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

Uso de Reconhecedor e Sintetizador de Voz Embarcados para Controle de Equipamentos Eletrônicos

Belém – Brazil
Abril/2015

Uso de Reconhecedor e Sintetizador de Voz Embarcados para Controle de Equipamentos Eletrônicos

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
201106840041

Projeto apresentado à disciplina Projetos de Harware e 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. Devido a abrangência de computadores pessoais e embarcados e da Internet, novas oportunidades e expectativas em termos de trabalho, estudos e até lazer são criadas 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.

Acredita-se que a síntese e o reconhecimento automático de voz (do inglês *text-to-speech* e *automatic speech recognition*, respectivamente, TTS e ASR) [1, 2] 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. Dentre as inúmeras as aplicações que utilizam os sistemas que envolvem processamento de fala, pode-se destacar a automação residencial que visa ajudar pessoas que tenham dificuldades no controle de alguns equipamentos eletrônicos, dando ênfase à acessibilidade alcançada através das tecnologias assistivas.

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 [3, 4]. 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, as mesmas soluções apresentadas podem ser úteis também 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 e síntese de voz em Português Brasileiro (PT_BR) 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, IDEs, *softwares*, bibliotecas e pacotes utilizados para criação dos sistemas e dos recursos utilizados possuem licença *open source* e são encontrados disponíveis livremente na Internet.

2 Objetivos

O objetivo principal consiste em criar um protótipo portável, baseado em um microcomputador embarcado, 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. Além disso, a informação a ser enviada para a TV deve ser armazenada em um banco de dados, também configurado no mesmo servidor.

2.1 Reconhecimento e Síntese 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. Este também será o principal programa do sistema, o qual receberá em seu código nativo todos os outros módulos.

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

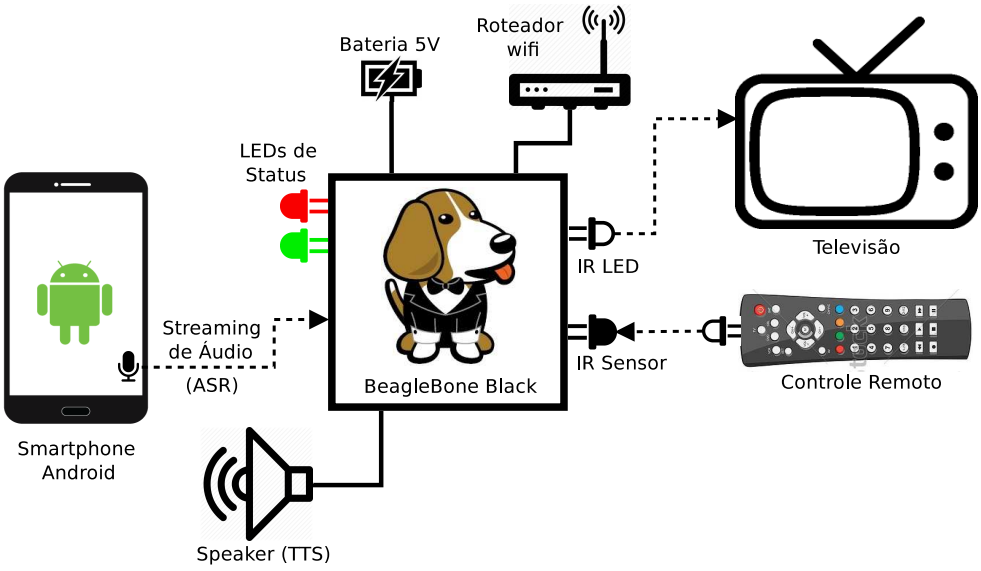

Figura 1: Novo van Gogh

PT_BR podem ser encontrados na página do Grupo FalaBrasil [5], bem como o software que cria o dicionário fonético (conversor grafema-fonema ou G2P) [6]. Entretanto, embora a taxa de acerto dos modelos seja satisfatória, ainda há casos nos quais a acurácia do modelo não é suficiente. Nesse caso, é possível melhorá-la através da criação ou treino de modelos específicos para a aplicação. O processo de treino é realizado pelo software HTK (*Hidden Markov Models Toolkit*), o qual é capaz de extrair segmentos de fala de um arquivo de áudio e assinalar uma referência à ele. Tal referência é retirada do dicionário fonético, previamente criado com o software G2P.

O software eSpeak é a principal referência em síntese de voz em ambientes Linux. Graças à disponibilização de uma API 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 áudio nativa, um auto-falante USB será utilizado.

2.2 Controle Remoto de TV e Servidor LAMP

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 (desligar a TV, mostrar o menu, etc).

Nesse sentido, um módulo receptor será adicionado à BeagleBone para que esta seja capaz de "hackear", através de um sensor infravermelho, informações dos controles remotos, dado que o acesso aos *datasheets* de diversos aparelhos não é trivial. O sistema de gerenciamento de banco de dados MySQL será instalado e configurado no servidor para armazenar o código, o qual é relacionado a ações como "aumentar volume" ou "desligar" da TV principal e dos demais aparelhos que vierem a ser cadastrados no banco. Toda a informação de saída a ser enviada para a TV através dos IR leds deve ser oriunda do banco de dados.

3 Justificativa

No censo realizado em 2010 pelo Instituto Brasileiro de Geografia e Estatística (IBGE), aproximadamente 23,9% dos brasileiros declaram ter alguma deficiência [7]. Esse número é realmente impressionante, pois revela que cerca de um quarto de uma população de 190 milhões de habitantes é portadora de necessidades especiais. 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.

Em [8], uma pesquisa foi realizada com brasileiros portadores de necessidades especiais para se determinar quais as características que esses cidadãos gostariam de adicionar, se pudessem, em controles remotos convencionais, visando minimizar suas dificuldades em utilizá-los. A sugestão de um *design* mais limpo foi unânime, enquanto os deficientes visuais, em particular, apontaram a necessidade de algum tipo de *feedback* quando os botões fossem pressionados e a implementação de alguma referência reconhecível pelo tato nos botões. Já os deficientes motores sugeriram um dispositivo menor, que não escorregasse facilmente das mãos, contendo um ângulo de controle mais abrangente e com botões maiores para aqueles que não conseguem ter controle absoluto sobre as mãos e/ou dedos. O estudo também mostrou que as referências sentidas pelo tato não foram incorporadas no design a fim de se manter o baixo custo.

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 de 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 hápticas. A síntese de fala também se torna muito importante no contexto da dificuldade visual, já que, nesse sentido, um *feedback* por fala será muito mais útil do que via texto. Em [9] é sugerido que as interfaces multimodais (métodos alternativos de controle como voz, escrita, gestos, etc.) associadas ao controle de equipamentos provêem uma experiência muito mais prazerosa e prática ao usuário do que os métodos convencionais (botões), apesar de diminuir a usabilidade e aumentar o tempo de execução da tarefa.

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 ressaltar 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. O fato de as comunicações ocorrerem em LAN diminui muito o tempo de espera do usuário, já que o áudio não precisa acessar servidores de voz distantes. Ainda sobre o 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.

4 Revisão Teórica

Como o iOS e o Android foram lançados, respectivamente, em 2007 e 2008, e a ascensão dos smartphones é relativamente recente, a ideia de adicionar a eles a acessibilidade no controle de equipamentos eletrônicos somente começou a revelar resultados concretos a partir de 2010. Em [10], 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, estando fisicamente conectada ao controle da TV, disparava um determinado *trigger*. A justificativa do trabalho era ajudar pessoas afetadas com tetraplegia a serem mais independentes; os testes avaliaram o desempenho de decodificadores embarcados e distribuídos.

Cassio: Add another reference here

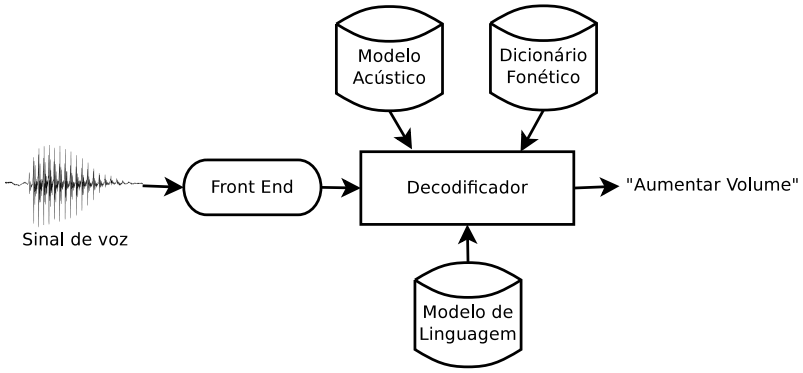

Figura 2: Esquemático de um sistema ASR

Os fundamentos do reconhecimento automático de voz, assim como os da síntese de voz, são descritos com bastante detalhes em [2]. 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 [6]; o tutorial para o treino do modelo acústico encontra-se disponível na página do projeto Voxforge [11], bem como capítulo 3 do livro do HTK [12, p. 22–42] (lembrando que o modelo acústico utilizado também está disponível na página do FalaBrasil); a gramática reconhecida pelo Julius é criada manualmente de acordo com o descrito na página oficial [13]. 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 [14], 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" [15].

A escolha da plataforma foi fundamental para a esquematização do projeto. Arduino, Raspberry Pi e BeagleBone Black foram as três principais opções a serem escolhidas. Diversos tutoriais de comparação entre as plataformas foram consultados e estão disponíveis na Internet [16, 17, 18]. 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, torna-se muito limitado quando o projeto requer um servidor estável e relativamente potente; O Raspberry Pi, por ser bastante completo, já se enquadra no conceito de 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, tornando-se 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	20 MHz 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 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.

↓ Thiago: LIRC ↓ Gabriel: IR theory ↓ References + Figures + Tables ↓

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 [19] e do blog *How Stuff Works?* da UOL [20]. A maioria dos aparelhos eletrônicos atualmente recebe informação através de sensores infravermelhos localizados em painéis frontais. Um grande problema é a inteferência que surge com a vasta transferência de informação via IR. Devido à isso, a comunicação entre o controle remoto e a televisão ocorre geralmente em 4 passos: um comando *start* inicia a transferência, seguido dos bits do comando específico (como aumentar o volume, por exemplo) e do endereço do aparelho. Por fim, um comando de *stop* encerra o envio de bits. Dessa forma, a chance de a informação ser reconhecida por mais de um aparelho é baixa (salvo o caso de serem dois equipamentos do mesmo tipo e da mesma empresa).

Uma das grandes dificuldades relacionadas à comunicação IR para controle de equipamentos é que cada empresa praticamente segue o seu próprio padrão de transmissão de informação: o número, a ordem e o significado dos bits é variado, a modulação e codificação usada são diferentes e a frequência dos pulsos pode oscilar entre 32 kHz e 40 kHz, chegando a 50 kHz em determinados aparelhos mais modernos. Além disso, tão raro quanto o seguimento de uma comunicação IR padronizada, a disponibilização de documentação pelos fabricantes também é bastante escassa. A Philips, por exemplo, utiliza seu próprio protocolo de

comunicação chamado RC-5. A última documentação foi liberada em 1992, quando ainda não existiam muitos dos equipamentos eletrônicos atuais, como *home theaters* ou DVDs. Nesse padrão, os bits são codificados de acordo com o código de Manchester, onde cada bit, transmitido dentro de um período fixo, é representado por uma transição *high-to-low* (0) ou *low-to-high* (1). Esses padrões de dados são obtidos através de uma operação do tipo XOR (OU-Exclusivo) realizada entre o clock do dispositivo e o dado propriamente dito [21].

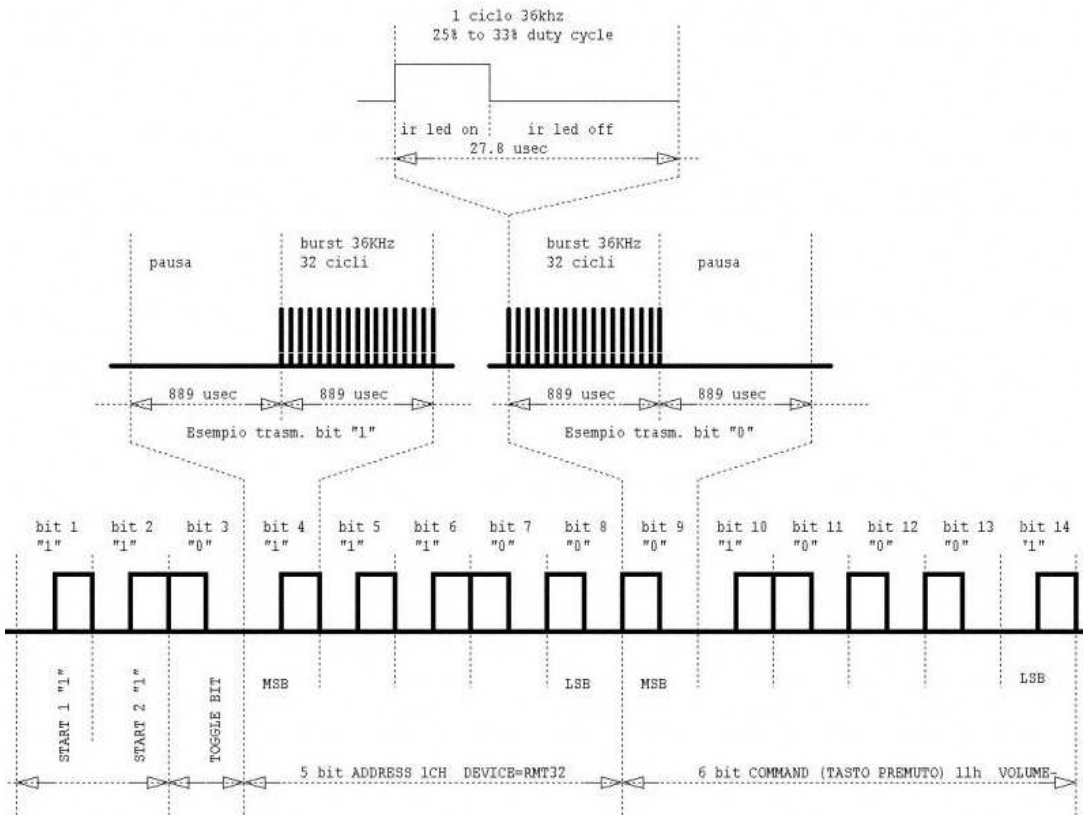

Figura 3: Esquemático do protocolo RC-5 da Philips.

A Figura 3 mostra a informação referente ao comando "diminuir volume" contida num vetor de 14 bits. Os 11 últimos bits definem o endereço do aparelho de destino e o comando em si. Pode-se observar que qualquer bit é representado por duas partes, sendo uma metade em nível baixo e a outra, em nível alto. Cada nível ocorre num intervalo de 32 perídos. O nível alto é gerado por um PWM de *duty cycle* igual a 25% do período do pulso. Essa percentagem define o tempo em que o IR Led permance aceso, ou seja, no nível alto, o IR Led permanece ligado por período de $0,25 \times 1/36\ kHz$ e desligado por $0,75 \times 1/36\ kHz$, sendo o processo repetido 32 vezes.

Os aparelhos mais novos já implementam a versão atualizada desse protocolo, chamado de RC-6. Embora a Philips não tenha disponibilizado qualquer documentação sobre este protocolo, há fóruns na Internet que, através da aplicação de engenharia reversa, conseguem descrever o padrão utilizado na construção do sinal. Pela Figura 4, é fácil notar que a quantidade de bits carregada pelo sinal (em comparação com o RC-5) aumentou de 14 para 22. O código Manchester também aparece invertido, já que valor lógico alto passa agora a ser representado pela transição *high-to-low* (1), enquanto o valor baixo (0) é definido por uma mudança *low-to-high*. O primeiro bit de *start* tem uma duração maior, para garantir o ganho do AGC no circuito receptor; já o segundo, também de *start*, é sempre mantido em valor alto. O bit de *toggle*, o qual muda de estado caso uma tecla deixe de ser pressionada, também possui uma duração

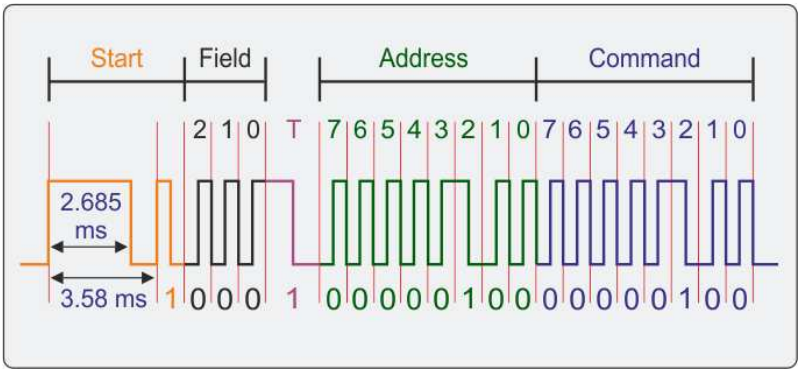

Figura 4: Esquemático do protocolo RC-6 da Philips.

mais longa do que os outros bits comuns. Por fim, os últimos 14 bits representam o endereço do aparelho e o referido comando a ser transmitido, respectivamente.

4.1 Produtos Relacionados

Existem alguns produtos disponíveis no mercado com a finalidade de tornar o controle de equipamentos eletrônicos mais prático. Um deles é o *Harmony Smart Control* [22], o qual possui em seu "pacote" um aplicativo para iOS e Android (porém sem versão para tablets; somente smartphones), um hub e um controle remoto genérico. Segundo a revisão da CNET, vale a pena pagar US$ 130 por todas as funcionalidades que o sistema apresenta, como usar uma conexão RF entre o smartphone/controle com o hub (que, infelizmente, ainda não conseguiu se livrar do tão antiquado IR), o que faz com que o usuário não precise apontar o controle para o dispositivo que precisa controlar. Todavia, o hub precisa estar numa boa posição para conseguir emitir a informação de forma clara para o aparelho desejado.

IRdroid é outro aplicativo que permite o controle de aparelhos televisivos com o celular [23]. Como o próprio nome sugere, funciona apenas em dispositivos Android, desde que seu módulo de hardware esteja acoplado na saída de audio jack do smartphone. Versões mais recentes já possuem o hardware que pode ser acessado via bluetooth, custando US$ 60 o mais caro. A grande vantagem é que o IRdroid, além de ser baseado na biblioteca LIRC, a qual possui uma vasta quantidade de equipamentos em seu banco de dados, possui código livre e disponível.

Outras diversas soluções são aplicadas apenas à *smart TVs*, onde a informação de controle é transmitida por wifi. Nenhuma das aplicações encontradas para TVs convencionais possui suporte à reconhecimento e síntese de voz offline em PT_BR.

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 do sistema operacional Ubuntu foi tomada como primeiro passo. As dependências a serem instaladas são mostradas 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 alsa-base alsa-utils sox

# eSpeak dependences
libespeak-dev libportaudio2 libportaudio-dev

# Julius dependences
libasound2 libasound2-dev

# LAMP dependences
apache2 libapache2-mod-fastcgi # apache server
mysql-server libapache2-mod-auth-mysql php5-mysql # MySQL
libmysqlclient-dev # C
phpmyadmin # (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.

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> ligar televisão </s>
<s> desligar televisão </s>
<s> cadastrar controle </s>
<s> selecionar controle </s>
```

```
aumentar    a u~ m e~ t a X
diminuir    dZ i~ m i~ n u j X
volume      v o l u~ m i
canal       k a n a w
mais        m a j s
menos       m e~ n u s
televisão   t e l e v i z a~ w~
...
```

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.

Toda a organização do trabalho, bem como a comunicação entre os membros sobre os passos tomados no decorrer do trabalho será feita através da plataforma Trello [**?**].

Um controle para a TV Philips 39PFL3008D/78 foi usado como base para a análise do sinal emitido. Inicialmente, um arduino UNO fui utilizado para verificar o tempo em que o IR led permanecia ativo e inativo, armazenando-o em uma matriz de duas colunas. O Matlab foi utilizado para converter a matriz em um vetor único, no qual os índices ímpares representavam o tempo de duração do modo *burst* do IR led e as posições pares, o tempo em que o IR led permanecia *idle*. Sendo assim, o vetor no qual as durações dos nívels altos e baixos alternavam-se entre si foi convertido para uma forma de onda quadrada, semelhante

à mostrada na Figura 4.

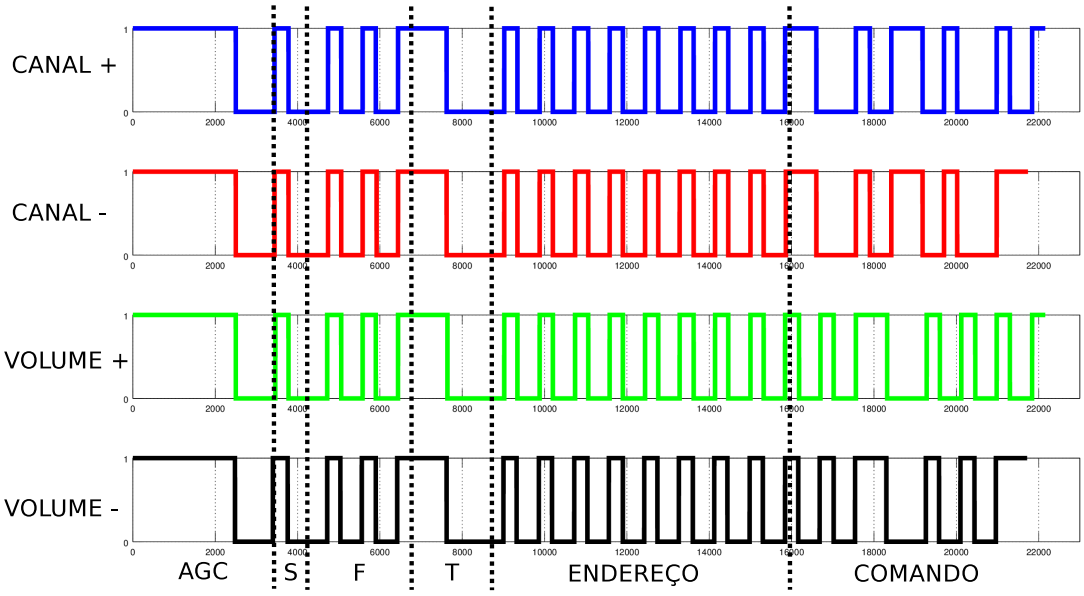

Figura 5: Mudança nos bits de comando após pressionar NÃO consecutivamente 4 botões diferentes.

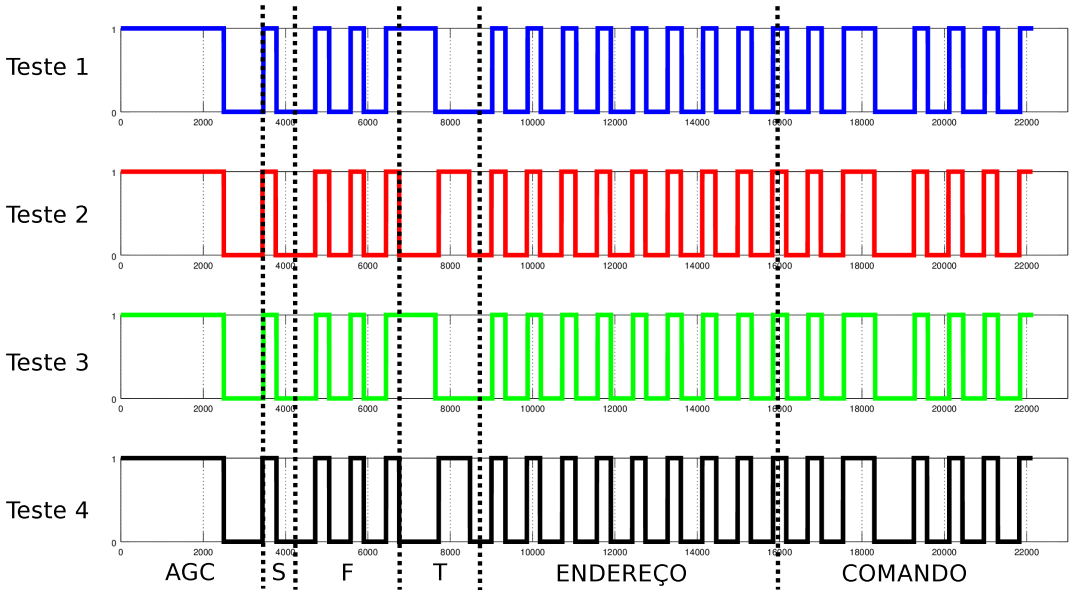

Figura 6: Mudança no bit de Toggle após pressionar o botão “Volume Mais” por 4 vezes consecutivas.

Predro: Detalhes sobre a modelagem do DB
Thiago: Detalhes sobre o que foi feito com o Arduino
Gabriel: Detalhes sobre o que foi convertido do Arduino para a BBB

6 Orçamento

Tabela 2: CARO PRA CARALHO ME AJUDA AI GILMA

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

1. A BeagleBone não possui saída de áudio nativa, tampouco conectores do tipo audio jack. No Debian, a saída padrão de áudio é pelo conector HDMI, mas pode ser substituída pelo USB mediante modificações em parâmetros do kernel. O modo mais fácil, considerando que redirecionar a saída do eSpeak para um GPIO seria muito trabalhoso, seria conectar um auto-falante USB à BeagleBone. Em se tratando de um protótipo, um headphone faz o papel de um speaker que deve consumir pouca energia e ter um tamanho limitado. Foi-se cogitada a construção de um circuito com um amplificador LM386 para o speaker, porém a obtenção de um D/A PCM2707 não custaria menos de US$ 15.
2. A BeagleBone não possui conexão wifi. Além da dificuldade em atualizar o kernel pra receber um shield/case, o mesmo teria de ser conectado na porta USB, a qual já estaria sendo usada pelo auto-falante. Portanto, a solução mais fácil foi conectar um roteador wifi à porta Ethernet da BBB através de um cabo RJ-45.
3. O Angstrom é PODRE. (Gabriel)
4. O código do Arduino que hackeia as paradas é PODRE. (Thiago)
5. Acessar MySQL from C é PODRE (Pedro)

8 Trabalhos Futuros

De acordo com o relato disponível em [**?**]: "Dado que o meu home theater é modesto, ele requer que eu consiga manejar APENAS 6 controles remotos para a simples tarefa de assistir a um filme". Seria marveolous se houvesse um controle remoto universal que permitisse acesso à TODOS os aparelhos do ambiente residencial, mesmo os que estão em cômodos diferentes do que eu estou agora. Seria mais maravilhoso que esse controle estivesse sempre com você. E que pudesse usá-lo mesmo quando estivesse fora de casa. E que fosse acessível por uma tecnologia hands free. Compre já o seu!

- Expandir para vários aparelhos, tornando a beagle beagle um servidor centralizado no ambiente doméstico
- Em cada compartimento onde houvesse um aparelho eletrônico a ser controlado, haveria um microcontrolador (a ser avaliado, preferencialmente mais barato que o arduino) capaz de controlar determinado(s) aparelhos
- A beagle beagle e todos os outros microcontroladores estariam conectados à mesma rede LAN. Somente a beagle beagle precisaria estar conectada à internet, de modo que não houvesse limitação de distância para a conexão com o smartphone.

Referências

[1] P. Taylor, *Text-To-Speech Synthesis*. Cambridge University Press, 2009.

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

[3] *Comitê de Ajudas Técnicas. Tecnologia Assistiva*. Brasília, Brasil: Subsecretaria Nacional de Promoção dos Direitos da Pessoa com Deficiência., 2009.

[4] G. J. Gelderblom and L. P. de Witte, "The assessment of assistive technology: Outcomes, effects and costs," IOS Press. Technology and Disability 91-94, 2002.

[5] "FalaBrasil: Reconhecimento de Voz para o Português Brasileiro," Visitado em Julho, 2014. `http://www.laps.ufpa.br/falabrasil/`.

[6] A. Siravenha, N. Neto, V. Macedo, and A. Klautau, "Uso de regras fonológicas com determinação de vogal tônica para conversão grafema-fone em Português Brasileiro," *7th International Information and Telecommunication Technologies Symposium*, 2008.

[7] "Perfis do censo demográfico. Visitado em Janeiro," 2010. `www.ibge.gov.br/`.

[8] L. C. P. C. et. al, "Accessibility in digital television: Designing remote controls," IEEE Transactions on Consumer Electronics, 2012.

[9] I. Wechsung and A. B. Naumann, "Evaluating a multimodal remote control: The interplay between user experience and usability," IEEE, 2009.

[10] S. B. et. al, "Designing android applications with both online and offline voice control of household devices," IEEE, 2011.

[11] "Tutorial: Create Acoustic Model," Visitado em Abril, 2015. `http://www.voxforge.org/`.

[12] S. e. Young, *The HTK Book*. Microsoft Corporation, Version 3.0, 2000.

[13] "Open-Source Large Vocabulary CSR Engine Julius," Visitado em Julho, 2014. `http://julius.sourceforge.jp/en_index.php`.

[14] "USB Audio on the BeagleBone," Visitado em Abril, 2015. `http://andicelabs.com/2014/03/usb-audio-beaglebone/`.

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

[16] "Raspberry Pi or Beaglebone Black," Visitado em Abril, 2015. `http://michaelhleonard.com/`.

[17] "Arduino vs. Raspberry Pi vs BeagleBone," Visitado em Abril, 2015. `http://randomnerdtutorials.com/`.

[18] "Arduino Uno vs BeagleBone vs Raspberry Pi," Visitado em Abril, 2015. `http://makezine.com/`.

[19] "Como funciona o controle remoto?," Visitado em Abril, 2015. `http://mundoestranho.abril.com.br/`.

[20] "Como funcionam os controles remotos," Visitado em Abril, 2015. `http://tecnologia.hsw.uol.com.br/`.

[21] "RC-5 Philips Procotol," Visitado em Abril, 2015. `http://en.wikipedia.org/wiki/RC-5`.

[22] "Logitech Harmony Smart Control," Visitado em Abril, 2015. `http://www.logitech.com/en-us/product/harmony-smart-control`.

[23] "IRdroid: Universal Remote for Android," Visitado em Abril, 2015. `http://www.irdroid.com/`.

A Pedro: Codigos LAMP

B Thiago: Codigos Arduino/C for IR Tx/Rx