UNIVERSIDADE FEDERAL DO PARÁ
INSTITUTO DE TECNOLOGIA
FACULDADE DE ENGENHARIA DA COMPUTAÇÃO E TELECOMUNICAÇÕES

Controle de Aparelhos Eletrônicos por Sistemas Embarcados: Uma Solução com Suporte à Reconhecimento e Síntese de Voz

Belém – Brazil
Abril/2015

Controle de Aparelhos Eletrônicos por Sistemas Embarcados: Uma Solução com Suporte à Reconhecimento e Síntese de Voz

Cassio Trindade Batista
cassio.batista.13@gmail.com
201106840003

Gabriel Peixoto de Carvalho
gaburiero.c@gmail.com
201106840010

Pedro Henrique C. F. Soares
pedrofigueiredoc@gmail.com
201106840007

Thiago Barros Coelho
tbarroscoelho@gmail.com
201106840040

Projeto apresentado à disciplina Projeto de Harware de Interfaceamento como requisito de avaliação. Professores: Jeferson Leite e Adalbery Castro.

Belem – Brazil
Abril/2015

Sumário

Lista de Figuras

1 Introdução

A interface homem-máquina encontra-se cada vez mais amigável. O que antes era portado somente por empresas e pessoas com poder financeiro diferenciado e acima da média, em termos de tecnologia, é hoje muito mais acessível e simples para usuários domésticos sem profundo conhecimento no assunto. Diversos estudos vêm sendo desenvolvidos a fim de melhorar ainda mais essa comunicação, de modo que a máquina se aproxime mais de ações típicas do ser humano, como pensar e falar.

Síntese e reconhecimento automático de voz (do inglês *text-to-speech* e *automatic speech recognition*, respectivamente, TTS e ASR) [**?**, 1] tornam a interface citada acima muito mais prática e natural, de forma que a comunicação de fato se assemelha àquela estabelecida entre duas pessoas. O ASR refere-se ao sistema que, tomando o sinal de fala digitalizado como entrada, é capaz de gerar o texto transcrito na saída. Já um sistema TTS realiza a função contrária, na qual um sinal analógico de voz é sintetizado de acordo com o texto posto na entrada. São inúmeras as aplicações que utilizam tais sistemas envolvendo processamento de voz. Dentre elas, pode-se destacar a automação residencial com foco em acessibilidade.

De acordo com [**?**, **?**], tecnologia assistiva (TA) é um campo da engenharia biomédica dedicada à aumentar a independência e mobilidade de pessoas com deficiência, englobando metodologias, práticas e serviços que objetivam promover sua autonomia, qualidade de vida e inclusão social. Tal tecnologia busca reduzir a necessidade vivenciada por pessoas que precisam de soluções que não as deixem à margem da utilização de dispositivos eletrônicos. Em outras palavras, para diminuir a exclusão digital imposta pela incapacidade de manipular certos dispositivos, a acessibilidade é vista como elemento fundamental para elevar a autoestima e o grau de independência dessas pessoas. Além disso, a implementação da acessibilidade também pode ser útil para os não portadoras de necessidades especiais, já que o controle de equipamentos se torna mais prático e confortável.

Nesse sentido, este trabalho busca preparar um servidor local portátil de reconhecimento de voz em Português Brasileiro (PT_BR) e de síntese de voz baseado no microcomputador BeagleBone Black de modo que, quando acessado pelo dispositivo que agirá como controle remoto — no caso, um smartphone com sistema operacional Android —, seja capaz de acessar as funções mais básicas de um aparelho televisivo. Vale ressaltar que todas as APIs e softwares utilizados para criação dos sistemas e dos recursos utilizados (com exceção do HTK, o qual será visto mais adiante) possuem licença *open source* e são encontrados disponíveis livremente na Internet.

2 Objetivos

O objetivo principal consiste em criar um protótipo portável, baseado em uma plataforma embarcada, que seja capaz de controlar um aparelho de televisão através do envio remoto de sinais. O sistema será configurado como um servidor que disponibiliza um serviço genérico de reconhecimento de fala, de modo que o aparelho de TV mencionado possa ser remotamente controlado através da voz do usuário; e um serviço de síntese de fala, provendo *feedback* das ações de acordo com o entendimento do sistema de ASR.

2.1 Reconhecimento Automático de Voz

Para que o reconhecimento automático de voz seja possível, o *software* Julius deverá ser instalado no servidor. Julius é um software capaz de processar e decodificar áudio em aproximadamente tempo real para tarefas de ditado de até 60 mil palavras.

Para que o Julius possa realizar o reconhecimento em Português Brasileiro, serão necessários basicamente dois recursos: um modelo acústico e um dicionário fonético. Modelos acústicos genéricos para PT_BR podem ser encontrados na página do Grupo FalaBrasil [**?**], bem como o software que cria o dicionário fonético (conversor grafema-fonema ou G2P) [**?**]. Entretanto, embora a taxa de acerto dos modelos seja satisfatória, é possível melhorá-la através da criação ou treino de modelos específicos para a aplicação.

O processo de treino será realizado pelo software HTK (acrônimo para kit de ferramentas dos modelos ocultos de Markov, livremente traduzido do inglês), o qual é capaz de extrair segmentos de fala de

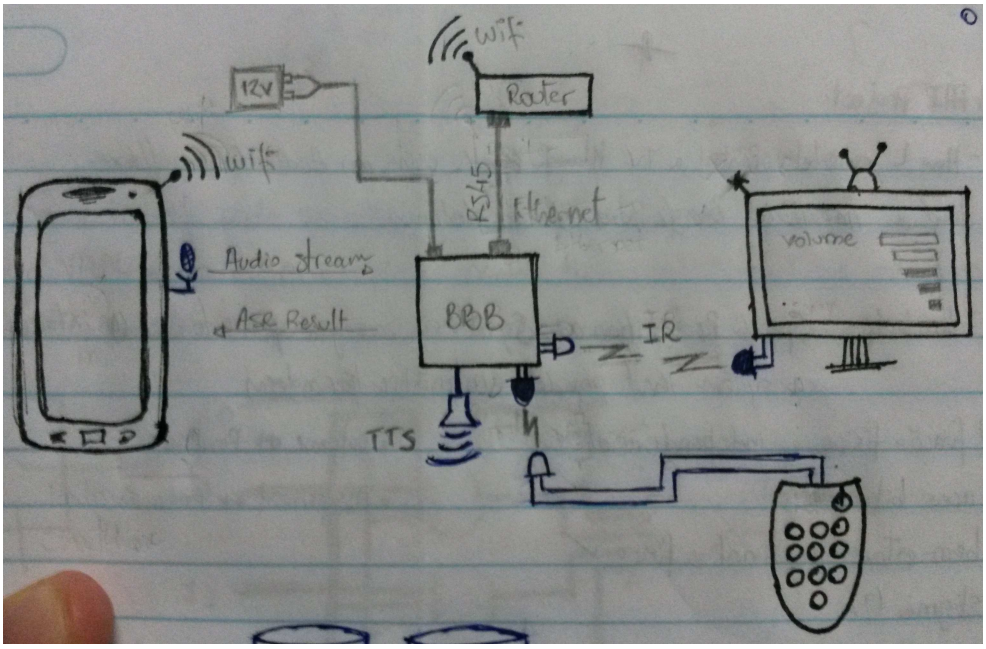

Figura 1: Novo van Gogh

um arquivo de áudio e assinalar uma referência à ele. Tal referência é retirada do dicionário fonético, previamente criado com o software G2P.

2.2 Síntese de Voz

O software eSpeak é a principal referência em síntese de voz em ambientes Linux. Graças à disponibilização de uma API do eSpeak no site oficial do desenvolvedor, o sistema, além de conseguir "ouvir e entender", também será capaz de "falar". O download dos modelos para PT_BR é feito juntamente com o das bibliotecas necessárias. Como a BBB não possui saída de audio nativa, será feito o uso de um auto-falante USB.

2.3 Servidor LAMP

PResdro
Predro
Pesro
Porredo
PResdro

2.4 Controle Remoto de TV

Os aparelhos televisivos atuais, assim como a grande maioria dos dispositivos eletrônicos domésticos, possuem a tecnologia de controle remoto baseada em luz infravermelha. Pode-se observar que na extremidade superior dos controles remotos, há pelo menos um led infravermelho (IR Led) capaz de emitir luz e, dessa forma, transmitir uma informação binária para o circuito localizado na parte frontal da TV. Esse circuito possui um sensor infravermelho (IR sensor), o qual, atuando como o receptor da comunicação, é capaz

de receber os bits transmitidos e repassá-los para o processador do circuito, o qual executará a tarefa relacionada à decodificação dos bits (diminuir o volume, trocar de canal, etc).

Nesse sentido, o *datasheet* de uma TV específica será estudado para que a BBB possa transmitir o conjunto de bits exatos, reconhecíveis pela TV em questão, para a execução de determinadas tarefas, como "aumentar volume" ou "mudar para o canal 18", por exemplo.

3 Justificativa

Segundo o Instituto Brasileiro de Geografia e Estatística (IBGE), no censo realizado em 2010, aproximadamente 23,9% dos brasileiros declaram ter alguma deficiência [**?**]. Esse número ainda é mais impressionante quando se pensa que cerca de um quarto de uma população de 190 milhões de habitantes é portadora de alguma necessidade especial. Ainda segundo os dados, 6,9% (13,3 mi) dos brasileiros apresentam algum grau de deficiência motora, enquanto 18,8% (35,7 mi) afirmam serem cegas ou terem alguma dificuldade para enxergar.

Essa pesquisa tem como finalidade apresentar uma solução para diminuir a exclusão digital vivenciada especialmente por pessoas com necessidade motora ou visual, as quais estão à margem da utilização de dispositivos eletrônicos por conta da ausência se soluções que os adaptem às suas necessidades. A tecnologia de reconhecimento de fala torna acessível a utilização de qualquer dispositivo eletrônico por usuários incapazes de realizar movimentos específicos com membros superiores, como segurar um controle físico e apertar botões ou digitar, por exemplo. Além disso, os portadores de necessidades visuais também poderão ser ajudados, já que nem todos os controles possuem referências reconhecíveis pelo tato. A síntese de fala também se torna muito importante no contexto da dificuldade visual, já que um *feedback* via texto, nesse sentido, seria pouco útil.

O Ato de Americanos com Deficiência (ADA) [**?**] é um documento que regula os direitos dos cidadãos com deficiência nos EUA, além de prover a base legal dos fundos públicos para compra dos recursos que estes necessitam. Algumas categorias de TA foram criadas com base nas diretrizes gerais da ADA, das quais podemos salientar três como justificativa do trabalho:

1. Recursos de acessibilidade ao computador
    - Equipamentos de entrada e saída (síntese de voz, Braille), auxílios alternativos de acesso (ponteiras de cabeça, de luz), teclados modificados, softwares especiais (reconhecimento de voz, etc.), que permitem as pessoas com deficiência a usarem o computador.
2. Sistemas de controle de ambiente
    - Sistemas eletrônicos que permitem as pessoas com limitações moto-locomotoras controlar remotamente aparelhos eletro-eletrônicos, sistemas de segurança, entre outros, localizados em seu quarto, sala, escritório, casa e arredores.
3. Auxílios para cegos ou com visão subnormal
    - Auxílios para grupos específicos que inclui lupas e lentes, Braille para equipamentos com síntese de voz, grandes telas de impressão, sistema de TV com aumento para leitura de documentos, publicações etc.

A decisão de criar um servidor próprio de síntese e reconhecimento de voz baseia-se principalmente na possibilidade de usufruir de tais recursos de forma offline, ou seja, sem a necessidade de conexão com a Internet. No caso do sistema ASR, pode-se também citar a vantagem de limitar o vocabulário de palavras utilizados através da implementação de uma gramática, já que serviços online de reconhecimento (como o disponibilizado pelo Google, por exemplo), trabalham com a inteira modelagem das palavras do idioma, impossibilitando a criação de um contexto específico para a aplicação. Além disso, o uso de APIs externas

não resulta no aprendizado sobre a filosofia do reconhecimento de fala, o que tornaria o trabalho menos interessante.

Add some conclusion to the section here

4 Revisão Teórica

Como o iOS e o Android foram lançados, respectivamente, em 2007 e 2008, e sendo a ascensão dos smartphones relativamente recente, a ideia de aplicar acessibilidade no controle de equipamentos eletrônicos através desse tipo de equipamento somente começou a revelar resultados concretos a partir de 2010. Em [**?**], o decodificador PocketSphinx foi embarcado em um smartphone android para que este pudesse controlar aparelhos domésticos através da interface de voz. O resultado era enviado para uma SparkFun IOIO Board, a qual disparava o *trigger* para o controle de uma TV. O foco do trabalho era ajudar pessoas afetadas com tetraplegia a serem mais independentes, além de avaliar o desempenho de decodificadores embarcados e distribuídos, claro.

Add another reference here

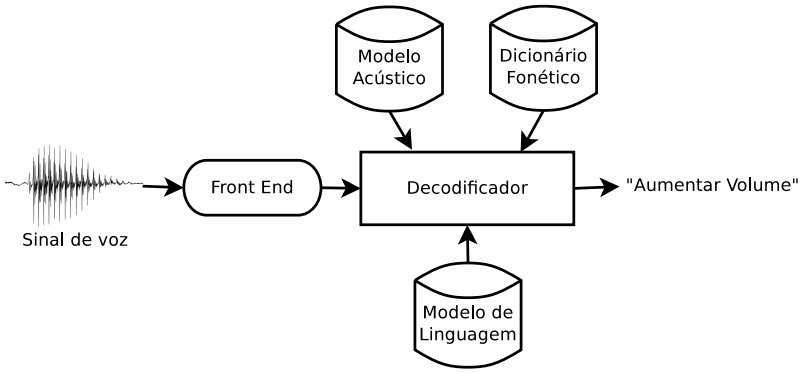

Figura 2: Esquemático de um sistema ASR

Os fundamentos do reconhecimento automático de voz, assim como os da síntese de voz, são descritos com bastante detalhes em [1]. A arquitetura mais geral e aceita na literatura é mostrada na Figura 2. Vale salientar que, ao invés do uso de modelos mais gerais, que descrevem a maior parte de uma linguagem, serão utilizadas gramáticas livres de contexto, as quais limitam o vocabulário utilizado a apenas um conjunto de sentenças possíveis, escolhidas pelo desenvolvedor do sistema. A construção do dicionário fonético para PT_BR dar-se-á através do software descrito em [**?**]; o tutorial para o treino do modelo acústico encontra-se disponível na página do projeto Voxforge [**?**], bem como capítulo 3 do livro do HTK [**?**]; a gramática reconhecida pelo Julius é criada manualmente de acordo com o descrito na página oficial [**?**]. Instruções de configuração e utilização do Julius encontram-se na documentação oficial [**?**].

Como saída analógica do sistema TTS, a voz sintetizada deve ser reproduzida por um dispositivo externo à BeagleBone, já que esta não possui auto-falantes próprios. Como visto em [**?**], o dispositivo primário de saída de áudio da BeagleBone é o HDMI, o qual pode ser desabilitado mediante modificações em parâmetros do kernel. Feito isso, o USB, que é o dispositivo secundário, se torna o principal, fazendo com que a solução mais simples seja plugar um auto-falante (*speaker*) na porta USB. Na página oficial do eSpeak, um arquivo de cabeçalho (*header*) permite a utilização de uma API em C/C++, a qual facilita o acesso aos módulos do software que permitem que a BeagleBone "fale" [2].

A escolha da plataforma foi fundamental para a esquematização do projeto. Arduino, Raspberry Pi e BeagleBone Black foram as três principais plataformas a serem escolhidas. Diversos tutoriais de comparação entre as plataformas foram consultados e estão disponíveis na Internet [**?**, **?**, **?**]. O Arduino, apesar de ser uma ferramenta flexível e com grande capacidade de interfaceamento com uma vasta quantidade de

dispositivos, é uma plataforma simples, recomendada para projetos de menor porte. O microcontrolador, que pode ser programado em C, se torna muito limitado quando o projeto requer um servidor estável e relativamente potente; O Raspberry Pi, por ser bastante completo, pode ser considerado um mini computador. Todo o seu armazenamento é fornecido por um cartão SD, além de ser possível conectá-lo à Internet através de um conector Ethernet. Sendo necessário a instalação de um sistema operacional, o Raspberry Pi ainda possui interface de saída HDMI e é muito útil para aplicações gráficas.

Tabela 1: Comparação entre as três principais plataformas

	Arduino UNO	BeagleBone Black	Raspberry Pi
Chip	-	TI AM3359	BCM2835 SoC full HD
CPU	ATMega328	1 GHz ARM Cortex-A8	700 MHz ARM1176JZ-F
GPU	-	PowerVR SGX530	Dual Core VideoCore IV
Armazenamento	2 kB SRAM	512 MB DDR3	512 MB SDRAM
Flash	32 kB	2 GB on-board eMMC, MicroSD	SD, MMC, SDIO card slot
GPIO	14	65	8
Video	-	mini HDMI	HDMI
OS	-	Linux	Linux
Amperagem (mA)	42	210-460	150-350
Voltagem (V)	7-12	5	5
USB	-	1 Host, 1 Mini Client	2 Hosts, 1 Micro Power
Ethernet	-	1 10/100 Mbps	1 10/100 Mbps
Preço	5 conto	300 conto	200 conto

A BeagleBone é comparável ao Raspberry Pi. Entretanto, por ter mais pinos (GPIO) e um processador mais poderoso, a BeagleBone é uma escolha óbvia para projetos mais elaborados. Além de possuir diversas opções de conexão, a BeagleBone une a flexibilidade de interfaceamento do Arduino com a capacidade de processamento rápido do Raspberry Pi. Apesar da desvantagem no preço, não restaram muitas dúvidas no momento da escolha dessa plataforma para o projeto. Uma comparação entre os principais parâmetros dos três equipamentos é dada na Tabela 1:

O funcionamento de controles remotos, com ênfase nos baseados em luz infravermelha para televisores, é explicado de de forma clara e detalhada em diversos tutoriais para "curiosos" disponíveis na Internet, como os da revista Mundo Estranho [**?**] e do blog *How Stuff Works?* da UOL [**?**].

5 Metodologia

O servidor, por ser o elemento chave na consolidação do projeto, deve ser o módulo a ser prioritariamente configurado, a fim de ser preparado para atender às devidas requisições, bem como executar qualquer tipo de aplicação solicitada. Sendo assim, a instalação da plataforma Ångström foi tomada como o primeiro passo. Ångström [**?**] é um sistema operacional, baseado em Linux, preparado exclusivamente para plataformas embarcadas, sendo o padrão para a própria BeagleBone. As dependências a serem instaladas são listadas na Lista 1.

É importante ressaltar que os sistemas operacionais embarcados são simplificações de sistemas operacionais mais robustos, tendo a maior parte das suas funcionalidades reduzidas ou simplificadas para se adequar à uma plataforma de menor porte. Por isso, a preparação deve ocorrer a partir dos pacotes mais básicos, como o GCC, por exemplo. Outros pacotes devem ser instalados de forma gradual, tais quais os requeridos pelo Julius, eSpeak e os necessários para a implementação do servidor LAMP em C.

Listing 1: Pre-instalação de dependências no servidor

```
# general dependences
build-essential alsa-tools sox

# eSpeak dependences
libespeak-dev libportaudio2 libportaudio-dev

# Julius dependences
libasound2 libasound2-dev

# LAMP dependences
apache2 libapache2-mod-fastcgi
php5 libapache2-mod-php5 php5-mcrypt # PHP (opcional?)
mysql-server libapache2-mod-auth-mysql php5-mysql # MySQL
phpmyadmin (opcional?)
libmysqlclient-dev # C
libmysqlcppconn7 libmysqlcppconn-dev # C++ (opcional?)
```

Em [**?**], o Julius foi configurado para funcionar em modo servidor através da opção nativa "-adinnet" (A/D *Input from Network*, conversão A/D com entrada pela rede). Isso permite que o Julius receba amostras de áudio via *streaming* através de uma comunicação com um cliente genérico via *socket*. O código foi alterado para que o resultado gerado pelo Julius, também conhecido como sentença, seja retornado ao cliente através desse mesmo *socket*. Além disso, uma aplicação foi construída sobre a plataforma Android 2.3 exclusivamente para se comunicar com o servidor. Basicamente, as amostras de áudio obtidas pelo microfone do aparelho são enviadas, enquanto são paralelamente analisadas a fim de se detectar o silêncio do fim da fala do usuário. Feito isso, o aplicativo apenas aguarda a sentença a ser enviada pelo servidor.

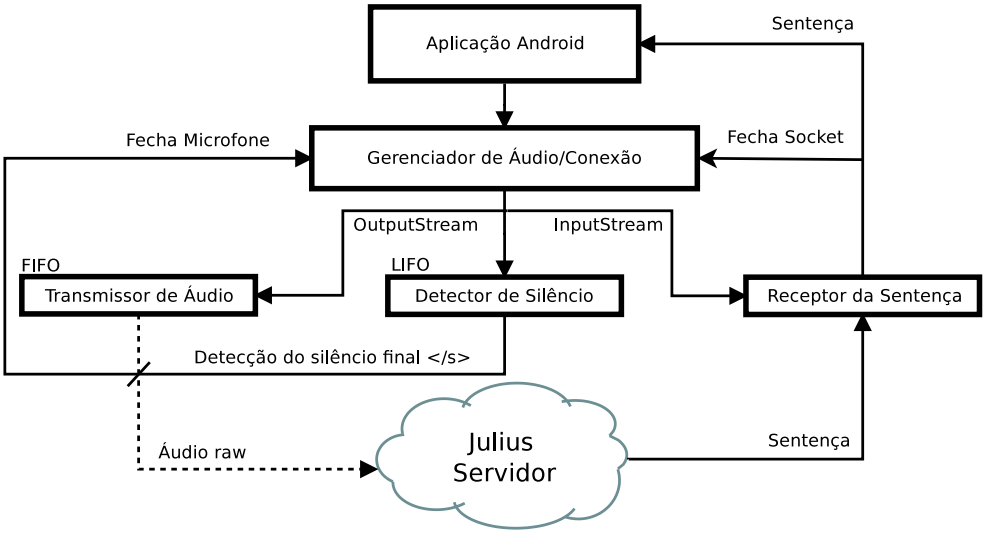

Figura 3: Esquemático do Cliente LaPS CSR.

A construção do dicionário fonético para o PT_BR se dá por meio do *software* `lapsg2p`, o qual recebe uma lista de palavras como entrada e gera suas transcrições fonéticas, conforme visto na lista abaixo, à direita. Já a gramática é utilizada para restringir o vocabulário, de modo a gerar somente uma das senteças listadas, como mostrado na lista abaixo, à esquerda. A construção da gramática no formato do Julius utiliza diretamente o dicionário fonético em seu escopo.

```
<s> aumentar volume </s>
<s> diminuir volume </s>
<s> canal mais </s>
<s> canal menos </s>
<s> canal <número> </s>
<s> ligar televisão </s>
<s> desligar televisão </s>
<s> cadastrar controle </s>
...
```

```
aumentar   a u~ m e~ t a X
diminuir   dZ i~ m i~ n u j X
televisão  t e l e v i z a~ w~
volume     v o l u~ m i
canal      k a n a w
um         u~
dois       d o j s
três       t r e j s
...
```

A proposta do trabalho é adicionar funcionalidades ao código do Julius, permitindo a produção de voz sintetizada através da incorporação da API do eSpeak e a transmissão de informação para a TV através de um led IR conectado a um GPIO. Um sensor IR ficará encarregado de receber informações de diferentes controles remotos para que sejam guardadas como registros no banco de dados.

6 Orçamento

Produto	USD (U$)	BRL (R$)	IOF (R$)	Total (R$)
BBB				
Smartphone				
IR Led				
IR Sensor				
USB Speaker 8 Ω				
Total				500 conto

7 Dificuldades e Soluções

- I was thinking about use the LAMP server to store data transferred from different kinds of remote controllers, so we could have a database fulfilled with information that would allow us to control a great range of TV devices. However I don't know how hard is to access the database from the Julius source. I mean, the default way to access a database is from PHP Code. I've seen Java for web that can incorporate SQL commands into the code, json and stuff like that, but I think to implement a code in C/C++ to push and pull data to a database is a monkey job. If some of you can try to deal with databases managed by C++ I would be glad.

- There's no sound output on BBB. The default audio device, as mentioned in some foruns on the web, is through the HDMI connector. The same forum has said that we can disable the HDMI as audio device by changing some kernel parameters, then the second device would become the main one. In this case, the USB would be the principal sound card. The easist way, then, would be attach an USB speaker to the connector, because the TTS would automatically output the result to that port. My point about the USB speaker:

  – It must be low power consuming

  – Its size must be limited. If the project was to build a thermistor plugged on the wall, we could not put a big speaker there. The circuit must be smaller than the BBB itself.

  It would be nice if we could build an amplifier and put the sinthesized audio through a BBB GPIO, but...

  - I don't know how to manipulate the TTS output to put in that pin
  - Even If I could do that, It's hard to play a sound with PWM. A chunck of code, size dependent of the sampling rate, should be put in that GPIO in a slot of time. And I think it's not as simple as this description.
  - The arduino codes I've found that play sound through a pin have the craziest codes I've ever seen. Almost everything is manually done.

- There's an arduino code that is capable to "hack" the information coming from any remote control. It acts as the receiver that we find in front of any TV device by using an IR sensor. Besides that, we could read the datasheet of the TV we're gonna use.

Referências

[1] X. Huang, A. Acero, and H. Hon, *Spoken Language Processing*. Prentice-Hall, 2001.

[2] "eSpeak text to speech," Visitado em Julho, 2014. http://espeak.sourceforge.net/.