UM SISTEMA PARA SÍNTESE DE VOZ E ANIMAÇÃO DE FACES VIRTUAIS VOLTADO PARA DISPOSITIVOS MÓVEIS

Manoel FIUZA Lima Junior(1); Michael Robson MONTEIRO(2); Carlos Maurício J. de M. DOURADO Jr(3); José Marques SOARES(4); Giovanni BARROSO(5)

(1) Cefet-Ce, Rua 04, 654 – Jereissati I, Maracanaú – CE, CEP: 61900-350, email: mfiuzajunior@yahoo.com.br (2) FIC, email: mkaelzinho@gmail.com (3) UFC, email: cmauriciojd@gmail.com (4) CEFET-CE, email: marques@cefetce.br (5) UFC, email: gcb@fisica.ufc.br

RESUMO

Este trabalho apresenta um sistema que sintetiza voz a partir de textos fornecidos como entrada e gera animações, sincronizando o áudio produzido com um vídeo que simula movimentos de lábios e face. O áudio é gerado por um Módulo *Text-to-Speech* (TTS) e o vídeo através da animação que pode ser gerada pela deformação de um modelo tridimensional da face humana ou por técnicas baseada em seqüência de imagens (*keyframing*). O sistema é formado por partes interdependentes. O primeiro bloco, chamado *front-end*, recebe como entrada o texto a ser sintetizado e executa um pré-processamento para gerar os dados repassados ao segundo módulo, chamado *back-end*. Este, por sua vez, apresenta como saída o áudio sintetizado correspondente ao texto de entrada. O áudio sintetizado é, em seguida, processado para extração das suas características, servindo como parâmetros de entrada para uma rede neural que identifica as vogais pronunciadas e a duração de cada fonema. Identificadas as vogais, o sistema busca em um Banco de *Visemas* (dados que associam um fonema a uma representação gráfica da face virtual) a identificação da imagem correspondente ou os parâmetros de deformação do modelo tridimensional para cada vogal identificada, gerando a animação e executando-a de maneira sincronizada ao sinal de áudio. Um protótipo do sistema encontra-se instalado em dois *smartphones* (HTC Touch e Treo 750) que usam sistema operacional Windows Mobile 6.0.

Palavras-chave: Text-to-speech, sintetizadores de áudio, Visemas

1. INTRODUÇÃO

O desenvolvimento de um sistema para síntese de voz e animação facial voltado para dispositivos móveis, principalmente celulares e *smartphones*, se faz oportuno devido ao aumento da popularidade e uso crescente de tais equipamentos. Além disso, seguindo a grande aceitação do serviço de mensagens curtas (*Short Message Service* – SMS), animar um modelo virtual que acompanha o áudio sintetizado a partir de texto representa um enriquecimento da interatividade do usuário com seu dispositivo e com outros usuários distantes. Esse tipo de solução pode ainda ser estendido, não se restringindo apenas à leitura de SMS, para aplicações que auxiliem usuários portadores de deficiência auditiva no uso do conjunto das facilidades que os serviços dos dispositivos móveis fornecem.

Neste trabalho é apresentado o modelo usado no desenvolvimento de um sistema de síntese de voz e animação facial voltado para dispositivos móveis. Tal modelo é composto por quatro módulos principais, que podem ser vistos na Figura 1: Módulo *Text-to-Speech* (MTTS), Módulo de Extração de Fonemas (MEF), Módulo de Interpretação de Visemas (MIV) e Módulo de Animação (MA).

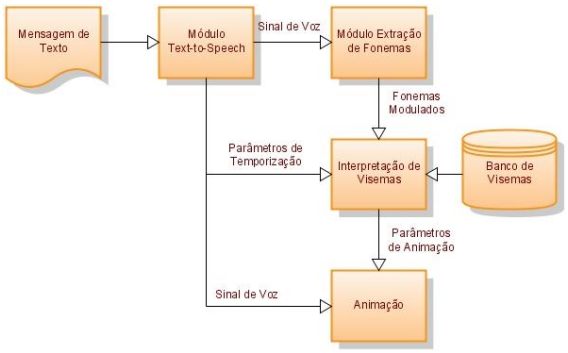

Figura 1 – Esquemático completo do sistema desenvolvido.

A entrada do sistema é uma mensagem de texto que passa inicialmente pelo MTTS que, após processamento, gera um arquivo de áudio contendo a voz sintetizada. Em seguida, o arquivo de áudio é repassado para o MEF e para o MA. O primeiro vai analisar o arquivo para identificação das vogais, com o auxílio de uma rede neural que identifica as vogais pronunciadas e o tempo de duração de cada uma, enquanto o segundo irá sincronizar o áudio com a animação final. O MEF, depois de identificar fonemas significativos no áudio, irá fornecer ao MIV os Fonemas Modulados. Apenas as vogais são identificadas. De posse desses dados, o MIV recupera em um Banco de Visemas as informações dos visemas correspondentes aos Fonemas Modulados, fornecendo os Parâmetros de Animação para o MA, que sincroniza o áudio gerado pelo MTTS à animação da face, a qual pode ser produzida a partir de imagens ou pela deformação de um modelo tridimensional da face.

Cada módulo contém partes interdependentes que desempenham tarefas bem definidas que vão desde a leitura do texto SMS à reprodução da animação usando um *avatar*. Além disso, essas partes trabalham com entrada de dados e fornecem saída de dados para as etapas subseqüentes.

O primeiro deles é o Módulo TTS, composto por um chamado *Front-end* e um *Back-end*. O *Front-end* é caracterizado pelo trabalho em conjunto de três blocos: Analisador Textual, Analisador Fonético e Analisador Prosódico. Ao *Back-end* é atribuída a atividade de renderizar o sinal de áudio, caracterizando a síntese de voz, o qual é remetido e em seguida processado para extração das suas características.

Seguindo o caminho de processamento do texto de uma mensagem para síntese de um sinal de áudio, tem-se o Módulo de Extração de Fonemas. Este módulo é responsável por identificar os fonemas contidos na stream de áudio gerada pelo Módulo *Text-to-Speech*, e repassar esses fonemas para o Módulo de Interpretação de

Visemas na forma de Fonemas Modulados. Um fonema é a menor unidade sonora de uma língua, e um visema é a representação visual do rosto humano ao pronunciar um fonema. Um mesmo visema pode representar vários fonemas diferentes e existem varias classificações para a correspondência visema/fonema [Salgado, 2001].

O Módulo de Interpretação de Visemas se encarrega de usar os Fonemas Modulados, fornecidos pelo Módulo de Extração de Fonemas, para serem associados a visemas correspondentes, carregados de um Banco de Visema, e enviar os Parâmetros de Animação ao Módulo de Animação para que então seja construída a animação final. Esta aplicação se apresenta como uma boa alternativa à transmissão de vídeo, pois requer menos largura de banda e, portanto, é economicamente mais viável.

2. MÓDULO TEXT-TO-SPEECH

O primeiro módulo a ser trabalhado foi o de TTS, responsável por transformar a mensagem textual do remetente em mensagem falada.

Este tipo de implementação é comum nos dias atuais e encontra-se empregada em algumas aplicações voltadas ao auxílio de deficientes visuais para o uso de computadores. Por exemplo, pode-se citar o Virtual Vision [Virtual Vision, 2008], atualmente na versão 5.0. Consiste em um leitor de tela para ambiente Microsoft. Lê telas do Windows, Office e Internet Explorer. Outra aplicação de grande uso é o Jaws [Jaws, 2008] outro leitor de tela para ambiente Windows.

Existem algumas implementações prontas e disponíveis na internet. Como neste trabalho as aplicações serão desenvolvidas prioritariamente em java, apenas as soluções disponíveis nesta linguagem foram consideradas.

Uma biblioteca gratuita e de código aberto chamada de freeTTS [freetts, 2008] foi modificada para o propósito deste trabalho, na etapa de testes iniciais deste projeto, os quais foram realizados inicialmente em uma plataforma *desktop*. Esta biblioteca permite ajustes nos parâmetros básicos, como intensidade, velocidade e freqüência, o que irá fornecer recursos para uma melhor expressão das emoções. Posteriormente, testes em um smartphone foram realizados fazendo uso de um SDK proprietário, SmartRead SDK [SmartRead]. É um conjunto de bibliotecas escritas em C++/C#, usado para implementar o Módulo TTS. Este se caracteriza pelas partes integrantes que serão descritas nas subseções seguintes.

2.1 Front-end

O *Front-end* é a parte do Módulo TTS que se encontra diretamente relacionada com o texto de entrada. É aqui onde o Módulo TTS recebe e começa a analisar o texto. Esse, por sua vez, poderá conter várias *tags* de formatação, fornecendo informações relevantes ao *Front-end* [Dorf, 2006]. Essas informações podem controlar características do texto como entonação ou emoções. O texto passa por vários blocos do *Front-end*, onde, em cada bloco, uma série de análises é feita para processar o texto de entrada.

O primeiro deles é o analisador textual, onde a estrutura de todo o texto de entrada é reconhecida. Neste bloco, estruturas como quebra de parágrafos e separação entre as várias sentenças são detectadas e marcadas. Dentro desse bloco é feita também a normatização do texto. Este processo caracteriza-se por identificar estruturas específicas como datas, moeda, siglas e abreviações, transformando-as em uma representação textual. Finalmente, a análise lingüística do analisador textual trabalha sobre o texto previamente processado pelas etapas anteriores, visando identificar estruturas lingüísticas no texto.

O próximo bloco é o analisador fonático, onde o texto de entrada é recebido com as marcações inseridas pelo Analisador Textual. O Analisador Fonético utiliza essas marcações para identificar fonemas no texto, realizando a transcrição de grafemas, representação comum da fala, baseada no alfabeto, para fonemas, uma representação gráfica dos sons individuais da fala, baseada no International Phonetic Alphabet [IPA, 2008].

O resultado é o texto com *tags* adicionais que delimitam os fonemas dentro do texto.

Por último tem-se o analisador prosódico, onde o texto com os fonemas identificados entram no Analisador Prosódico para que possam ser identificadas informações de freqüência e duração para cada fonema encontrado.

2.2 *Back-end*

O *Back-end* é a parte do Módulo TTS que fica próxima à saída. É onde será gerado o arquivo de áudio contendo a representação sintetizada do texto de entrada. A maneira mais simples de se alcançar esse objetivo é concatenar unidades fonéticas gravadas em um banco de dados, correspondendo à estrutura fonética do texto marcado [Dorf, 2006].

3. MÓDULO DE EXTRAÇÃO DE FONEMAS

Este módulo usa o método baseado na Predição Linear [Kshirsagar, 2005] para identificar os fonemas vocálicos no arquivo de áudio emitido pelo Módulo TTS. Esse método será descrito nas subseções que se seguem.

3.1 Método Baseado na Predição Linear

O método se constitui na identificação dos fonemas vocálicos existentes em um sinal de áudio e posterior modulação destes a partir da energia com que são, respectivamente, pronunciados, definindo, assim, uma maior ou menor abertura da boca. Como a base da linguagem é feita sobre as vogais, é possível obter uma animação correspondente a pronuncia do que foi dito desprezando as consoantes e utilizando as apenas as vogais [Kshirsagar, 2005] [Thalmann, 2004].

O método funciona bem para vogais, pois a Predição Linear leva em conta que as vogais são produzidas pelas cordas vocais, enquanto as consoantes são formadas através de modificações no som pelo trato bucal [Morgadinho, 2008]. Além disso, a Predição Linear tem a vantagem de necessitar de menor poder de processamento por causa da simplificação do problema, restringindo-o apenas à identificação de vogais, sendo ideal para o uso em dispositivos portáteis.

Cada vogal, em um período pequeno de tempo, apresenta uma forma de sinal diferente das outras, fazendo assim o reconhecimento de vogais um exercício de identificação de padrões [Kshirsagar, 2005] [Thalmann, 2004]. Os Coeficientes de Predição Linear são usados para se ter uma representação compacta deste padrão, caracterizando todos os instantes como uma determinada vogal em um sinal de voz. Estes coeficientes são, então, servidos a uma Rede Neural, que tem a função de identificar estes padrões, para determinar a vogal, ou grupo de vogais, correspondente. Também é calculada a energia com que o fonema foi emitido, para que possa ser especificada a abertura da boca da face virtual.

O método baseado na Predição Linear é definido consiste no pré-processamento do arquivo de áudio com a posterior extração de uma série de informações desse arquivo [Morgadinho, 2008].

No pré-processamento o sinal de voz é tratado para a extração de suas características, sendo feita, inicialmente, a pré-ênfase, que tende a diminuir distorções provocadas pelos lábios no sinal de voz. Em seguida, o sinal é dividido em pequenas partes, chamadas frames, com duração de dez a vinte milissegundos, e é realizado o Janelamento Hamming [Parks, 1987][Proakis, 2000], técnica utilizada para aumentar as características espectrais do sinal, o que facilita o reconhecimento dos

fonemas.

Depois de processado, para cada frame, é calculada a energia média, que determina a intensidade da vogal e os coeficientes da predição linear, que podem ser em torno de dez ou doze, além do *cross rate* que indica a pronúncia de fricativos (fonemas que, apesar de possuírem uma energia muito baixa não indicam o fechamento da boca).

3.2 Rede Neural

A Rede Neural recebe os coeficientes da predição linear de cada *frame* e tenta associá-los a um padrão. O uso de uma rede com mecanismo de aprendizado por retropropagação (*backpropagation*) e três camadas, sendo dez nós na primeira para recebimento dos coeficientes, vinte e cinco nós internos e cinco na última camada representando os padrões de vogais, mostrou-se eficiente para a o reconhecimento das vogais básicas [Byorick, 2003]. Como essa rede apresenta uma topologia simples e realiza apenas cálculos lineares esperava-se que exigisse o mínino de recursos de *hardware* para a identificação dos fonemas, o que ficou comprovado nos testes realizados.

3.3 Observações Sobre o Modelo de Predição Linear

Alguns passos merecem atenção especial no desenvolvimento do modelo, sendo o primeiro deles é o agrupamento dos fonemas vocálicos em visemas, visto que pode-se encontrar na bibliografia a existência de vinte fonemas vocálicos no inglês [São Francisco, 2008]. Estes devem ser agrupados pelas características, inclusive visuais, de forma a facilitar o reconhecimento pela rede neural.

O mais importante dos passos é o desenvolvimento e treino da rede neural. O sucesso do método depende de como a rede for treinada, uma vez que a identificação das vogais é fundamental para um resultado ótimo, sendo necessário um grande grupo de exemplos que suportem a maioria dos possíveis casos.

Outro passo importante é a identificação da energia máxima e mínima de uma vogal, pois é a partir dessas que se normalizará a maior abertura da boca do modelo, assim como uma energia mínima correspondente ao silêncio.

4. MÓDULO DE INTERPRETAÇÃO DE VISEMAS

Este bloco é responsável por fazer a associação entre as vogais reconhecidas e seus visemas correspondentes no banco de visemas, levando em consideração os parâmetros emocionais utilizados ou identificados no momento em que o fonema é dito e a energia com que a vogal é pronunciada. Com estes dados devem ser construídos os parâmetros de animação.

5. MÓDULO DE ANIMAÇÃO

5.1 Visão Geral

O módulo de animação é o responsável pela construção da animação, sintetizando o vídeo (ou usando imagens em técnicas de *keyframing*) e aplicando os parâmetros ao modelo da face associados ao áudio. Este processo consiste em algumas etapas que serão descritas a seguir.

A primeira delas é a definição do modelo facial a ser animado e consiste desenvolver um modelo 3D em alguma ferramenta específica ou utilizar algum disponível na Internet.

Uma vez escolhido o modelo, parte-se para a obtenção do conjunto de informações, denominadas Parâmetros Pré-calculados, que serão usadas pelo Algoritmo de Deformação para que este possa realizar a animação. Vale enfatizar que os parâmetros são ditos pré-calculados pelo fato de serem definidos antes da execução em tempo real do animador, utilizando-se, para sua obtenção, uma ferramenta apropriada, desenvolvida no decorrer das pesquisas para esta finalidade. A captura de tela desta ferramenta pode ser vista na Figura 2).

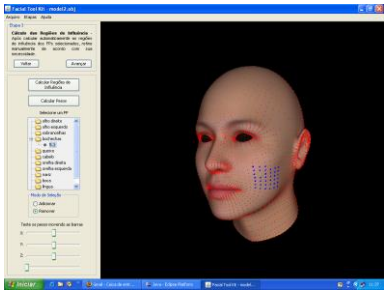

Figura 2 – Ferramenta utilizada para definição dos parâmetros pré-calculados.

A etapa seguinte consiste na busca no banco de dados contém as definições dos Facial Animation Parameters (FAPs) especificados pelo padrão MPEG-4. Esses FAPs contêm as informações relativas à animação. Enquanto os parâmetros pré-calculados constituem informações especificas do modelo (por exemplo, a localização do canto direito da boca), um FAP pode conter simplesmente a instrução “abra a boca”, que é independente do modelo usado.

Por fim, tem-se o Algoritmo de Deformação, (Figura 3), que é o responsável pela integração de todos os elementos presentes no processo. Ao receber um FAP, o algoritmo verifica qual a configuração desejada, através de uma consulta ao Descritor de FAPs, localiza a região do Modelo Facial a ser modificada, analisando seus Parâmetros Pré-calculados e envia instruções ao Motor Gráfico para que este redesenhe o modelo com a nova expressão. Na Figura 3 é apresentado o diagrama de blocos deste módulo.

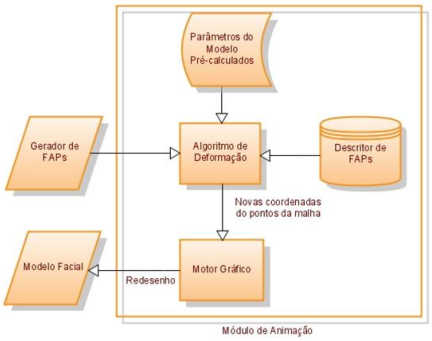

Figura 3 – Diagrama de blocos do módulo de animação.

É importante ressaltar que o Gerador de FAPs indicado na Figura 3 é composto por todos os módulos de análise fonética previamente descritos.

5.2 Modelo Facial

O primeiro passo do processo de animação facial consiste na especificação do modelo tridimensional utilizado. Este, a princípio, deverá ser composto por três malhas, a primeira contendo a geometria da face propriamente dita com abertura dos olhos e da boca, outra contendo a geometria do olho direito e uma última contendo a geometria do olho esquerdo.

As três malhas deverão estar no mesmo nível de hierarquia, independentes e sem agrupamentos.

Para uso em dispositivos móveis, foi utilizado o formato M3G para o modelo facial completo [Höfele, 2004]. Neste formato, é possível definir um identificador para cada nó do grafo da cena, possibilitando que o algoritmo de animação possa localizar cada malha e realizar as deformações correspondentes. Neste contexto, cada uma das três malhas deve conter um identificador próprio (M3G UserID), definidos da seguinte forma: face = 1, olho direito = 2, olho esquerdo = 3.

O modelo pode possuir qualquer quantidade de pontos (dentro dos limites estabelecidos pelo padrão M3G que é de 65536 vértices). Entretanto, quanto maior a resolução (numero de vértices), menor será o desempenho do algoritmo de deformação. Para aplicações em dispositivos móveis, modelos com baixa resolução apresentam resultados satisfatórios. O modelo deve apresentar um conjunto mínimo de pontos que o torne apto a ser parametrizado segundo normas do padrão MPEG-4.

5.3 Parâmetros Pré-calculados

A parametrização do modelo consiste em definir para o mesmo um conjunto de informações usadas para guiar o processo de deformação da malha, feita na etapa seguinte. O algoritmo de deformação tem caráter genérico, ou seja, deve ser capaz de animar qualquer tipo de modelo, independente de forma, tamanho, ou qualquer outra particularidade. Sendo assim, o animador precisa "conhecer" o modelo 3D para que possa manipulá-lo (deformá-lo) adequadamente. Como exemplo, para que a boca do modelo facial seja movimentada pelo animador, é necessário que este consiga localizá-la na malha.

O diagrama apresentado na Figura 4 resume as etapas do processo de parametrização do modelo facial.

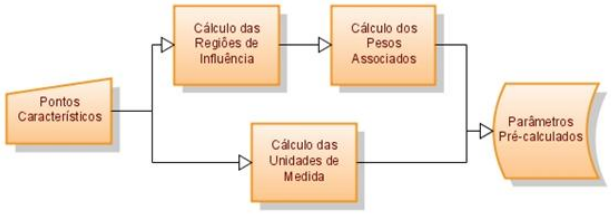

Figura 4 – Diagrama de blocos do processo de parametrização do modelo facial.

Vale explicitar que pontos característicos (Feature Points – FP) são pontos que caracterizam a face que, representando os pontos-chave na deformação da malha facial e as regiões de influência dos FPs consistem na área de atuação do FP, ou seja, a movimentação de um dado FP deforma a malha movendo os vértices vizinhos que estão contidos nessa área.

É ainda importante saber que o peso é o fator de intensidade da influência de um ponto

característico em um vértice. É o que garante que a deformação seja mais intensa próximo ao FP e mais suave em regiões afastadas.

6. TESTES

Os testes iniciais deste projeto foram realizados inicialmente em uma plataforma desktop, utilizando como equipamento um computador dotado de processador Pentium 4 3.0 Ghz e 1GB de memória. Posteriormente foram realizados testes em um dispositivo móvel do tipo *smartphone* da empresa HTC modelo Touch e processador de 200Mhz.

No ponto da realização dos testes, a plataforma de TTS de geração do áudio não foi ainda integrada aos demais módulos do sistema. Dessa forma, o TTS gera um arquivo de áudio no formato WAV do tipo mono com amostras de 8 bits e 11Khz de freqüência.

Foram utilizados dois tipos de fontes geradoras desse arquivo, sendo a primeira o programa do TTS e a segunda um programa de gravação de áudio normal, tendo sido configurados os parâmetros de gravação já citados.

6.1 Realização dos Testes no PC

O primeiro passo para o início dos testes foi a definição da rede neural a ser utilizada. Com base no artigo de Jeff Byorick, Ravi P. Ramachandran e Robi Polikar [Byorick, 2003] optou-se por uma rede neural do tipo backpropagation. O artigo dava margem para a utilização de 10 ou 12 neurônios na primeira camada, além da utilização de 25 neurônios na camada intermediária e 5 na camada de saída.

Os primeiros testes com a rede neural foram feitos através de um framework específico para modelagem de redes neurais, o JoonEdit [Joone, 2008]. Através dos testes realizados pôde-se verificar que a utilização de 12 neurônios na primeira camada implicava em um peso de processamento consideravelmente maior que o uso de apenas 10. Como o objetivo era poder portar o aplicativo também para dispositivos móveis, optou-se pela utilização de 10 neurônios.

O treinamento da rede neural foi realizado apenas com o arquivo gerado pelo Módulo TTS, pois o objetivo final do trabalho é a utilização apenas dessa saída como fonte de áudio. Foram utilizadas duas vozes distintas fornecidas pelo programa e 5 frases diferentes

Já treinada a rede, colocou-se o sistema para carregar o arquivo de áudio em fatias de 20ms (segundo [Byorick, 2003], esse é o tempo ideal para a correta codificação para este tipo de áudio) e para cada amostra dessas foram gerados os coeficientes do LPC, sendo estes processados pela rede neural.

Os resultados dos testes foram satisfatórios quando se utilizava um dos textos utilizados na fase de treinamento da rede, porém, ao se utilizar uma frase diferente, pôde-se perceber visualmente uma diferença entre o fonema falado no momento e a face do avatar. Quando o arquivo utilizado era o de uma pessoa através do gravador de som do computador, a diferença se tornava ainda maior.

Após alguns testes com modificações no número de neurônios na camada intermediária da rede, foi verificado um melhor resultado ao se utilizar 30 neurônios nessa camada e não 25 como proposto em [Byorick, 2003]. Após os testes utilizando o framework de JoonEdit, a rede neural foi

implementada na linguagem Java de maneira a poder ser executada tanto no PC como no dispositivo móvel.

Outra modificação realizada, em que se pôde notar um bom ganho na qualidade do resultado, foi que a análise de 20 ms de amostragem de áudio foi aplicada em intervalos de 80 ms o que tornava a animação menos fragmentada e mais fluida, apresentando resultados excelentes tanto para o arquivo gerado pelo Módulo TTS como para o arquivo gravado do áudio de uma pessoa, além de diminuir o processamento necessário na identificação dos fonemas.

6.2 Realização dos Testes no Dispositivo Móvel

Os códigos em Java foram embarcados no dispositivo móvel facilmente. Entretanto, o desempenho apresentado para carregamento e processamento do arquivo de áudio foi baixo, tendo o aparelho gasto 40s para carregar o arquivo de áudio e apresentar a animação da face, um tempo exageradamente grande. Realizou-se então uma investigação para descobrir a causa de tal demora, sendo observado que o motivo do baixo desempenho era o carregamento do arquivo externo de áudio usado para teste, que leva 30s no smartphone HTC Touch.

Analisando o desempenho apenas após a carga total do áudio, e após a realização de algumas otimizações no código tentando simplificar as instruções, obtiveram-se os tempos de 202ms para carregamento do arquivo WAV, 606ms para cálculo dos coeficientes do LPC e de 5.200ms para identificação das vogais pela rede neural e sua associação a um visema, tendo então um tempo total de aproximadamente 6s até a reprodução da animação, o que é considerado um tempo aceitável.

CONSIDERAÇÕES FINAIS

Este trabalho apresentou um sistema que sintetiza voz a partir de textos fornecidos como entrada, gera animações e as sincroniza com o áudio produzido. Simulando assim movimentos de lábios e face.

No início do desenvolvimento foi adotado o Java como plataforma de desenvolvimento, porém dificuldades de integração entre o framework proprietário de TTS escolhido e os módulos de processamento do áudio forçou uma mudança para a utilização do .NET, suportado pelo equipamento testado e de simples utilização.

Houve problemas na identificação das vogais, porém estes foram resolvidos através de modificações no número de neurônios na camada intermediária da rede neural e na forma como as amostras do áudio eram analisadas. Problemas de sincronia apareceram após a mudança no desenvolvimento do Java para o Compact .NET, devido a uma limitação do *framework* para a manipuação de arquvios de áudio, tal problema foi amenizado através de mudanças no controle da sincronia, que passou a ser gerenciada por um temporizador, mas uma solução definitiva envolvendo a execução do áudio encontra-se em estudo.

O desenvolvimento do protótipo apresentou como subproduto uma ferramenta de ajuste de parâmetros de modelo, que se mostrou muito útil para outras aplicações a serem desenvolvidas nessa mesma linha e um crescimento indiscutível de conhecimento para os membros desenvolvedores.

No estado atual, o protótipo é capaz de realizar o processo inteiro a que se propõe (etapas que se

iniciam no recebimento da mensagem e se encerram na exibição da animação com áudio) em um tempo de 6s, um resultado considerado bom dentro dos parâmetros atuais, principalmente quando se leva em conta que o equipamento utilizado nos testes, o HTC Touch, possui um pequeno poder de processamento, se comparado a outros modelos existentes no mercado.

REFERÊNCIAS

[Byorick, 2003] Jeff Byorick, Ravi P. Ramachandran e Robi Polikar (2003). Isolated Vowel Recognition Using Linear Predictive Features and Neural Network Classifier Fusion.

[Dorf , 2006] Richard C. Dorf, Circuits, Signals, and Speech and Image Processing, CRC, 2006, p. 16-1 a 16-13.

[freetts, 2008] Disponível eletronicamente em http://freetts.sourceforge.net/docs/index.php - acesso em 13/06/2008.

[Höfele, 2004] Claus Höfele. Mobile 3D Graphics: Learning 3D Graphics with the Java Micro Edition.

[Joone, 2008] Joone - Java Object Oriented Neural Engine. Disponível em <http://www.jooneworld.com> - acesso em 13/07/2008.

[Kshirsagar, 2005] Sumedha Kshirsagar e Nadia Magnenat-Thalmann. (2005). Lip Synchronization Using Linear Predictive Analysis, Geosynthetics'87, IFAI, New Orleans, LA, USA, Vol. 1, p. 95-107.

[Morgadinho, 2008] Nuno Morgadinho e Cláudio Fernandes. (2003). Voice Coder, Universidade de Evora.

[Parks, 1987] T.W. Parks e C.S. Burrus.(1987). Digital Filter Design. Editor Wiley, New York.

[Proakis, 2000] J.G. Proakis e D.G. Manolakis. (2000). Digital Signal Processing-Principles, Algorithms and Applications. Editora Prentice-Hall, New Delhi.

[Thalmann, 2004] Magnenat-Thalmann e Thalmann. (2004). Handbook of Virtual Humans. Editora Wiley, Inglaterra.

[Virtual Vision, 2008] MicroPower. Disponível em http://www.micropower.com.br/v3/pt/acessibilidade/vv5/index.asp – acessado em 18/07/2008.

[Jaws, 2008] JAWS For Windows Overview. Disponível em http://www.freedomscientific.com/fs_products/software_jaws.asp – acessado em 18/07/2008.

[SmartRead SDK, 2008] SmartySoft[smartysoft.com]: SmartRead Mobile TTS SDK. Disponível em http://www.smartysoft.com/smmobile/sdk.html – acessado em 07/04/2008.

[IPA, 2008] International Phonetic Alphabet. Disponível em http://en.wikipedia.org/wiki/International_Phonetic_Alphabet. - acessado em 07/04/2008.

[São Francisco, 2008] Fonemas Vogais do Inglês e do Português. Disponível em http://www.colegiosaofrancisco.com.br/alfa/ingles/fonemas-vogais-do-ingles-e-do-portugues.php - acessado em 01/08/2008.

[Salgado, 2001] Paula Lucena Salgado. (2001). Experimentos com sincronização de audio e vídeo, Rio de Janeiro, RJ, Brasil. Dispnoível em http://www.telemidia.puc-rio.br/~pslr/mestrado/disciplinas/eo/docs/EO_Paula_versao1.pdf - acesso em 20/07/2008.