio comprimido existentes na literatura, lista exemplos de ferramentas de ajuste de áudio relevantes e realiza uma comparação entre elas e o algoritmo de ajuste de áudio proposto.

Trabalhos relacionados de algoritmos e ferramentas de ajuste em fluxos audiovisuais e somente visuais são descritos por Cavendish (Cavendish, 2005).

2.1.
Categorização de algoritmos de ajuste para áudio

Alguns trabalhos propõem uma categorização dos algoritmos de ajuste elástico, como em (Arons, 1992; Lee et al., 2004; Bernsee, 2003). As subseções a seguir descrevem classes de algoritmos de ajuste, de acordo com os trabalhos de Arons e Lee (Arons, 1992; Lee et al., 2004), em ordem crescente de qualidade e custo computacional.

2.1.1.
Reprodução rápida/lenta

Os algoritmos dessa categoria modificam a taxa de exibição das amostras da mídia, o que acarreta uma modificação do tempo necessário para a apresentação da mídia. No entanto, tais algoritmos possuem o efeito indesejável de alterar as freqüências componentes do sinal. A referência (Lemay, 1998) apresenta um *applet* que efetua uma série de cálculos relacionados à aplicação dessa categoria de ajuste elástico e à alteração das freqüências componentes do sinal.

A Figura 1 ilustra a modificação do sinal representado no domínio do tempo quando seu tempo de exibição foi duplicado. A metade de cima da figura ilustra o áudio original e a metade inferior mostra o áudio processado. Nesse caso, o áudio perdeu altas freqüências (Nyquist) e se tornou mais grave.

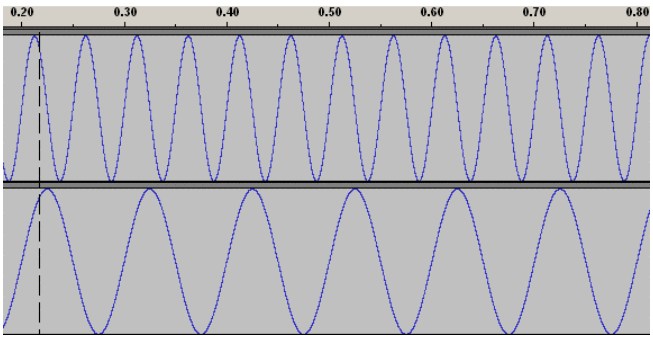

Figura 1 - Mudança da taxa de exibição de um sinal.

Embora essa categoria possa ser considerada um caso particular da categoria no domínio do tempo, descrita a seguir, esse trabalho optou por descrevê-la separadamente devido aos efeitos resultantes típicos nos áudios processados por algoritmos dessa categoria, que é análogo ao áudio resultante da alteração de velocidade de reprodução em fitas e LPs. O áudio resultante é modificado, porém inteligível.

2.1.2. No domínio do tempo

Essa classe de algoritmos processa o áudio no domínio do tempo. O áudio é divido em pequenos quadros e o ajuste é realizado manipulando-os. A idéia de dividir o áudio em pequenos pedaços no domínio do tempo para efetuar algum processamento é bem antiga, datando de 1946 (Gabor, 1946).

È importante ressaltar que as modificações realizadas no domínio do tempo influem em outros domínios do sinal, como freqüência (e vice-versa). A Figura 2 ilustra um exemplo de aplicação de algoritmos no domínio do tempo. Na figura, pequenos quadros do sinal foram removidos, acelerando a reprodução sem perder altas freqüências, mas degradando a qualidade do sinal.

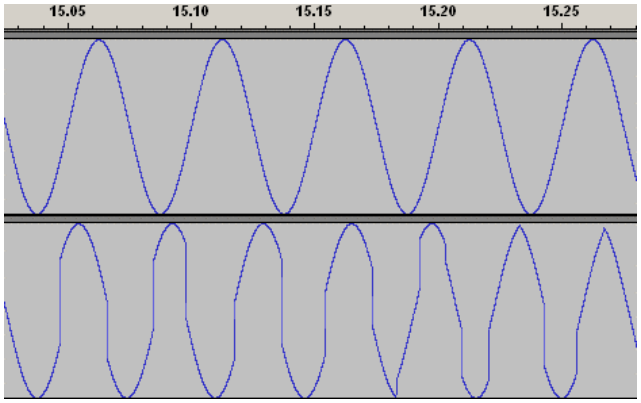

Figura 2 - Exemplo de cortes de quadros de um sinal.

Esses algoritmos possuem baixo custo de processamento. No entanto, a qualidade é limitada a uma pequena variação da taxa. Tipicamente, esses algoritmos atingem boa qualidade com fator de ajuste limitado ao intervalo *0.8<f<1.2* (Lee et al., 2004).

O *Ajuste Regular* é um exemplo de algoritmo dessa classe que descarta ou duplica quadros de áudio a intervalos regulares sem considerar o conteúdo do sinal. Na Figura 3 cada número representa um segmento de áudio. O áudio original possui *6* quadros. É possível reduzir a duração do áudio à metade descartando um segmento sim e outro não, indicado na figura com *3* quadros. Para duplicar o tempo do áudio, é necessário replicar todos os quadros, ilustrado na figura com *12* quadros.

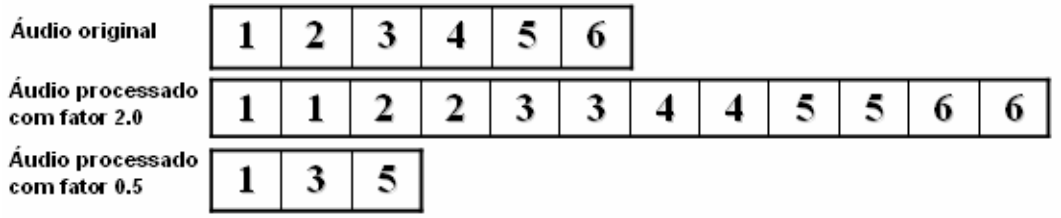

Figura 3 - Algoritmo *Ajuste Regular*.

Quando aplicado à voz humana, o tamanho de tempo de um segmento de áudio deve ser maior do que o tempo necessário para mudar freqüências do sinal de voz (por exemplo, maior do que *15ms*) e menor do que o tempo de pronunciar um fonema de modo a minimizar a degradação da qualidade do áudio (Portnoff, 1981).

Outros algoritmos dessa classe tentam melhorar o desempenho do *Ajuste Regular*. Aron (Arons, 1994) sugere reproduzir os quadros descartados de um canal em outro canal deslocado no tempo. Desse modo, só ocorrem perdas do sinal original quando se utiliza um fator menor do que *0.5*. Na Figura 4 cada número representa um segmento de áudio. O áudio original possui *9* quadros. É possível reduzir a duração do áudio descartando um segmento sim e outro não. Os quadros não descartados são reproduzidos no canal esquerdo e os demais no canal direito. Esse algoritmo aumenta inteligibilidade e compreensão do ouvinte depois de rápida sensação de confusão.

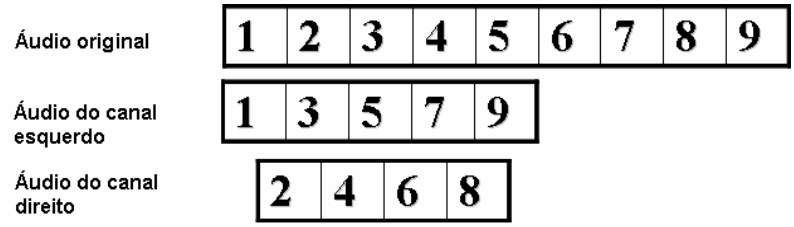

Figura 4 - Ajuste elástico com reprodução dos quadros intercalados.

Outros algoritmos sugerem que o segmento a ser descartado ou duplicado deve ser escolhido de acordo com uma análise (simples) das características do sinal. O algoritmo é aplicável a áudios bem-comportados, como a voz humana. Alguns estudos sugerem que é possível retirar até *50%* do silêncio entre palavras e sentenças sem prejuízos à compreensão (Soares, 2005). No entanto, a proporção de remoção do silêncio da voz é assunto de bastante discussão na literatura (Arons, 1992).

Por fim, alguns algoritmos sugerem que os quadros não sejam simplesmente descartados (ou duplicados), mas sim interpolados. Esse é caso dos conhecidos algoritmos como OLA (*Overlap Add Method*) e SOLA (*Synchronized Overlap Add Method*).

2.1.3. No domínio da freqüência

Os algoritmos dessa classe realizam o ajuste manipulando as freqüências componentes do sinal de áudio. O exemplo mais representativo é o algoritmo *Phase Vocoder (*Hammer, 2001).

A idéia desse algoritmo é similar ao *Ajuste Regular*, no entanto a alteração da duração do sinal é realizada no domínio da freqüência. Seu objetivo é alterar o número de ciclos de freqüências componentes de um sinal, sem mudar quais são as freqüências. A idéia para realizar o ajuste elástico é dividir o sinal original em quadros e realizar o ajuste elástico alterando o tamanho desses quadros. A Figura 5 ilustra um exemplo de ampliação em dois quadros da duração do sinal utilizando o ajuste.

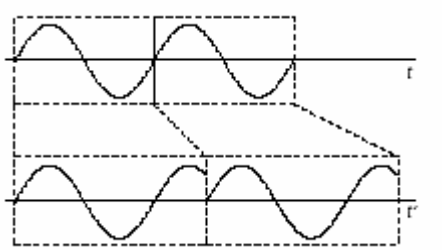

Figura 5 - Ajuste elástico de quadros do sinal no domínio da freqüência.

O problema da alteração descrita é que as extremidades dos quadros alterados não estão alinhadas em relação à fase do sinal, o que introduz altas freqüências. Além disso, quando o sinal é composto por mais de uma freqüência, o algoritmo trata cada freqüência independentemente, o que causa reverberação no sinal processado. Existem alguns trabalhos que aprimoram esse algoritmo (Laroche & Dolson, 1999; Hammer, 2001).

2.1.4. Baseados em análise detalhada

São algoritmos que realizam ajuste elástico após uma análise detalhada do sinal. Tais algoritmos, em geral proprietários, geram áudio de alta qualidade e exigem grande custo computacional, não sendo aplicáveis no tempo de exibição.

O *MPEX* (Prosoniq Mpex, 2004) é um algoritmo que utiliza rede neural treinada para simular algumas propriedades da percepção humana e obter um áudio processado com excelente qualidade. A grande vantagem desse algoritmo é que ele não é baseado estritamente em rígidos modelos matemáticos.

2.2. Ferramentas que realizam ajuste elástico

A maioria das ferramentas comerciais possui algoritmos da categoria do domínio do tempo e da freqüência, uma vez que a reprodução rápida/lenta modifica bastante o áudio original e o custo computacional dos algoritmos baseados em análise detalhada é muito alto. As subseções a seguir detalham os trabalhos relacionados mais representativos para o contexto deste trabalho. Pelo que foi observado, todas as ferramentas citadas realizam o ajuste de modo o mais linear possível e obviamente tentam preservar a fidelidade ao máximo.

2.2.1. Sound Forge

O *Sound Forge* (Sony, 2005) é um *software* para gravação e edição profissional de áudio. A ferramenta oferece suporte a vários formatos de áudio, no entanto, sempre realiza um pré-processamento para abrir arquivos comprimidos.

A funcionalidade de ajuste elástico é acessada através da janela apresentada na Figura 6. Nela, o usuário seleciona o modo e o fator a ser aplicado. A operação de ajuste elástico pode ser experimentada em tempo de exibição, provavelmente utilizando o arquivo pré-processado. No entanto, somente quando o usuário fechar essa janela (clicando em OK) é que o *software* gera o fluxo de dados comprimido, resultante da operação, e essa ação não é realizada em tempo de exibição.

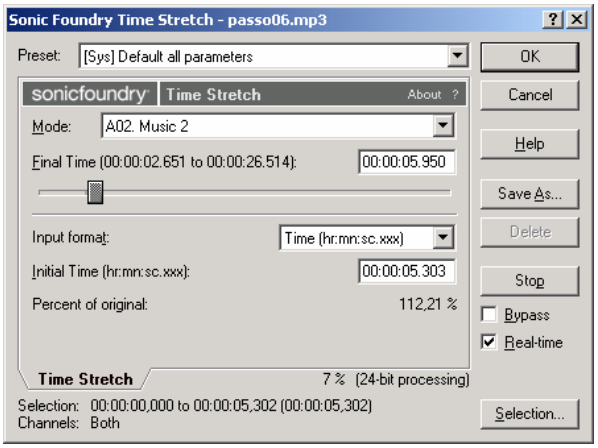

Figura 6 - Interface de aplicação de ajuste elástico em arquivos de áudio no *Sound Forge*.

Existem *19* modos diferentes de aplicar ajuste elástico para o usuário escolher em função de suas necessidades. Alguns modos são adequados para preservar a qualidade da voz, ou do som de algum instrumento (como a bateria), enquanto outros modos tentam preservar a qualidade do áudio como um todo. O fator de ajuste pode variar entre *0.5* e *5*.

2.2.2. Windows Media Player 10

O *Windows Media Player 10* (Microsoft, 2005) é um *software* que permite exibir conteúdo audiovisual dos formatos Windows Media Áudio (WMA), MP3 e ASF. A ferramenta permite selecionar um fator de ajuste a ser utilizado para a reprodução do fluxo de mídia, como ilustrado na Figura 7, e modificá-lo em

tempo de exibição. Embora o fator possa variar entre *0.0625* e *16*, o limite recomendado pela especificação do programa é entre *0.5* e *2.0* para manter alta qualidade.

Ainda que a especificação do programa não indique qual o algoritmo de ajuste elástico utilizado, supomos que essa ferramenta processa o áudio depois da decodificação por dois motivos. O primeiro é que o *Windows Media Player* precisa descomprimir o áudio para exibi-lo e nesse cenário é mais vantajoso aplicar o algoritmo de ajuste no áudio sem compressão (já que é menos custoso) e o segundo motivo é que não é possível salvar o áudio processado em formato comprimido.

Utilizando o *Microsoft Windows Media Player 10 Software Development Kit (SDK)* é possível interagir programaticamente com *Window Media Player*. Dentre as possibilidades, pode-se monitorar marcações de tempo e exibir não somente arquivos, mas também fluxos audiovisuais sendo processados em tempo de exibição.

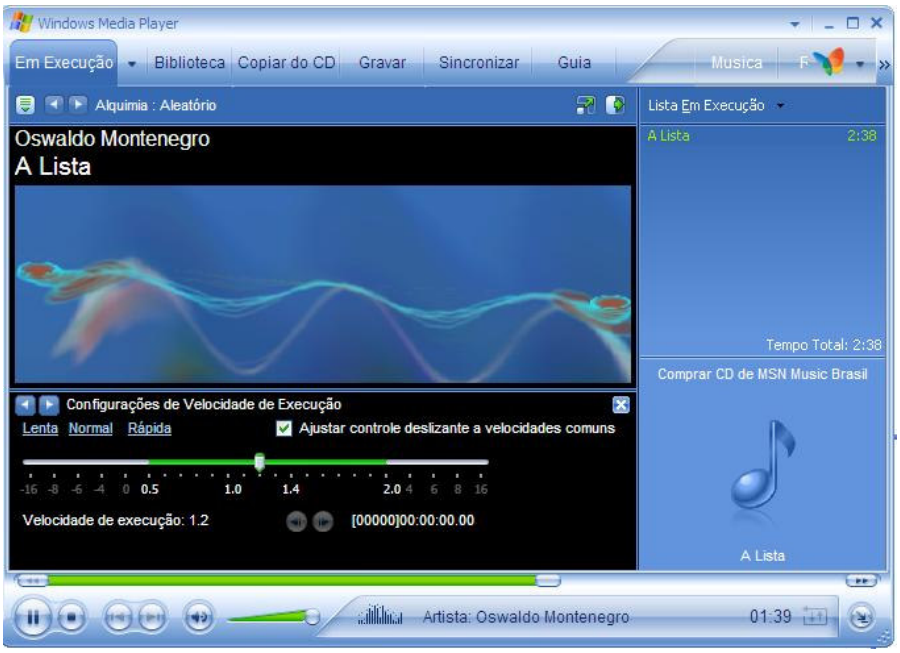

Figura 7 - Interface de aplicação de ajuste elástico em arquivos de áudio no *Window Media Player 10*.

2.2.3. Amazing Slow Downer

O *Amazing Slow Downer* (Roni Music, 2005) é um *software* que permite exibir áudio nos formatos MP3, Wave e Windows Media Áudio (WMA) com uso

de ajuste elástico em tempo de exibição para fatores variando entre *0.5* e *4*, sendo possível alterá-lo durante a apresentação. A Figura 8 mostra a tela principal da ferramenta.

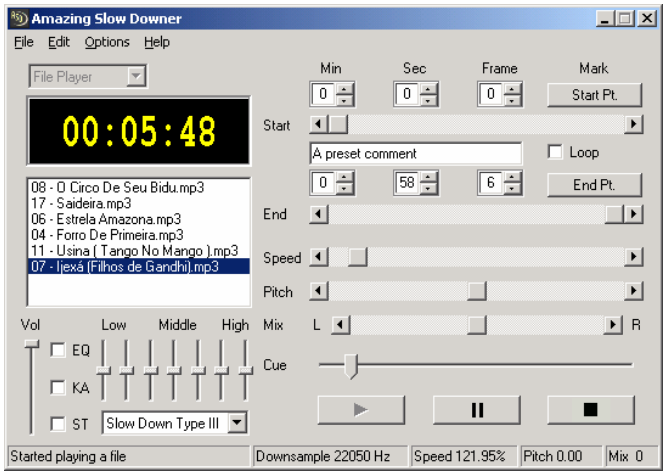

Figura 8 - Interface do *software Amazing Slow Downer*.

A ferramenta disponibiliza três algoritmos diferentes para efetuar ajuste elástico, com crescente uso de recursos computacionais e níveis de qualidade. Pelos mesmos motivos expostos para o *Windows Media Player*, supomos que essa ferramenta aplica o ajuste no áudio sem compressão. Nessa ferramenta é possível salvar os arquivos que foram ajustados, mas somente no formato Wave.

2.2.4. Enounce 2xAV

Enounce 2xAV (Enounce, 2003) é um *plug-in* que adiciona uma barra de controle de taxa de exibição aos programas *RealPlayer*, *RealOne Player* e *Windows Media Player*. Utilizando o *plug-in*, é possível aplicar ajuste elástico em tempo de exibição a fluxos de vídeo e áudio comprimidos com fator *f* dentro dos limites $0.3 \leq f \leq 2.5$ (como ilustra a Figura 9), sendo possível variar o valor do fator aplicado também durante a exibição. Mais uma vez, supomos que essa ferramenta aplica o ajuste no áudio sem compressão pelos motivos já expostos para o *Windows Media Player*.

Figura 9 - Interface do *Enounce* 2xAV.

2.2.5.
585 Time Scaling Processor

O *hardware 585 Time Scaling Processor* (Dolby, 2005b) é um exibidor de mídia que permite realizar ajuste elástico em áudio sem compressão com até oito canais com fatores de ajuste variável entre *0.85* e *1.15*.

Os algoritmos de ajuste elástico utilizados são proprietário da empresa *Dolby* (Dolby, 2005a). Se o requisito de tempo de exibição não for necessário, o ajuste elástico gera áudios com excelente qualidade (distorção máxima de *0.01%* quando o áudio de entrada possui freqüências de até *1kHz* e de *0.02%* quando possui freqüências de *20Hz* até *20kHz*). Este algoritmo é ideal para pequenos programas (cerca de *23* minutos para áudio com um canal e de apenas *3* minutos para áudio com oito canais), já que o *hardware* pode armazená-lo para posterior exibição. No entanto, se for necessário aplicar o ajuste elástico em tempo de exibição, o processador aplicará um algoritmo da categoria 2.1.1, ocasionando mudança proporcional nas freqüências do sinal. A Figura 10 ilustra a interface do processador.

Figura 10 - Interface de controle do *585 Time Scaling Processor*.

2.2.6.
DIRAC

O *DIRAC* (Bernsee, 2005) é uma biblioteca que permite realizar ajuste elástico em áudio não comprimido em tempo de exibição com possibilidade de mudança do fator, que pode variar entre *0.5* e *2.0*. A biblioteca é escrita em C/C++ e é distribuída em três versões, sendo uma delas gratuita.

O *DIRAC* manipula apenas sinais sem compressão, sendo assim, se for aplicar ajuste a sinais comprimidos, é necessário primeiro decodificá-lo e, se necessário, depois codificá-lo novamente.

2.2.7. FastMPEG

O *FastMPEG* (Covell et al., 2001) propõe três algoritmos de ajuste elástico para áudio MP2, um deles da categoria reprodução rápida/lenta e dois do domínio do tempo. Todos os algoritmos são aplicados após uma decodificação parcial do fluxo de dados, seguidos, então, por uma codificação parcial. Segundo os autores, os algoritmos funcionam em tempo de exibição, mesmo com o custo do pré- e pós-processamento. O fator de ajuste é variável entre o intervalo de *2/3* a *2.0*.

Existem quatro passos no funcionamento do *FastMPEG*, ilustrados na Figura 11. No primeiro passo, o fluxo MPEG é analisado, os fatores de escala são removidos e os *30* MDSSs[3] são separados. No segundo, algum dos três algoritmos de ajuste elástico é aplicado em paralelo em todos os *30* MDSSs. O modelo de mascaramento psicoacústico é inferido e modificado no terceiro passo. Por último, o fator de escala é novamente aplicado, os novos sinais são quantizados com o novo modelo de mascaramento psicoacústico e os bits são organizados para compor um novo fluxo MPEG.

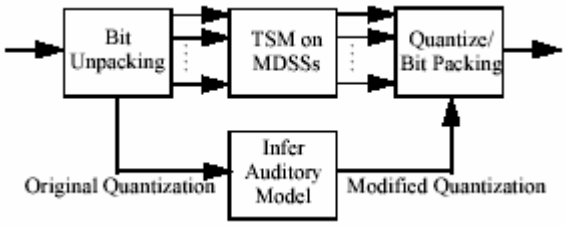

Figura 11 - Estrutura de funcionamento do *FastMPEG*.

A qualidade dos áudios modificados pelo *FastMPEG* é comprometida principalmente pelo algoritmo de ajuste, por pré-supostos da codificação que não são mais verdadeiros quando o sinal é modificado e pelo modelo de mascaramento psicoacústico recuperado. Não existe uma ferramenta disponível com a implementação dos algoritmos, mas alguns resultados de mídias ajustadas utilizando tais algoritmos são apresentados por Covell (Covell et al., 2001).

[3] O codificador MPEG aplica uma ou mais transformadas nas amostras, dividindo-as em sub-bandas. Uma sub-banda contém um conjunto de amostras correspondente a uma faixa do espectro de freqüências audível. Cada conjunto é conhecido como MDSS (*maximally decimated subband streams*).

2.2.8. PICOLA do MPEG-4

O padrão de áudio do MPEG-4 (ISO, 2001b) codifica sinais de voz humana e áudio multicanal com alta qualidade e também oferece suporte a áudios naturais e sintéticos. Por tratar sinais tão distintos, o MPEG-4 áudio possui diferentes mecanismos de codificação de acordo com as características do áudio e da taxa de bits a ser atingida.

Um decodificador MPEG-4 de áudio, como o software de referência MPEG (Moving Picture Experts Group, 2001), deve saber manipular todos os diferentes formatos de áudio e poder aplicar ferramentas de efeito ou mixagem no áudio decodificado antes de o enviar para exibição. A Figura 12 ilustra esse tipo de decodificador.

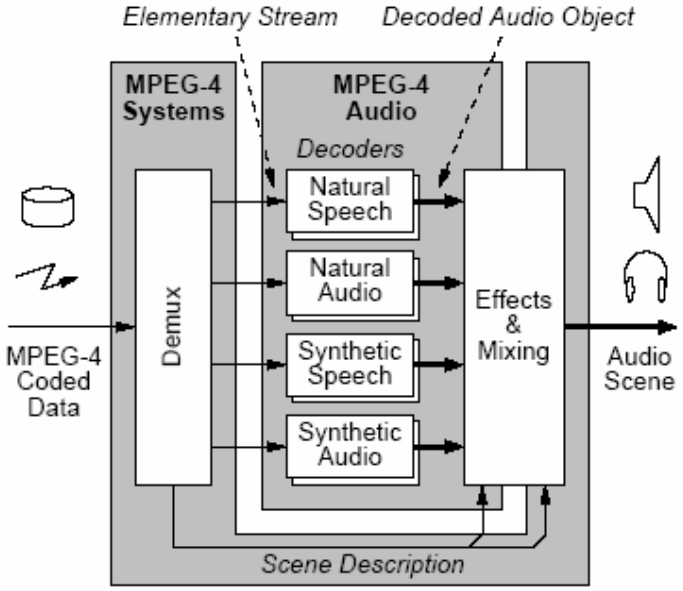

Figura 12 - Decodificador MPEG-4 - fonte: (Wolters & Kjörling, 2003).

Um exemplo de ferramenta de efeito é a chamada *PICOLA* (*Pointer Interval Controlled OverLap Add*), que permite realizar ajuste elástico utilizando algoritmos do domínio do tempo em sinais de áudio MPEG-4 monofônicos com taxa de amostragem de *8kHz* ou *16kHz*. O ajuste pode ser realizado em tempo de exibição com fator variável entre *0.5* e *2.0*.

2.2.9.
Padrão sines + transients + noise

Levine (Levine & Smith, 1998) propõe um novo padrão de áudio comprimido que facilita a realização de processamento no domínio comprimido. O padrão proposto é capaz de produzir áudio com taxas altas de compressão (*16-48kbps*) com qualidade similar a do MPEG-2 AAC, quando se compara utilizando uma taxa de *32kbit/canal* em ambos os padrões.

O padrão especifica que o áudio deve ser dividido em três sinais independentes: *sine*, *transient* e *noise*. O sinal *transient* contém os sons de ataques do áudio original. O sinal *sines* é uma soma de freqüências da região de *0* a *5kHz*. O sinal *noise* modela altas freqüências que não estão no *transient*. Essa separação permite que cada uma das três partes possa ser manipulada de modo diferenciado e eficientemente codificada.

É possível aplicar ajuste elástico de boa qualidade no fluxo modificando apenas os sinais *sine* e *noise*. O sinal *transient* é apenas deslocado para acompanhar a mudança dos demais sinais, mantendo seu envelope temporal. Desse modo é possível realizar ajuste elástico conservando os ataques do áudio original. Numa música, por exemplo, é possível realizar ajuste elástico em instrumentos harmônicos e vozes e ainda manter os ataques de instrumentos de percussão. Alguns exemplos de áudios processados nesse formato são apresentados por Levine (Levine, 1998).

2.3.
Comparação dos trabalhos relacionados com o proposto

A comparação dos trabalhos citados com o proposto por este trabalho deve ser realizada considerando os requisitos das aplicações descritas na Seção 1.1: tipo de sinal processado, tempo de processamento, possibilidade de variação do fator em tempo de exibição, amplitude do fator de ajuste, fidelidade, linearidade, independência do exibidor de conteúdo, suporte a dados “ao vivo” e a monitoramento de âncoras.

A primeira análise comparativa diz respeito ao tipo de sinal a ser processado em tempo de exibição. Os trabalhos *Windows Media Player 10*, *Amazing Slow Downer*, *Enounce 2xAV*, *FastMPEG* e *PICOLA* podem manipular sinais de áudio comprimidos em tempo de exibição. O *DIRAC* e *585 Time Scaling Processor* não

oferecem essa funcionalidade e o *Sound Forge* precisa primeiramente pré-processar o áudio para descomprimi-lo (ver Subseção 2.2.1). Vale ainda ressaltar que o *PICOLA* do MPEG-4 só oferece suporte a áudios monofônicos com determinadas taxas de amostragem.

Todos os algoritmos apresentados permitem modificar o fator de ajuste em tempo de exibição (exceto o *Sound Forge*), possuem uma amplitude do fator de escala maior ou igual a *10%*, tentam preservar a fidelidade ao máximo e realizam o ajuste de modo o mais linear possível, assim como o algoritmo proposto.

Em relação à independência do exibidor de conteúdo, uma análise mais detalhada é necessária. Embora não seja claro como os algoritmos de *Sound Forge*, *Windows Media Player*, *Amazing Slow Downer* e *Enounce 2xAV* funcionam, supomos que essas ferramentas processam o áudio depois da decodificação. Sendo assim, o algoritmo de ajuste dessas ferramentas é dependente do exibidor de conteúdo (que, muitas vezes, é a própria ferramenta). A especificação do *PICOLA* do MPEG-4 define que novos decodificadores devem prover ajuste elástico e funciona para esses decodificadores. O *DIRAC* e *585 Time Scaling Processor* não manipulam áudio comprimido, por isso, fica fácil para tais ferramentas serem independentes do exibidor de conteúdo. O *FastMPEG* atua diretamente no domínio comprimido, gerando um novo fluxo de áudio comprimido. No entanto, o algoritmo que o *FastMPEG*[4] utiliza é fortemente dependente do modo como arquivos MPEG BC são codificados, sendo bastante complicado generalizar um algoritmo desse tipo para outros formatos de áudio.

Em síntese, todos os algoritmos de ajuste elástico citados não manipulam o fluxo de áudio comprimido diretamente, adotando a opção de decodificar (ainda que parcialmente), processar e, se necessário, codificar novamente o fluxo de áudio. Essas soluções são custosas e dependentes da decodificação (e muitas vezes do exibidor de conteúdo). O mecanismo de ajuste proposto por este trabalho opera diretamente no domínio comprimido, sendo independente do processo de decodificação.

Vale ainda destacar que embora seja interessante definir um novo padrão de áudio, como o mencionado na Subseção 2.2.9, que facilite a realização de

[4] A referência do *FastMPEG* é um artigo de resumo, mas nenhuma aplicação executável parece estar disponível na Internet.

transformações como o ajuste elástico, é difícil acreditar que este formato conquistará o mercado de áudio rapidamente, principalmente sem o apoio de um órgão conhecido de padronização e grandes empresas. Sendo assim, este trabalho opta por manipular formatos de áudio padronizados e maciçamente utilizados.

Por fim, é interessante comparar as características da ferramenta de ajuste proposta com as demais. Apenas o *Windows Media Player* (via SDK), o *DIRAC*, o *FastMPEG* e o algoritmo proposto foram desenvolvidos com objetivo de facilitar integração via programação com outras aplicações. Duas características de uma ferramenta de ajuste merecem destaque: suporte a dados "ao vivo" e monitoramento de marcações de tempo. O *Windows Media Player* (via SDK) e o trabalho proposto oferecem suporte a essas características. O DIRAC, como mencionado, não funciona para dados comprimidos e a especificação do *FastMPEG* não deixa claro se a ferramenta possui tais características.

Em resumo, percebe-se que nenhum dos trabalhos relacionados encontrados atende satisfatoriamente a todos os requisitos propostos. Em todo material pesquisado na literatura nenhuma ferramenta foi encontrada com propósito e solução similares aos apresentados por este trabalho.