UNIVERSIDADE FEDERAL DE OURO PRETO
ESCOLA DE MINAS
COLEGIADO DO CURSO DE ENGENHARIA DE CONTROLE E AUTOMAÇÃO - CECAU

ACIONAMENTO DE DISPOSITIVOS VIA DECODIFICAÇÃO DE SONS UTILIZANDO A TRANSFORMADA RÁPIDA DE FOURIER EM UM MICROCONTROLADOR

MONOGRAFIA DE GRADUAÇÃO EM ENGENHARIA DE CONTROLE E AUTOMAÇÃO

MARIA AMÉLIA PEREIRA

OURO PRETO, 2017

MARIA AMÉLIA PEREIRA

ACIONAMENTO DE DISPOSITIVOS VIA DECODIFICAÇÃO DE SONS UTILIZANDO A TRANSFORMADA RÁPIDA DE FOURIER EM UM MICROCONTROLADOR

Monografia apresentada ao Curso de Engenharia de Controle e Automação da Universidade Federal de Ouro Preto como parte dos requisitos para a obtenção do Grau de Engenheiro de Controle e Automação.

Orientador: Prof. Adrielle de Carvalho Santana

Ouro Preto

Escola de Minas – UFOP

2017

P436a Pereira, Maria Amélia.
Acionamento de dispositivos via decodificação de sons utilizando a Transformada Rápida de Fourier em um microcontrolador [manuscrito] / Maria Amélia Pereira. - 2017.

70f.: il.: color; tabs.

Orientador: Prof. Dr. Adrielle de Carvalho Santana.

Monografia (Graduação). Universidade Federal de Ouro Preto. Escola de Minas. Departamento de Engenharia de Controle e Automação e Técnicas Fundamentais.

1. Fourier, Transformações de - Transformada Rápida de Fourier. 2. Decodificadores (Eletronica). 3. Codificador de voz. 4. Sistemas de processamento da fala. I. Santana, Adrielle de Carvalho. II. Universidade Federal de Ouro Preto. III. Titulo.

Catalogação: ficha@sisbin.ufop.br CDU: 681.5

Monografia defendida e aprovada, em 03 de maio de 2017, pela comissão avaliadora constituída pelos professores:

Profa. M. Sc. Adrielle de Carvalho Santana - Orientadora

Prof. Dr. Agnaldo José da Rocha Reis – Professor Convidado

Prof. M. Sc. Danny Augusto Vieira Tonidandel – Professor Convidado

AGRADECIMENTOS

Primeiramente, à minha família pelo apoio, à minha mãe Márcia, meu pai Carlos Roberto, meus irmãos e ao meu avô Noé pela confiança. Esta vitória também é de vocês!

Aos amigos pelos incentivos, pelo apoio, carinho e por serem motivos de sorrir. Sem eles, essa luta não teria sido vencida.

Agradeço ainda a todos os professores da UFOP pelos ensinamentos, principalmente àqueles do DECAT. Em especial a professora Adrielle, pela oportunidade e por todo conhecimento adquirido nesse projeto.

À cidade de Ouro Preto por ter proporcionado tantos momentos de diversão e aprendizado, além da acolhida, e à querida Escola de Minas pelo crescimento pessoal e profissional.

A todos que de alguma forma contribuíram para essa jornada, o meu muito obrigado!

“O insucesso é apenas uma oportunidade para recomeçar de novo com mais inteligência.”

(Henry Ford)

RESUMO

A decodificação de sons é uma ferramenta útil nos dias atuais em aplicações como sistemas de segurança e sistemas de escrita automática. Aplicações diversas podem ser geradas por meio do reconhecimento de determinados sons, incluindo a fala humana, podendo acarretar comodidade, praticidade e simplificação para inúmeras ações. Em se tratando da decodificação de voz, que é um método mais limitador devido à exclusividade da fala, e relativamente seguro, é possível a realização de acionamento de dispositivos de sistemas de segurança, como por exemplo, desligar ou ligar um alarme, ou ainda abrir uma porta. Com base nesse cenário, neste trabalho foi desenvolvido um sistema capaz de decodificar sons por meio de suas frequências, utilizando a Transformada Rápida de Fourier (FFT), e através dessa decodificação propor o acionamento de dispositivos. A primeira etapa para o uso do sistema é a gravação. O usuário pronuncia a palavra a ser utilizada e o sistema, ao realizar a FFT do sinal gravado, identifica as frequências características desse. Uma vez que a etapa de gravação esteja concluída, o sistema entra em funcionamento aguardando a pronúncia da palavra pelo usuário. Nessa etapa, é feita a FFT do sinal da palavra pronunciada e suas frequências identificadas são comparadas às da palavra gravada anteriormente. Se as frequências forem semelhantes (considerando uma faixa aceitável de erro), ocorre o acionamento do dispositivo desejado. Foram realizados testes com diversas palavras e observados suas magnitudes e frequências, utilizando posteriormente o reconhecimento de uma dada palavra para o acionamento de um LED. O acionamento pode ser feito também através de uma ligação telefônica, bastando ligar a saída de som do celular à entrada de som do sistema. Assim, é possível fazer o acionamento remoto da carga por voz. Os resultados obtidos comprovam a eficácia do método utilizado, tendo sido possível o reconhecimento das palavras desejadas. Melhorias ao método são necessárias para melhorar sua precisão e área de utilização, como gravação de mais palavras, diminuição da margem de erro, aumento do número de amostras e de frequências para caracterização.

Palavras-chave: Transformada Rápida de Fourier, decodificação de sons, reconhecimento de voz, acionamento de dispositivos.

ABSTRACT

Decoding is an useful tool in present days applications such as security and automatic writing systems. Various applications can be generated through the recognition of sounds, the human speech for example can be used to facilitate the use of machines and computers providing comfort and practicality for numerous actions. When it comes to decoding voice, that is more limiter and relatively safe method, it is possible to perform device activation of security systems, for example, to turn an alarm on and off, and open a door. Based on this scenario, in this work we developed a system able to decode sounds through its frequencies, using the fast Fourier transform, and through this decoding pronounces the word to be used and the system, to perform the Fast Fourier Transform (FFT) of the recorded signal, identifies the characteristics of this frequency. Once the recording step is completed, the system enters in operation and waits for the full pronunciation of the word by the user. If the frequencies are similar (considering an acceptable range of error) the desired device is activated. Tests were carried out with several words and observed their magnitudes and frequencies, using the recognition of a given word to drive an LED. The activation can be done also through a phone call simply by connecting the sound output from the phone to the system's input. Thus, it is possible to load remote activation by voice. The results prove the effectiveness of the method used, having been possible the recognition of words. Improvements on the method are required to further increase accuracy and usefulness, such as recording more words, decreasing the margin of error, increasing the number of samples and frequencies of characterizations.

Keywords: Fast Fourier Transform, sound decoding, voice recognition, device activation.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

AFT	*Arithmetic Fourier Transform*
DFT	*Discrete Fourier Transform*
DTMF	*Dual-Tone Multi-Frequency*
FFT	*Fast Fourier Transform*

SUMÁRIO

1. INTRODUÇÃO

A tecnologia evolui de forma intensa e rápida, fato que se pode observar diariamente com o aumento gradativo do surgimento de inovações científicas e tecnológicas. Essa revolução causa grande impacto no cotidiano da população, já que em sua grande maioria as descobertas tem a finalidade de tornar o trabalho, lazer e conforto mais eficiente e econômico para todos. Nesse âmbito pode-se citar o acesso remoto, termo utilizado para a conexão entre dispositivos eletrônicos sem a necessidade de conexão física entre eles, que pode ser realizada por meio da web e/ou ligações telefônicas, permitindo a mobilidade dos usuários e, dependendo da situação, aumentado a produtividade. O uso do acesso remoto se torna cada vez mais disseminado e pessoal com a popularização dos dispositivos móveis.

É possível utilizar comandos sonoros para realizarem acesso remoto, ou mesmo para uso presencial, como em casos de utilização para sistemas de segurança. A expansão de métodos utilizando comandos sonoros simplifica a infraestrutura, gera mais confiabilidade e otimiza a logística necessária para determinada ação ou controle.

Vários tipos de sinais sonoros podem ser utilizados para executar uma ação, como por exemplos, comandos por voz, áudios gravados, sinais DTMF (*Dual-Tone Multi-Frequency*), notas musicais, ruídos, entre outros. Esses sinais sonoros necessitam ser decodificados antes de serem utilizados para realizar qualquer atuação, ou seja, é necessário ser identificado características próprias de cada som emitido e reconhecê-las.

A decodificação pode ser realizada de diversas formas, sendo um exemplo o uso de filtros e processadores digitais de sinais (VERDAN, 2016). Um método bastante eficaz e recorrente para decodificação de sinais sonoros é o algoritmo da Transformada Rápida de Fourier, do inglês *Fast Fourier Transform* (FFT).

Neste cenário enquadra-se o corrente trabalho, que abordará os conteúdos brevemente mencionados nos parágrafos anteriores, utilizando-os em conjunto para a realização do acionamento de dispositivos empregando a decodificação de sons como instrumento para tal, utilizando a FFT.

1.1 Objetivo

Desenvolver um sistema de decodificação de sons da fala humana a fim de realizar o acionamento por voz de dispositivos utilizando a Transformada Rápida de Fourier em um microcontrolador.

1.2 Justificativa

Seguindo a forte tendência do uso da automação na segurança e para conforto, o uso de sons para acionamento e controle de dispositivos pode ser cada vez mais empregado. Seja para emprego em sistemas de segurança, para comodidade ou mesmo praticidade, a utilização de sons alcança um grande cenário.

Por ser tratar de um método limitador (devido à exclusividade da fala) e relativamente seguro, o uso da voz ou de sons específicos pode ser de grande utilidade para implementação de sistemas de segurança em casas, pequenas empresas, ou para acionamento de determinados dispositivos, visando praticidade para os usuários.

Trata-se então em um campo onde é possível haver vários estudos e desenvolvimentos para aprimoramento e/ou criação de técnicas para a decodificação de sons a fim de ser utilizado como sinais de comando, sons de segurança e alarme ou reconhecimento de voz.

1.3. Estrutura do Trabalho

O conteúdo deste trabalho é disposto da seguinte maneira:

No capítulo 1 temos uma apresentação sobre os assuntos referentes ao problema que se deseja resolver, a descrição dos objetivos e as justificativas relevantes. Já no capítulo 2 há a descrição de alguns trabalhos que cercam o tema de alguma maneira e sua forma de resolução, a fim de dar um embasamento científico. No terceiro capítulo é feita uma descrição geral do sistema implementado e feito um levantamento bibliográfico sobre os dispositivos, *softwares* e/ou algoritmos que serão utilizados como suporte no projeto. No quarto capítulo é realizada uma análise específica dos métodos e materiais utilizados para a decodificação do

som, explicitando as funções utilizadas no código computacional e explicando seus objetivos, além de ser feito um detalhamento do processo no geral. Foram realizados alguns testes para a verificação dos sons identificados, obtendo suas FFTs e especificando suas frequências, que serão demonstrados e especificados no capítulo 5. Já no capítulo 6 tem-se a conclusão do trabalho e sugestões para trabalhos futuros e, finalmente, têm-se as referências bibliográficas que foram utilizadas.

2. REVISÃO BIBLIOGRÁFICA

A Transformada de Fourier tem sido muito utilizada no que se refere à decodificação de sons, sendo o trabalho de Souza, Sobral Cintra e Oliveira (2005) um bom exemplo desse argumento, no qual foi proposto um algoritmo para estimar sinais harmônicos baseando-se em aproximações quantizadas do modelo de Fourier. Os autores justificam seu uso como uma boa referência para ser utilizado em tempo real quando há necessidade de grande cálculo de estimações de harmônicos.

Para entendimento do trabalho é necessário que se saiba que grande parte dos instrumentos musicais emite sons sonoros (sons produzidos quando a glote encontra-se fechada e a corrente de ar instiga as cordas vocais as fazendo vibrarem) e em menor proporção sons surdos (produzidos a glote está totalmente aberta, permitindo a passagem da corrente de ar, as cordas não vibram), e a avaliação espectral em tempo real é um meio necessário e preciso a ser utilizado em práticas relacionadas à música. Nos instrumentos musicais o som é formado basicamente por uma nota fundamental e por uma quantidade de harmônicos que o definem, sendo por isso que uma mesma nota musical, em instrumentos diferentes, produzem sons diferentes. A potência produzida pelo som é definida pela amplitude dos sinais produzidos. Geralmente a identificação do conteúdo harmônico de um sinal é feito através do algoritmo FFT. (SOUZA, SOBRAL CINTRA e OLIVEIRA, 2005).

Para Martin e Kim (1998 apud Souza, Sobral Cintra e Oliveira, 2005) a relação harmônica permite identificar qual o tipo de instrumento utilizado em um determinado trecho musical, através da estimativa das relações entre componentes harmônicas e a fundamental, que alimentam uma rede neural capaz de reconhecer padrões que identifiquem o instrumento utilizado.

O trabalho de Lima et al. (2004) reforça a credibilidade de Fourier propondo um método alternativo para a decodificação de sinais DTMF (*Dual-Tone Multi-Frequency*), baseado na Transformada Aritmética de Fourier (AFT).

Os autores, através de um levantamento bibliográfico, definiram a AFT como método utilizado pelo seu baixo grau de complexidade aritmética quando comparado às suas antigas versões e pela implementação com processamento paralelo, diminuindo consideravelmente o número de multiplicações necessárias se comparado às FFTs. A AFT simplifica o cálculo dos

coeficientes da Série de Fourier, sendo mais equilibrado devido à sua baixa complexidade aritmética e esforços computacionais para cálculo dos coeficientes.

Orfanidis (1996, apud Lima et al., 2004) apresenta DTMF como um sistema utilizado na telefonia, que combina dois tons senoidais em soma para cada dígito. Na figura 1 é demonstrado a combinações de frequências que geram os tons DTMF.

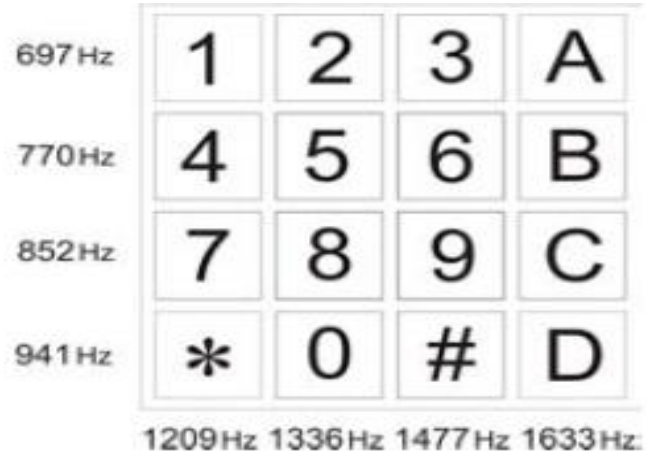

Figura 1 – Teclado DTMF
Fonte: Lima et al. (2004)

Logo depois de emitido e recebido, o sinal é analisado a fim de se identificar qual dígito ele representa. A análise é feita calculando a sua DFT (*Discrete Fourier Transform*), onde serão observadas as magnitudes das oito componentes associadas às frequências mais próximas das frequências DTMF. Ao identificar as duas componentes mais fortes, dá-se início a decodificação.

Para utilizar a AFT na análise do segmento frequencial de um sinal quantizado no tempo é necessário a definição de parâmetros que influenciam a precisão e a eficiência computacional. O primeiro a ser citado é a taxa de amostragem do sinal contínuo, a qual os autores definiram em 8kHz, por ser o valor real utilizado em sistemas telefônicos. Outro fator é o comprimento da transformada (N), sendo necessária a escolha de apenas um único valor que possa detectar as frequências DTMF. Além de imprescindível a realização de interpolações, e o tipo de interpolação escolhida afeta diretamente a inserção do erro no cálculo e no custo de uma implementação prática do algoritmo. No trabalho os autores decidiram por utilizar um programa em que se utiliza interpolação linear, ou seja, de primeira ordem.

Foi priorizado pelos autores que o resultado da análise do sinal não seja afetado pela sua fase, através da garantia de que o número de pontos possibilite remover as raias espectrais desejadas. Apenas dois harmônicos fortes devem ser presentes no sinal, sendo as outras componentes insignificantes perante o que se pretende detectar.

O procedimento utilizado por Lima et al. (2004) reduz significativamente a complexidade aritmética da AFT simplificada quando se deseja decodificar mais sinais DTMF. Utilizando a taxa de amostragem de 8 kHz e N=114, 16 multiplicações e 470 adições se fizeram necessárias para o cálculo das oito componentes que correspondem às frequências DTMF. Concluindo-se que o método proposto é mais eficiente em termos de complexidade computacional se comparado às técnicas geralmente utilizadas.

Petry, Zanuz e Couto Barone (1999) desenvolveram um trabalho em que utilizaram técnicas de processamento digital de sinais para reconhecimento de pessoas pela voz, exemplificando a possível utilização desse processo na área de segurança.

Para a aquisição de voz, eles utilizaram um microfone ligado a um filtro passa baixa, com a saída ligada a uma placa de som instalada em um computador. As amostras coletas são então processadas por um algoritmo de detecção de limite de palavras, que reconhece o momento em que uma palavra começa e termina, e guarda essas informações em um arquivo. Na figura 2 é demonstrado o algoritmo para realizar a detecção de limite das palavras.

Fizeram então um pré-processamento do sinal de voz, para organizar as amostras de tal modo que seja possível ler suas componentes, para serem utilizadas no algoritmo de reconhecimento de padrões. Essa etapa é dividida em duas subetapas: a pré-ênfase, onde se pretende excluir uma tendência espectral de -6dB/oitava presente naturalmente na fala, a divisão do sinal em frames e janelamento, objetivando dividir o sinal em pequenas partes, e utilizando a janela *hamming*, que possui características atenuantes na transição entre frames e adjacentes.

Logo em seguida é necessário ser realizado a extração dos parâmetros, ou seja, obter os coeficientes que representam um frame de voz. Neste trabalho foram utilizados os coeficientes cepstrais e os mel-cepstrais (obtidos com o auxílio da Transformada de Fourier direta e inversa), que diminuem a quantidade de dados sem afetar a informação necessária para o reconhecimento de fala. Os autores, para reconhecimento de padrões na fala, utilizaram a técnica Quantização Vetorial Multisecção. As etapas para identificar e classificar os padrões no vetor de coeficientes, extraídos de um frame de voz, são geração de codebook, quantização

de padrão desconhecido e comparação ou medida de distorção. (PETRY, ZANUZ e COUTO BARONE, 1999).

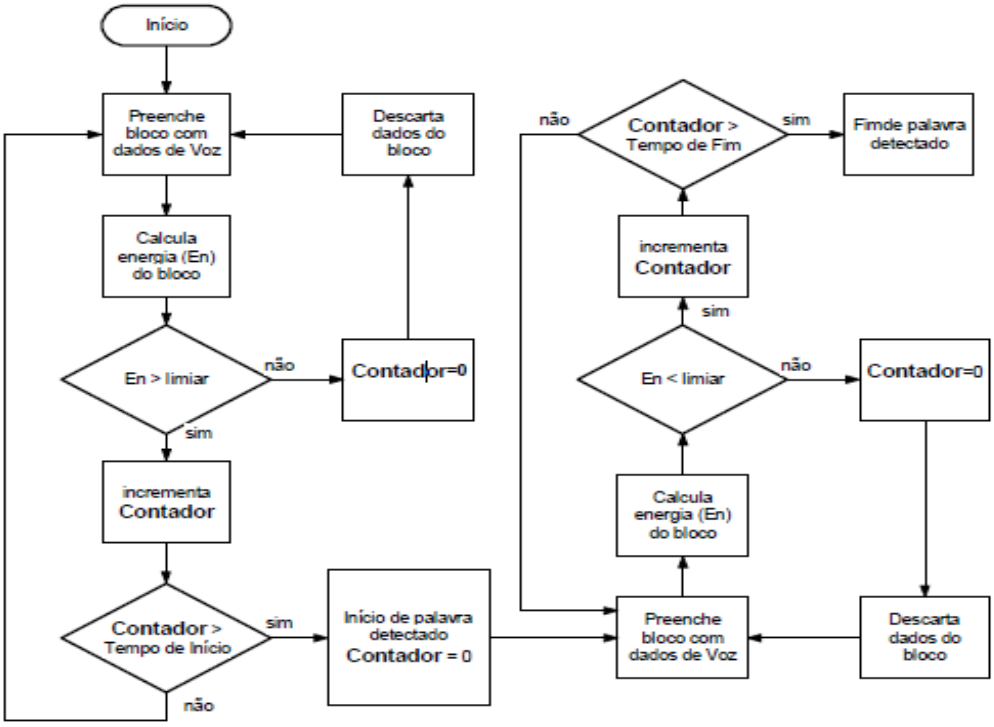

Figura 2 – Algoritmo para detecção de limite das palavras
Fonte: Petry, Zanuz e Couto Barone (1999)

Já Perico, Shinohara e Sarmento (2014), em seu trabalho de conclusão de curso, ousaram em utilizar a decodificação de sons para acionamento de dispositivos, apresentando a elaboração de um sistema para controle de uma plataforma elevatória através de reconhecimento de voz, com foco na área de tecnologia assistiva. No projeto, foi utilizada uma plataforma disponibilizada pelo Programa de Tecnologia Assistiva (PROTA), um minicomputador Raspberry Pi e, como decodificador de fala, foi utilizado o software Julius e sua variante Coruja.

Foi instalado um microfone de eletreto para a aquisição da voz e um sensor de presença, garantindo que o funcionamento do sistema apenas quando um usuário estiver na plataforma elevatória. Em seguida, o sinal de voz é transmitido para o computador e decodificado pelo software Julius, e a partir do dicionário pré-definido de palavras, será emitido um sinal para acionamento da plataforma utilizando relés. Foi desenvolvida também uma interface para o usuário acompanhar o estado do sistema, através de LEDs. O decodificador retorna três

parâmetros, sendo o sinal de voz em texto, nível de confiança com que esse sinal foi decodificado e o coeficiente de Viterbi, que determina o tamanho do caminho entre os fonemas traçados pelo código. Se esses parâmetros estiverem dentro de valores determinados, será então emitido um sinal para o acionamento da plataforma. (PERICO, SHINOHARA e SARMENTO, 2014).

Na figura 3 é mostrado o procedimento do sistema através de um fluxograma.

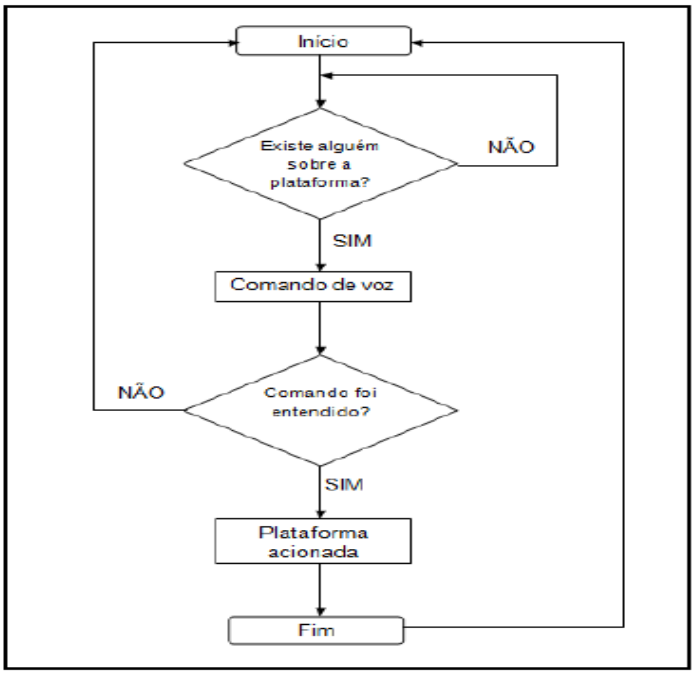

Figura 3 – Fluxograma do sistema de reconhecimento de voz
Fonte: Perico, Shinohara e Sarmento (2014)

Os autores utilizaram o decodificador Julius, que, de acordo com Lee (2010, apud Perico, Shinohara e Sarmento, 2014), é um software para reconhecimento de fala contínua, com código aberto e gratuito, que pode decodificar fala em tempo real através de dispositivos como o microfone, ou mesmo em arquivos de áudio. O software pode ser utilizado para reconhecer qualquer idioma, contanto que o idioma seja fornecido no dicionário, modelo de linguagem e modelo acústico, parâmetros que irão determinar a precisão da decodificação do som. Para o reconhecimento de voz em português brasileiro existe a interface denominada Coruja, que permite o controle do software Julius.

Através de um teste de confiabilidade, computado pelos autores, foi verificado as palavras que o software reconhecia mais facilmente, e assim selecionados para o acionamento da plataforma, que foram: ativa, ative, aciona, liga, ligue, anda, para, parada e parou.

Os autores concluíram que o sistema desenvolvido foi um sucesso, sem ser necessário o uso de internet para o seu funcionamento. Emitiram algumas observações sobre o microfone possuir limitações na captação do sinal de voz, tendo eles utilizado também um microfone de CFTV (Circuito Fechado de Televisão), com maior alcance, porém, por ser mais sensível, o microfone captou muitos ruídos, prejudicando assim o reconhecimento, sendo logo descartado. As falhas observadas são justificadas pela limitação do sistema mecânico, podendo-se citar movimento lento, ruído e mau contato.

Atualmente existem diversas ferramentas para reconhecimento de voz já utilizadas. Uma muito reconhecida e utilizada é o reconhecimento de voz do Google, que permite fazer pesquisas sem o uso da digitação, utilizando apenas a fala, presente em computadores e celulares. A taxa de erros no reconhecimento de voz da Google é de apenas 8% e, este fato é devido, segundo Kleina (2015), a:

> O segredo está no uso de *Deep Neural Networks* (Redes Neurais Profundas), um sistema interconectado e formado por camadas que envia quantidades imensas de dados para a inteligência artificial da empresa de forma parcelada. Desse modo, a máquina "aprende" determinada quantidade de reconhecimento, e de acordo com as respostas obtidas, recebe a próxima carga para corrigir erros, expandir idiomas e aprimorar o que ela já adquiriu.

Também é possível fazer a digitação e edição de documentos somente com a voz através de um recurso do Google, que está disponível no navegador Chrome e em *smartphones*, e o microfone do computador precisa estar ligado e funcionando.

Outra ferramenta que se pode citar, utilizada para o reconhecimento de voz é o *Voice Recognition Module V3*, demonstrada na Figura 4. Compatível com Arduino, é um módulo de reconhecimento de voz da fabricante Elechouse dependente de um microfone, com volume único, caracterizado por ser de fácil controle. É necessário treinar o módulo com comandos de

voz a fim de fazê-lo reconhecer tal comando. No total, 80 comandos de voz são suportados pela placa e, até 7 deles podem funcionar ao mesmo tempo. (ELECHOUSE, 2017).

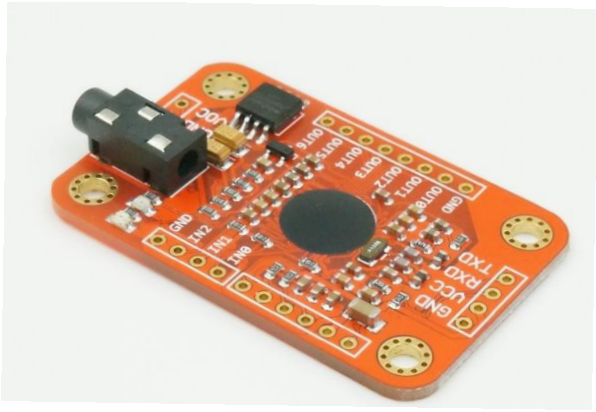

Figura 4: Voice Recognition Module V3
Fonte: ELECHOUSE, 2017

3. MATERIAIS E MÉTODOS

Neste capítulo será feito uma descrição geral do sistema implementado e um estudo inicial sobre os dispositivos, softwares e técnicas abordadas neste trabalho, que serão especificados por respectivos subcapítulos.

3.1 Descrição do sistema

O funcionamento do sistema proposto dá-se da seguinte forma: um som de uma palavra é transmitido a uma entrada analógica do microcontrolador Arduino Nano. Para substituir o uso do microfone portátil (não disponível) na fase de testes, foi utilizado o software Audacity (utilizando o microfone do computador para a gravação dos sons) para reproduzir uma palavra previamente gravada nele e foi feita ligação da saída de som do computador diretamente ao circuito do Arduino Nano. No microcontrolador é realizada a FFT do sinal deste som obtendo-se o espectro de frequências do sinal e é devolvido este espectro de frequências melhorado através do uso da janela *Hamming*, sendo possível assim fazer a decodificação da palavra emitida através de seus picos de frequências características.

Durante a fase de testes foi utilizada a Serial como auxílio para gravação e uso do sistema. Quando o sistema é iniciado um LED vermelho é aceso, à espera da pronúncia de uma palavra, e são gravadas suas frequências características. Esse processo se repetirá algumas vezes até que as frequências características se repitam uma quantidade específica de vezes, atenuando a possibilidade de frequências provenientes de ruído, serem tomadas como pertencentes ao sinal da palavra. Se a palavra for reconhecida o LED verde se acenderá para sinalizar a gravação, sendo possível em seguida acionar a carga, no caso o LED amarelo, ou reiniciar o processo para a gravação de uma nova palavra.

Caso seja escolhida a opção de acionar a carga após a gravação, o LED vermelho voltará a ficar aceso indicando que a palavra deve ser pronunciada. A FFT do som é realizada e as frequências características extraídas e comparadas àquelas armazenadas na etapa de gravação. Caso um número específico de frequências coincida, a carga é acionada e o LED vermelho se apaga; caso contrário, a carga não é acionada e o LED vermelho volta a se acender esperando uma nova pronúncia.

Botões nas entradas digitais 8 a 12 do Arduino possibilitam escolher algumas opções durante o funcionamento do sistema. Logo após a gravação de uma palavra o usuário tem a opção de gravar nova palavra, reiniciando o sistema (entrada 9) ou colocar o sistema em funcionamento para o acionamento de uma carga com a palavra gravada (entrada 8). Caso seja escolhida a opção do acionamento da carga, outra opção se torna disponível (entrada 12) na qual o usuário pode escolher gravar nova palavra em vez de acionar a carga, reiniciando assim o sistema. Caso a carga seja acionada, mais duas opções se tornam disponíveis, sendo uma a desativação da carga e rearme do sistema para acionar a carga novamente utilizando a palavra já gravada (entrada 10), e a outra opção é reiniciar o sistema com a gravação de nova palavra (entrada 11).

3.2 Arduino Nano

Em 2005, na Itália, Massimo Banzi e David Cuartielles criaram uma plataforma de prototipagem eletrônica denominada Arduino, caracterizada por ser *open-source*, de baixo custo e acessível a todos, inclusive àqueles sem conhecimento algum em programação e eletrônica (FILHO, 2014).

Existem inúmeras vantagens em se utilizar tal plataforma, como a facilidade que o *software* Arduino oferece ao usuário, o fato de poder ser utilizado sem ter que se pagar direitos autorais ou royalties e o hardware que se adaptou a diversas formas, dentre as quais pode-se citar Arduino Uni, Arduino Mega 2560, Arduino Nano, entre outros. (VIEIRA, 2011). Neste trabalho será utilizado o Arduino Nano.

O Arduino Nano é uma das menores versões de placas Arduino, assim também como a mais completa, tendo sido produzido pela Gravitech e baseada no ATmega328. Pode ser diretamente acoplado ao protoboard e possui um miniUSB, substituindo o usual. (ARDUINO, 2017).

Todos os 14 pinos digitais do Arduino Nano, que operam a 5 volts, pode ser utilizado como entrada ou saída, através das funções pinMode (), digitalWrite () e digitalRead (). Além disso, alguns pinos têm funções especializadas: o serial: 0 (RX) e 1 (TX) é usado para receber e transmitir dados em série TTL, interruptores externos (2 e 3): podem ser configurados para disparar uma interrupção em um valor baixo, um limite ascendente ou descendente ou uma alteração no valor, PWM (3, 5, 6, 9, 10 e 11): fornecem a saída PWM de 8 bits com a função

analogWrite (), SPI (10 (SS), 11 (MOSI), 12 (MISO), 13 (SCK)): dão suporte à comunicação SPI, LED 13: o LED se acende quando o valor é HIGH. (ARDUINO, 2017).

Na figura 5 pode-se observar a estrutura de hardware do Arduino Nano, com seus respectivos pinos e conexão para o cabo miniUSB.

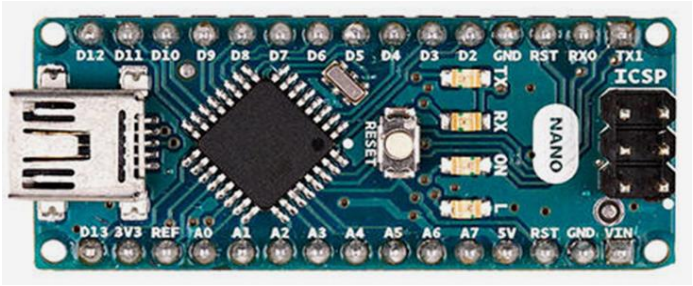

Figura 5 – Arduino Nano
Fonte: Arduino (2017)

O software Arduino possui duas funções extremamente importantes, sendo elas: Setup (), que se executa no início do programa a fim de iniciar as configurações da programação e Loop (), a qual é executada durante todo o processo, sendo interrompida somente quando o programa é finalizado ou o usuário o interrompa. Para o desenvolvimento de um código utilizando este software não é necessário mais do que uma noção básica de programação na linguagem C/C++. (VERDAN, 2016).

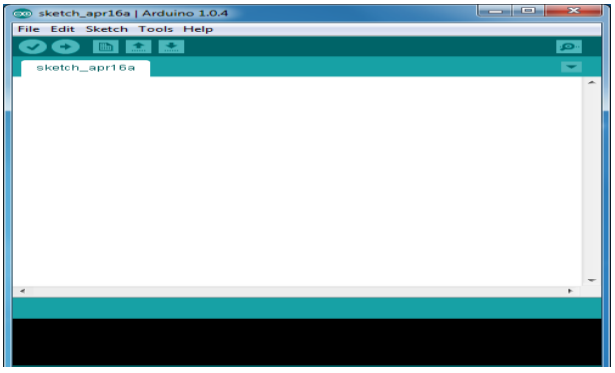

Figura 6 – Monitor serial da IDE do Arduino
Fonte: Arduino (2017)

Na figura 6 é possível observar o design do monitor serial da IDE do Arduino, que irá nos auxiliar durante a observação do comportamento da FFT (Transformada Rápida de Fourier).

3.3 FFT

Para entender o que é a Transformada Rápida de Fourier, é necessário, antes, introduzir os conceitos das Séries de Fourier e da Transformada de Fourier.

3.3.1 Séries de Fourier

Após pesquisas introdutórias de Euler, D'Alembert e Daniel Bernoulli, o físico Jean Baptiste Joseph Fourier (1768 – 1830) estudou meticulosamente as séries infinitas, futuramente denominadas séries de Fourier, em sua homenagem. Intitulado Théorie Analytique de la Chaleur, publicado em 1822, seu estudo de propagação de calor em corpos sólidos, onde essa propagação se dava por ondas de calor e sabendo-se que função senoidal é a forma de onda mais simples, foi o que culminou a descoberta das séries de Fourier, demonstrando que toda função, independente do nível de complexidade, pode ser decomposta como uma soma de cossenos e senos. Os resultados obtidos por Fourier foram aprimorados por Dirichlet e Riemann anos depois com maior precisão e formalidade. (AMORIM, ALVES e LOPES, 2017).

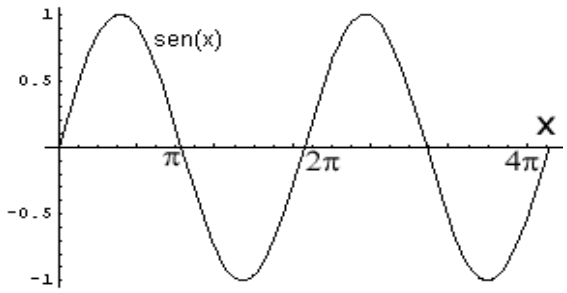

Figura 7 – Gráfico de sen(x)
Fonte: SEARA, 2017

Na Figura 7 está ilustrada a função sen(x), sendo x um ângulo medido em radianos. Esta função é periódica, ou seja, as curvas apresentam as mesmas características em intervalos subsequentes (períodos). O máximo valor da função, que é a distância entre o eixo horizontal e a crista da onda, denominado de amplitude, é igual a 1.

A função cos(x), representada graficamente na Figura 8, sendo também uma função periódica, é deslocada de $\pi/2$ em relação à função sen(x), isto é, diferem-se na fase de $\pi/2$.

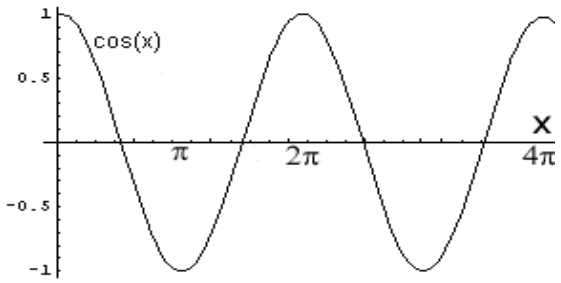

Figura 8 – Gráfico de cos(x)
Fonte: SEARA, 2017

Na Figura 9 é demonstrada a soma das funções sen(x) e cos(x), em vermelho, essa curva é obtida somando-se os valores de seno e cosseno de ponto a ponto.

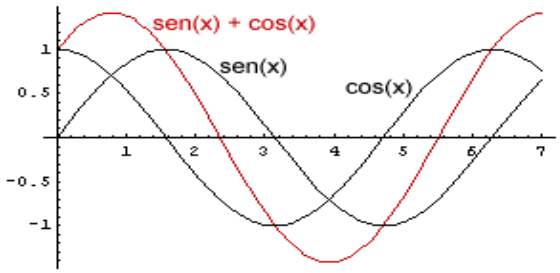

Figura 9 – Gráfico soma das funções sen(x) e cos(x)
Fonte: SEARA, 2017

Tomando-se uma função periódica, independente do grau de complexidade, segundo Fourier, ela pode ser representada como a soma de várias funções de seno e cosseno, escolhendo-se as fases e períodos de maneira favorável, de forma geral:

$$f(x) = a_0 + a_1 sen(x) + a_2 sen(2x) + a_3 sen(3x) + \ldots + b_1 cos(x) + b_2 cos(2x) + b_3 cos(3x) + \ldots \quad (1)$$

Sendo que quanto maior o número de termos na expansão da série de Fourier, melhor será a aproximação com a forma da função original. Os coeficientes a_0, a_1, a_2,..., b_1, b_2, b_3,... representam as amplitudes de cada onda, podendo ser inúmeros, dependendo da função que está sendo calculada. O que Fourier conseguiu foi justamente uma forma de calcular esses coeficientes, sendo dados por:

$a_0 \leq f(x) \geq$ média de f(x) em um período;

$a_n = 2 < f(x)\text{sen}(nx) \geq$ 2 vezes a média de f(x)sen(nx) em um período;

$b_n = 2 < f(x)\cos(nx) \geq$ 2 vezes a média de f(x)cos(nx) em um período.

A série de Fourier para a função degrau, também denominada onda quadrada, caracterizada pela oscilação entre amplitudes nula e máxima, como se pode observar na Figura 10, com seus cinco primeiros termos, é demonstrada pela seguinte equação:

$$f(x) = {}^{1}/_{2} + {}^{2}/_{\pi}\ \text{sen}(x) + {}^{2}/_{(3\pi)}\text{sen}(3x) + {}^{2}/_{(5\pi)}\ \text{sen}(5x) + {}^{2}/_{7\pi}\ \text{sen}(7x) + \cdots \quad (2)$$

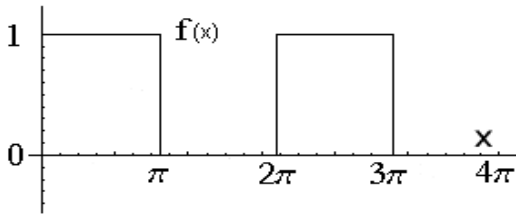

Figura 10 – Onda Quadrada
Fonte: SEARA, 2017

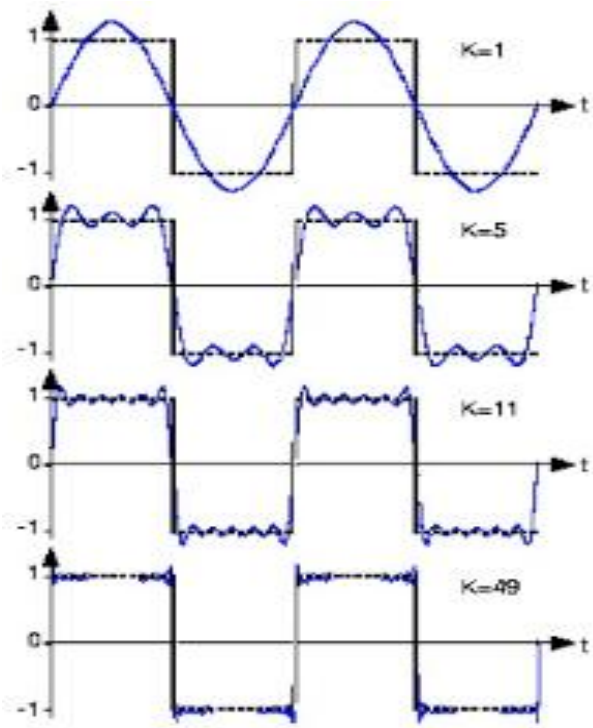

Figura 11 – Gráfico de Onda Quadrada com a expansão dos termos da Série de Fourier
Fonte: MELO, 2017

Na Figura 11 pode-se ver um gráfico de onda quadrada juntamente com o gráfico da expansão com os termos da série de Fourier, tendo sido expandida primeiramente com 1 termo e continuamente por 49 termos.

3.3.2 Transformada de Fourier

A série de Fourier nos mostrou como reescrever qualquer função periódica em uma soma de senoides e cossenoides. A Transformada de Fourier é a extensão dessa ideia para funções não-periódicas. (THE FOURIER, 2017)

A Transformada de Fourier decompõe qualquer função em uma soma de senos e cossenos. Cada uma dessas funções de base é uma exponencial complexa de uma frequência diferente, nos fornecendo uma maneira única de ver qualquer função. (THE FOURIER, 2017)

A transformada de Fourier garante a solução de problemas antes tidos como insolúveis, além de facilitar a solução de diversos outros. Sua notoriedade é devida às suas diversas aplicações nos campos da ciência e da engenharia, como por exemplo, na ressonância magnética, processamento de sinais, eletromagnetismo, física quântica e matemática teórica. (THE FOURIER, 2017)

Possui diversas aplicações nos campos da ciência e da engenharia, como: modulação de sinal, processamento de áudio e voz, projetos de filtros de frequência, processamento de sinais biomédicos, processamento de imagens, entre outras.

Nos sistemas de comunicação, o sinal da função é multiplicado por um sinal senoidal e o espectro da frequência é realocado, como pode ser obervado na Figura 12.

Nos sinais de áudio e de voz, para frequências baixas mais grave é o som e para altas frequências o som é mais agudo. Na Figura 13 pode ser obervado a filtragem no domínio da frequência.

A filtragem no domínio do tempo é dada por convolução e a filtragem no domínio da frequência é dada pela transformada aplicada às funções do filtro e do sinal, seguida de um produto e de uma transformada inversa.

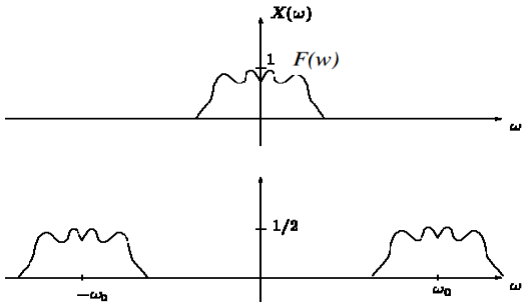

Figura 12 – Traslado do espectro de frequência
Fonte: FECHINE, 2010

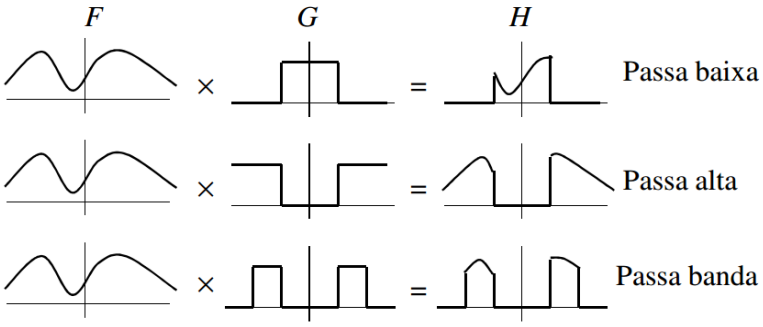

Figura 13 – Filtragem no domínio da frequência
Fonte: FECHINE, 2010

A Transformada de Fourier Unidimensional é usada em sinais biológicos, como o eletrocardiograma (ECG), que é realizado numa largura de banda menor, onde o principal interesse é medir o ritmo e desprezando os detalhes morfológicos, como pode ser observado na Figura 14.

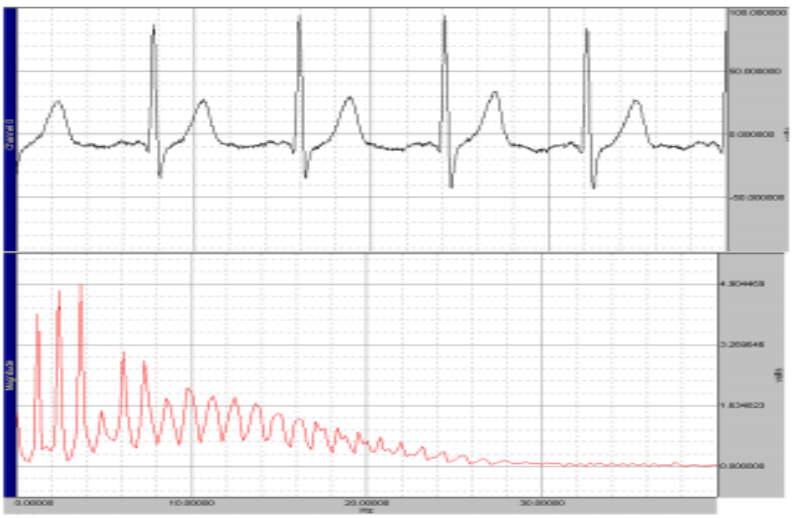

Figura 14 – ECG em largura de banda menor
Fonte: FECHINE, 2010

Já transformada de Fourier bidimensional é utilizada em imagens, onde o ponto (0, 0) denota a intensidade média da imagem. Os coeficientes de índices baixos (frequências) correspondem aos coeficientes da imagem que variam pouco (de um pixel para o pixel vizinho) e os coeficientes de alta frequência são associados a variações bruscas de intensidade. Na Figura 15 é demonstrada a aplicação da Transformada Bidimensional de Fourier em imagens, fazendo a comparação do espectro de Fourier de imagens de impressão digital, onde (a) e (b) estão sem ruídos e (c) e (d) com ruídos.

E também pode ser utilizada em processamento de imagens, com filtragem passa-baixa, passa-faixa e passa-alta, o que pode ser observado através da Figura 16.

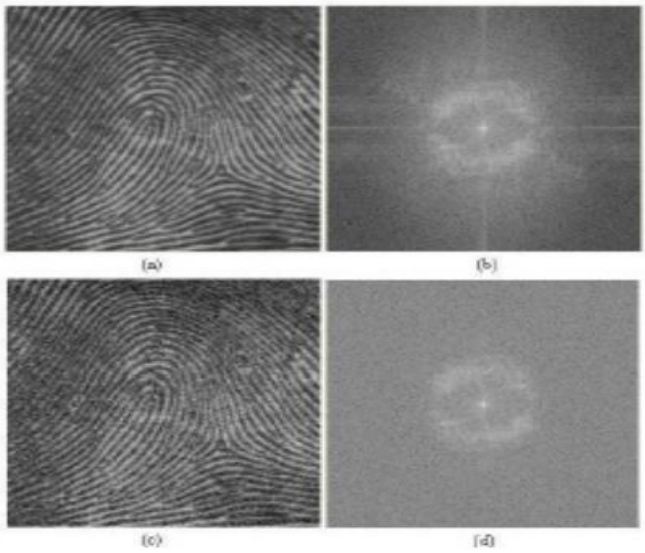

Figura 15 – Comparação do espectro de Fourier de imagens de impressão digital
Fonte: FECHINE, 2010

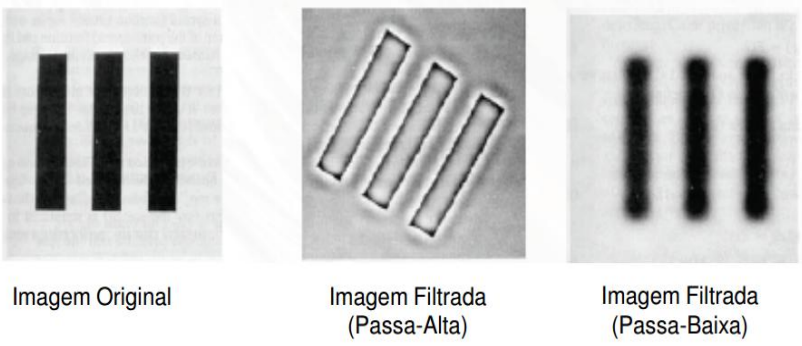

Figura 16 – Processamento de imagens com filtragem passa-alta e passa-baixa
Fonte: FECHINE, 2010

O fato de utilizar um número infinito de amostras no domínio do tempo e, consequentemente, um número infinito de pontos no domínio da frequência, representa um problema para a implementação da transformada de Fourier na prática.

3.3.3 Transformada Discreta de Fourier

A Transformada Discreta de Fourier (DFT), ou também Transformada Rápida de Fourier, FFT, do inglês *Fast Fourier Transform*, utiliza um número finito de pontos no domínio do tempo e define uma representação discreta do sinal no domínio da frequência. A FFT é um algoritmo eficiente para calcular a DFT e sua inversa, baseado no método de dobramentos sucessivos, “dividir para conquistar”.

Existem diversas implementações para a FFT, como o mapeamento em índices multidimensionais, que transforma uma DFT de uma dimensão em uma DFT de duas ou mais dimensões; o algoritmo Cooley-Turkey, que usa o mapeamento de índices de tempo e frequência (COOLEY e TURKEY, 1965); FFT por dizimição no tempo que descompõe a sequência de tamanho N em sequências sucessivas menores. E o método mais utilizado e eficiente é o FFT por dizimição em frequência que usa todas as dimensões do mesmo tamanho e computa a DFT quando o tamanho N da sequência é uma potência de 2. Sua complexidade é dada por O(n log n) contra O(n^2) para o cálculo por definição. (Calcula uma DFT de tamanho N, em termos de duas DFTs de tamanho N/2.) (NEYRA-ARAOZ, 2017).

3.4 Janelamento

Durante o processo de obtenção do espectro de frequência de um sinal quantizado, a fim de obter a DFT, a função janela, também denominada de função de ponderação, que é aplicada ao sinal pode afetá-lo por estar mal dimensionada. Isso faz surgir um vazamento espectral, fenômeno em que o espectro de frequência contenha componentes de frequência desconhecidos. Com apenas uma janela é difícil compreender os ciclos inteiros de todos os sinais que fazem parte do sinal original, fazendo com que este espalhamento ocorra constantemente na prática. Para tentar minimizar esse dano utiliza-se uma técnica denominada

janelamento, que suaviza as frequências indesejadas, resultando em redução da descontinuidade nas bordas do sinal. (SANTANA, 2016)

Existem diversas janelas, tais como, a Triangular, Exponencial, *Hanning, Hamming*, *Flatop*, *Kaiser-bessel*, *Welch*, *Blackmann*, entre outras.

A janela *Hamming* é uma variação da janela *Hanning*, porém, no domínio do tempo, a *Hanning* se aproxima do zero, diferenciando-se assim da *Hamming*. Como pode ser observado na Figura 17, sua forma é similar a uma onda de cosseno. A equação 3 define uma janela *Hamming* de tamanho N. (MACHADO, MOECKE, 2017).

$$w[n] = 0.54 - 0.46\cos\left(\frac{2\pi n}{N}\right), n = 0, 1, 2, \ldots, N-1 \qquad (3)$$

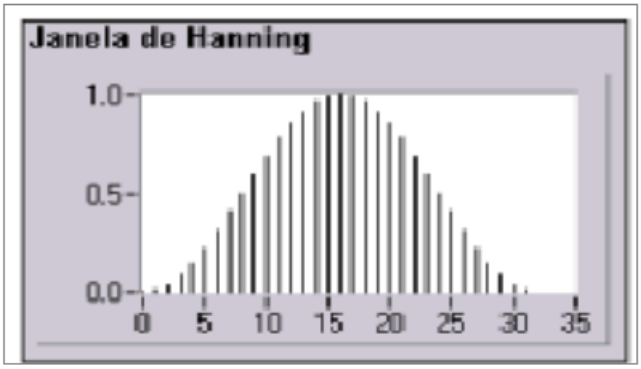

Figura 17 – Janela Hamming
Fonte: MACHADO, MOECKE, 2017

3.5 Audacity

O Audacity é um *software* gratuito para gravação e edição de áudio, possui diversas funcionalidades, como por exemplo, para renderização, mixagem, adição de efeitos e conversão para diversos formatos de áudio.

A interface do programa é simples e de fácil interpretação, garantindo ao usuário acesso rápido e fácil, como demonstrado pela Figura 18.

O software foi utilizado na realização do projeto como base para gravação de voz e plotagem do espectro de frequência do sinal coletado, utilizados para a realização de testes.

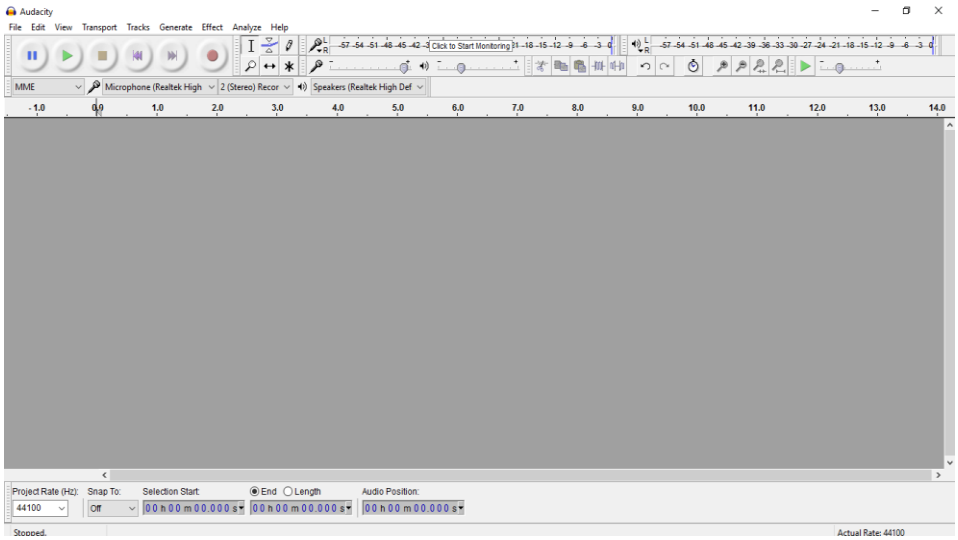

Figura 18 – Interface Audacity
Fonte: Própria

3.6 Reconhecimento de voz

O objetivo da área de reconhecimento de voz é desenvolver máquinas com a mesma capacidade de entender a linguagem falada do ser humano. O sistema de reconhecimento de voz se distingue em três tipos básicos: o de palavras isoladas, que procura reconhecer uma dentre um grupo de palavras pré-estabelecidas; o de palavras conectadas, que identifica uma frase falada baseado em modelos individuais e, por último, o tipo de fala contínua, em que as palavras são faladas de continuamente e reconhecidas com base em modelos de subpalavras.

> O reconhecimento de voz é essencialmente um problema de classificação de padrões realizada a partir de uma seqüência de parâmetros, ou atributos, que caracterizam o sinal de voz. Tipicamente, a forma de onda é dividida em intervalos de tempo que se sobrepõem, e para cada intervalo calcula-se um conjunto de parâmetros que caracterizam o aparelho vocal, obtidos com base em um modelo de produção de voz (ALCAIM e OLIVEIRA, 2011, 22).

O sistema de reconhecimento de voz tem sua criação os anos 1990, através do *Automatic Answer Network System for Eletrical Requests* (ANSER), desenvolvido pela companhia telefônica japonesa NTT. A aplicação permitia o diálogo simples entre o ser humano e um computador, tecnologia que hoje é utilizada em telefones celulares e ferramentas da internet (ALCAIM e OLIVEIRA, 2011). O processo básico do sistema de reconhecimento de voz é composto pelo microfone, conversão do áudio capturado de analógico para digital, extração

de parâmetros característicos, classificação de padrões – de acordo com referências pré-estabelecidas e, por fim, identificação da fala.

3.7 Auxílio a deficientes auditivos

O aprendizado da fala está relacionado com a produção de movimentos do aparelho vocal a fim de reproduzir um som baseado no que o indivíduo está ouvindo. Pessoas com deficiência auditiva são impedidas de desenvolver a fala por conta deste motivo. Técnicas que envolvem o sistema de reconhecimento de voz permitem que indivíduos surdos consigam desenvolver sua fala através de dispositivos visuais. A Figura 19 elucida a implementação destes métodos.

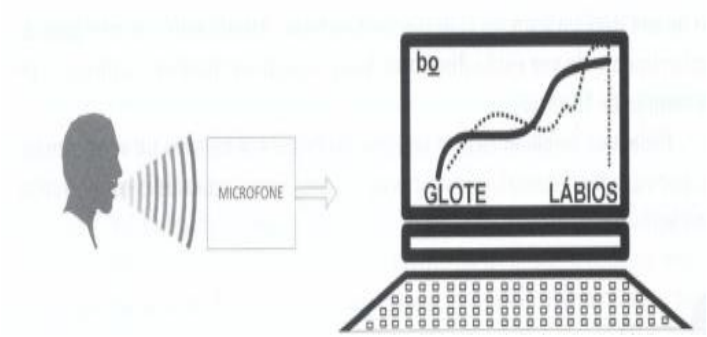

Figura 19 – Ilustração de um esquema de auxílio a deficientes utilizando processamento de voz
Fonte: ALCAIM, OLIVEIRA, 2011

Os indicativos produzidos pelos sons do indivíduo formam uma curva, que é utilizada como referência da forma adequada de colocar o aparelho vocal, ao serem colocadas em comparação com os padrões pré-estabelecidos. Este treinamento faz com que pessoas com deficiência consigam ter uma representação da forma correta e tentem produzir um som que esteja o mais próximo possível da curva pré-estabelecida.

3.8 Melhoria da qualidade do sinal de voz

A qualidade do sinal de voz degradado é um objetivo bastante almejado, distorções advindas de ruídos e ecos são alvos de técnicas de processamento digital “desenvolvidas para melhorar qualidade de voz degradada pela distorção não linear sofrida pelas ressonâncias do aparelho vocal”. Um exemplo a ser citado é a utilização da transformada *Fourier* em intervalo curto, permitindo o aumento da inteligibilidade de 40 para 70% (ALCAIM e OLIVEIRA, 2011).

3.9 O sinal de voz

Alcaim e Oliveira (2011) atribuem os sons da fala ao resultado de movimentos voluntários do aparelho respiratório e mastigatório, que são propagados através de uma onda de pressão acústica. A Figura 20 demonstra a estrutura envolvida no processo de fala.

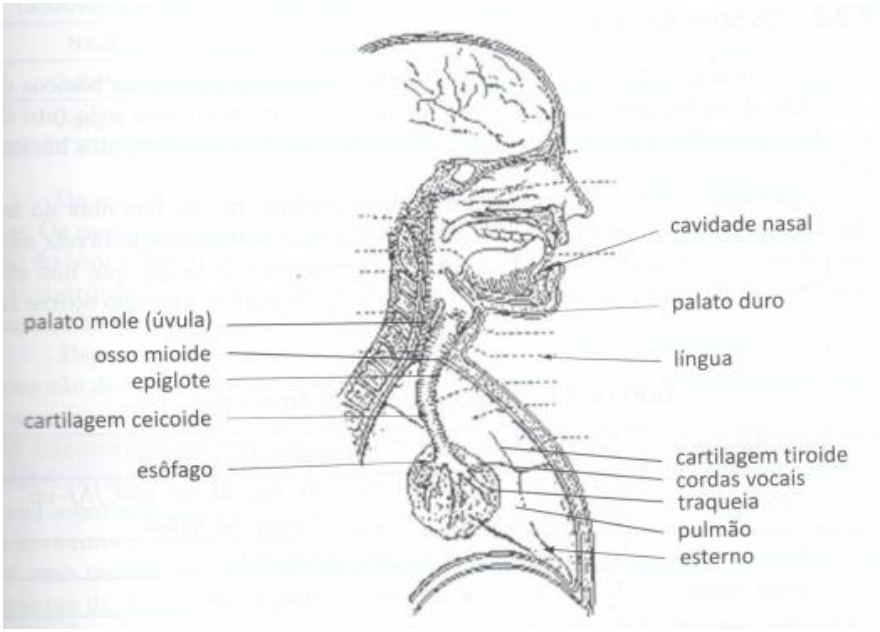

Figura 20 – Mecanismo vocal humano
Fonte: ALCAIM, OLIVEIRA, 2011

O aparelho vocal é determinado pela posição dos lábios, da mandíbula, da língua e da úvula, e pode variar de acordo como o tempo.

> O acompanhamento acústico entre os aparelhos vocal e nasal é controlado pela abertura da úvula, de forma que durante a produção de sons não nasais a cavidade nasal e obstruída pela úvula. A área de abertura controlada pela úvula varia de zero a aproximadamente 5 cm² (ALCAIM e OLIVEIRA, 2011, 24-25).

O processo de fala envolve músculos localizados no tórax e abdômen, que exercem pressão nos pulmões através de contrações do tórax e expele o ar pela traquéia em direção à faringe.

Existem três mecanismos básicos de excitação envolvidos nos sons gerados pela fala. Os “sons sonoros”, como o som /u/ de uva; os “sons fricativos surdos”, como o /s/ em sala e os “sons oclusivos”, como pelo /p/ em pato (ALCAIM e OLIVEIRA, 2011).

Os fonemas são elementos básicos da linguagem humana, eles se caracterizam por diferenciarem duas palavras apelas por um de seus elementos básicos. As Tabelas 1 e 2 representam os fonemas da língua inglesa e portuguesa, em que é possível perceber essa diferenciação.

Tipos de som	Exemplos
Vogais	/i /- eve, /ɪ/- it, /ɛ/- met, /e/- hate, /æ/- at, /û/- bird, /ʌ/- up, /u/- boot, /U/- foot, /ô/- all, /o/- obey, /a/-father
Fricativas Sonoras	/v/- vote, /ð/- then, /z/ zoo, /ʒ/ azure
Fricativas Surdas	/f/ - for, /θ/- thin, /s/- see, /ʃ/- she, /h/- he
Oclusivas Sonoras	/b/- be, /d/- day, /g/- go
Oclusivas Surdas	/p/- pay, /t/- to, /k/- key
Nasais	/m/- me, /n/- no, /ŋ/- sing
Glides	/w/- we, /ɪ/- you
Semivogais	/r/- read, /l/- let
Ditongos	/aI/- I, /ôI/- boy, /aU/- cut, /eI/- say, /oU/- go, /Iu/- new
Africadas	/tʃ/- chew, /dʒ/- jar

Tabela 1 – Fonemas do Inglês Americano
Fonte: ALCAIM, OLIVEIRA, 2011

Tipos de som	Exemplos
Vogais	/i /- vi, /ɛ/- vela, /e/- vê, /a/- vala, /u/- uva, /ô/- bola, /o/- bobo
Fricativas Sonoras	/v/- chuva, /z/- zelo, /ž/- gelo
Fricativas Surdas	/f/ - faca, /s/- sala, /š/- chuva
Oclusivas Sonoras	/b/- bato, /d/- dedo, /g/- gola
Oclusivas Surdas	/p/- pato, /t/- tatu, /k/- capa
Nasais	/m/- mala, /n/- nada, /ñ/- manha
Laterais	/l/- cala, /l̃/- calha
Vibrantes	/r/- cara, /R̃/- carro

Tabela 2 – Fonemas do Português brasileiro
Fonte: ALCAIM, OLIVEIRA, 2011

Os sons se classificam em continuados, representados pelas vogais, fricativas e nasais, e em não continuados, que é o caso dos sons oclusivos. De acordo com Alcaim e Oliveira (2011), os sons também se dividem em sonoros e surdos, de acordo com a presença, ou não, de vibração das cordas vocais.

> Observa-se que, para os sons sonoros, a forma de onda é aproximadamente periódica. O intervalo T_0 entre os picos principais fornece uma medida do período fundamental para aquele locutor em particular. O inverso de T_0 corresponde à freqüência fundamental F_0, que pode apresentar variações de uma oitava no decorrer de uma sentença falada por uma mesma pessoa (ALCAIM e OLIVEIRA, 2011, 27).

Na Figura 21, analisando a forma de onda dos sons sonoros pode-se perceber oscilações amortecidas, derivadas das ressonâncias espectrais, também denominadas formantes, da cavidade vocal, refletindo singulares aspectos do aparelho vocal humano. Na forma de onda dos sons surdos já é possível notar a presença de ruídos.

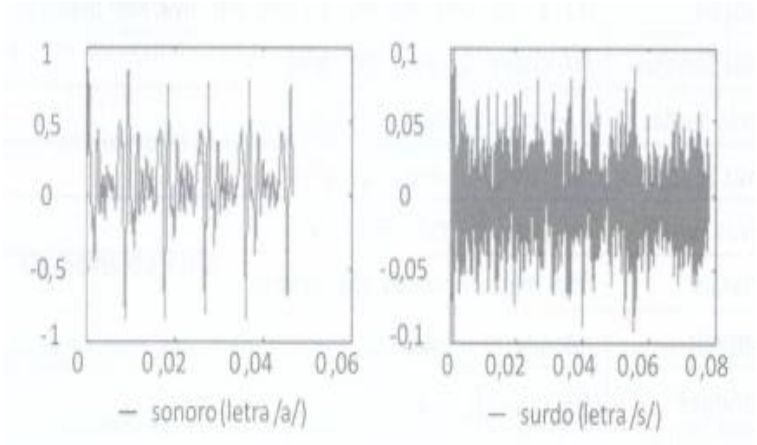

Figura 21 – Formas de onda de sons sonoro e surdo
Fonte: ALCAIM, OLIVEIRA, 2011

Os sons da voz podem ser classificados em vogais: sons sonoros, provenientes de vibrações das cordas vocais e geralmente longos, nasais: sons sonoros caracterizados pela saída de ar pela boca e pelo nariz, na língua portuguesa são os fonemas /m/, /n/, e /nh/, em seu espectro a primeira frequência possui baixa ressonância, um zero espectral e características uniformes para frequências altas, fricativos: sonoros como em /v/ ou surdos como em /f/, variando conforme a presença ou ausência de vibração nas cordas vocais, caracterizados por ruído contínuo, e oclusivos: sonoros como em /b/ ou surdos como em /p/, possuem excesso de pressão em um ponto do aparelho vocal, logo após há um desprendimento espontâneo de ar.

3.9.1 Modelo de produção da fala

Foi desenvolvido um modelo de produção da fala baseado nas características do mecanismo vocal humano, sendo a fonte de excitação e o filtro do aparelho vocal considerados como

distintos sistemas. Os sistemas de filtragem do aparelho e algumas fontes de excitação produzem como resposta o sinal de voz s(t), o qual tem suas características geradas ao longo do tempo através das características da fonte e do filtro.

Na Figura 22 está ilustrado o diagrama de blocos desse sistema, já discretizados no tempo.

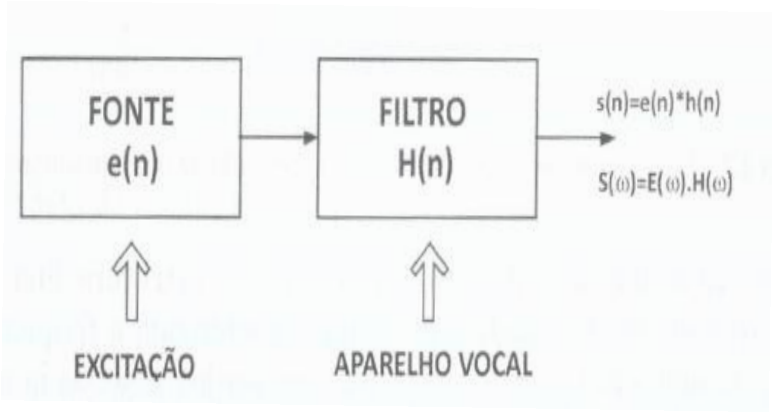

Figura 22 – Diagrama de blocos do modelo de produção da fala
Fonte: ALCAIM, OLIVEIRA, 2011

Sendo o espectro de amplitude, em dB, dado por:

$$|S(\omega)| = 20\log_{10}(|E(\omega)|.|H(\omega)| = |E(\omega)| + |H(\omega)|$$

São verificadas a existência de duas componentes, sendo uma estrutura fina relacionada a $|E(\omega)|$, com uma envoltória espectral suave, relacionada a $|H(\omega)|$, que podem ser observadas na Figura 23.

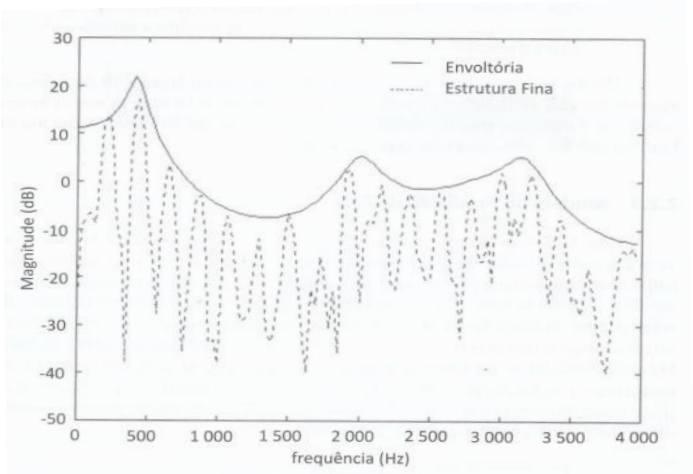

Figura 23 – Espectro de amplitude em intervalo curto (dB) de um som sonoro da fala
Fonte: ALCAIM, OLIVEIRA, 2011

A estrutura fina, componente da excitação, possui picos com distâncias similares, que caracterizam a frequência fundamental deste som sonoro.

Para sinais de fala o ouvido humano é praticamente insensível a variações da forma de onda geradas por distorções de fase. Porém essas podem ser audíveis como reverberação do sinal quando correspondem a retardos maiores de 50ms e alteram o espectro de amplitude em curto intervalo.

4. DESENVOLVIMENTO

Durante este capítulo será apresentado o desenvolvimento do projeto, desde a análise dos sinais de voz até o acionamento realizado, descrevendo inclusive o circuito utilizado para tal, assim como o código implementado.

4.1 Esquemático da solução implementada

Foi feito o acionamento de um LED para representar o acionamento de possíveis dispositivos ligados ao sistema, ao se ter uma palavra decodificada. A decodificação pode ser realizada *in loco* por meio de um microfone ou por meio de uma ligação telefônica caso o sistema esteja adaptado a um celular receptor para acionamento remoto.

O software Audacity foi utilizado na realização de testes, simulando o uso de um microfone acoplado ao sistema, e num segundo teste, a saída de som do celular. No caso do celular, o som também foi gravado usando um aplicativo qualquer de gravação de voz, somente para testes, já que o objetivo é que a voz seja passada para o sistema pela saída de som do celular durante uma ligação.

O som é transmitido para o circuito do Arduino, a FFT devolve o espectro de frequência melhorado, utilizando a janela *hamming*, e então o algoritmo identifica a frequência dos principais picos do espectro, decodificando assim a palavra emitida. Esse procedimento é realizado tanto na etapa de gravação das frequências característica de uma palavra quanto na etapa de reconhecimento da palavra para o acionamento de uma carga (um LED, neste trabalho).

Para o desenvolvimento do sistema, mensagens enviadas para a Serial foram utilizadas a todo o momento, bem como alguns comandos básicos dados como entrada nos pinos 8 a 12. Cabe lembrar que na utilização do sistema, apenas os LEDs servirão de apoio para guiar a gravação e uso do sistema e a função de algumas das entradas nos pinos citados. Dessa forma, o não uso da comunicação Serial diminui o processamento a ser realizado pelo Arduino bem como aumenta a sua velocidade.

O circuito utilizado no processo está esquematizado na Figura 24, na qual o microfone pode ser substituído pelo uso do software Audacity e do celular para recepção do sinal sonoro.

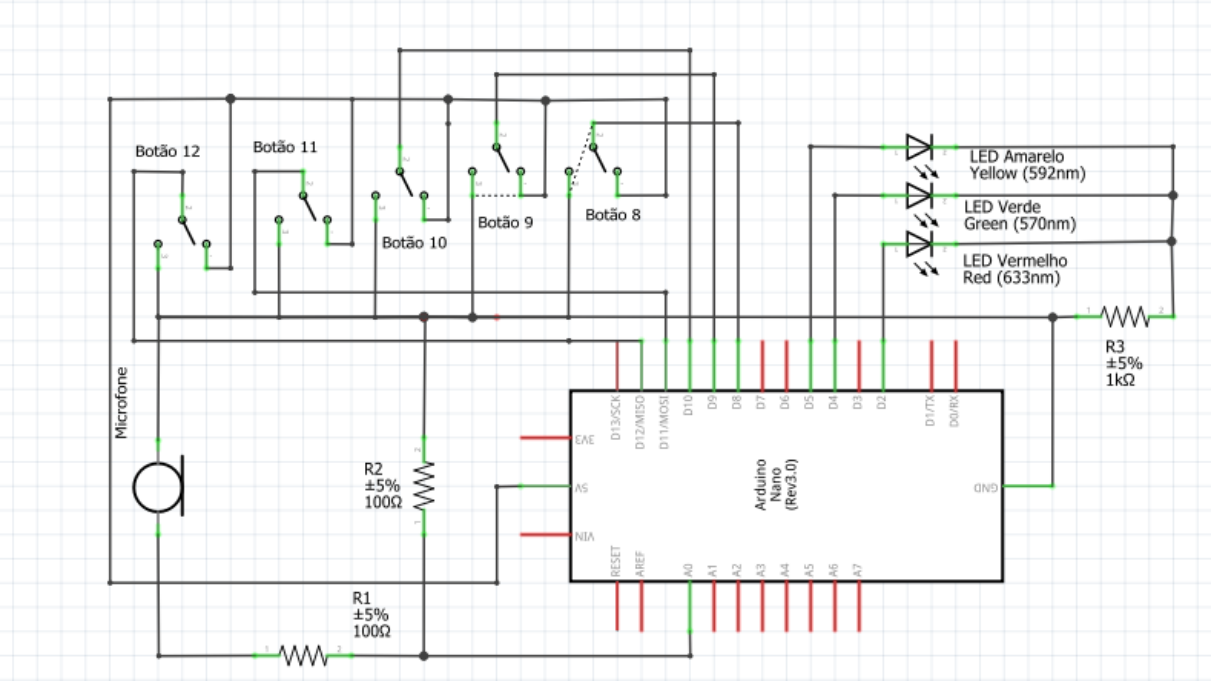

Figura 24 – Esquemático do circuito utilizado
Fonte: Própria

4.2 Análise de sinais

Para análise dos sinais foi utilizado o software Audacity, onde foram gravadas as palavras que futuramente seriam utilizadas na decodificação. A palavra, depois de gravada, é selecionada e é emitido, através de uma funcionalidade do software, o espectro de frequência desse sinal, que representa a sua FFT. Os testes foram realizados com as palavras “direita”, “esquerda”, “abra”, “alto”, “baixo” e “para”.

Com esse gráfico do espectro de frequência do sinal pode-se observar os valores dos picos e, com isso, comparar com os valores obtidos no Arduino quando o som é reproduzido e decodificado, a fim de se ter uma ideia do funcionamento correto do sistema no microcontrolador.

Para identificar as frequências para uma determinada palavra pronunciada por uma determinada pessoa, deve-se identificar as frequências típicas que ocorrem durante sua reprodução, criando um padrão. Para isso foram realizadas três gravações com a mesma palavra no software Audacity e geradas lado a lado suas respectivas FFT’s e seus sinais gerados, utilizando 128 amostras e a janela Hamming.

4.3 Implementação da Transformada de Fourier

Para a implementação da Transformada de Fourier, foi utilizado como base o código de Didier Longueville (2010), modificado por Ted Hayes em 2011, que implementa um algoritmo da Transformada Rápida de Fourier (Fast Fourier Transform – FFT) para a plataforma Arduino.

4.3.1 Plain FFT

O algoritmo utilizado como base para implementação do sistema decodificador através da FFT, denominado PlainFFT, possui em sua biblioteca .cpp, funções específicas para essa filtragem de sinais (HAYES, 2010 apud LONGUEVILLE, 2011). A função “windowing” realiza o janelamento do sinal, a “compute” avalia o vetor janelado e emite a FFT do sinal, a função “complexToMagnitude” retorna as magnitudes da FFT e a função “majorPeak” identifica o maior pico de magnitude.

No código implementado não é utilizada a função majorPeak, pois o objetivo é identificar diversos picos como referência e não apenas 1, logo, essa aquisição foi realizada através de uma rotina explicada mais a frente.

4.3.2 Algoritmo

Foi criado, primeiramente, um código teste para analisar os valores de magnitudes que poderiam ser usados para limite. O código está programado para o reconhecimento de apenas uma palavra, que depois de reconhecida, faz o acionamento de um LED.

O algoritmo está divido em duas etapas, como ilustrado na Figura 25, na primeira etapa o foco é reconhecer as frequências, dadas pelos índices das magnitudes capturadas (picos), da palavra pronunciada, sendo ela repetida e analisada diversas vezes, a fim de atenuar erros. Na segunda etapa, as frequências reconhecidas como padrão são gravadas em um vetor e, em seguida, são comparadas com uma nova entrada de frequências, obtidas através de uma nova pronúncia da palavra. Logo após, se a decodificação reconhecer a palavra pronunciada, então é realizado o acionamento.

Mais detalhadamente, a primeira etapa é dividida em duas seções, sendo a primeira responsável pela identificação dos picos, através do limite estabelecido para as magnitudes, e seu armazenamento num vetor que acumula 7 frequências. Já a segunda seção verifica se os picos armazenados durante a primeira seção estão se repetindo, quando se repete a palavra. O vetor encarregado da contenção das frequências após essa verificação é denominado medPicos. Esta verificação, é realizada 5 vezes, e se faz necessária para evitar a interpretação de ruídos como picos no sinal. Para a detecção de picos de magnitude foi criada uma rotina que os identifica através de um valor limite definido. O limite definido para captura dos picos depende do aparelho de som, quanto maior for o volume maior deverá ser o limite definido. Como no caso desse projeto o volume é variável (o sinal pode vir de um computador, celular ou microfone), então foi criada uma rotina para contornar essa divergência, que ajusta o limite assim que necessário. Na prática, com o uso de um microfone externo, com volume fixo, essa questão do volume seria contornada bem como o uso de um celular com volume fixo.

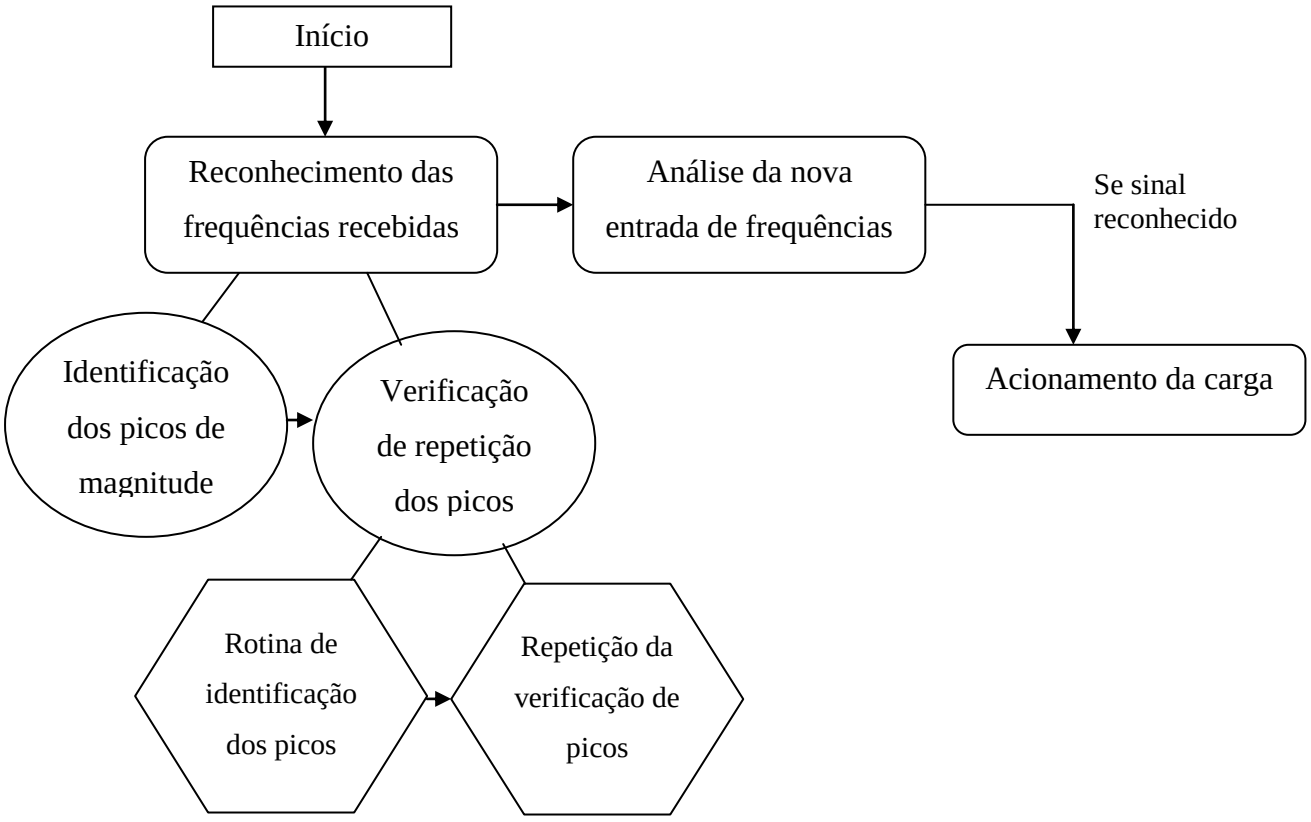

Figura 25 – Esquema do algoritmo utilizado
Fonte: Própria

A frequência de amostragem escolhida foi a de 8000Hz, pois é a frequência normalmente utilizada para áudios. Segundo Silva Melo (2008, p.115 e 116), o ser humano emite a voz

num intervalo de frequências entre 80Hz e 12kHz, porém, comprovadamente têm-se que o receptor do som reconhece as palavras claramente se os sinais de voz forem emitidos numa de frequência entre 300Hz e 3400Hz (apud ITU, 2007). Para o sistema de telefonia, os EUA adotaram a faixa de frequência entre 200Hz a 3200Hz, sendo que a maior energia do sinal está retido nessa faixa.

O teorema da amostragem, ou Teorema de Nyquist, estabelece que um sinal analógico transmitido por uma largura de banda B Hz, será reconstruído pelo receptor, após a filtragem, com uma frequência de no mínimo 2B vezes por segundo. Essa frequência é reconhecida como Frequência de Nyquist, ou frequência de amostragem. (FELIPPETO DE CASTRO, 2017). Portanto, isso prova que a frequência escolhida de 8000Hz é suficiente para essa decodificação, já que, mesmo considerando um máximo de 4000Hz para a frequência de voz, 8000Hz estaria dentro do valor aceitável.

O código é iniciado com a chamada da biblioteca, cria-se então o objeto FFT, e é definida a quantidade de amostras e a frequência de amostragem além de serem criadas algumas variáveis. O ‘void setup’ inicializa a serial, que será utilizada apenas para auxílio de visualização das frequências na verificação nos testes, ou seja, o sistema não precisará dela. A serial inicializa pinos digitais para serem usados, sendo leds nos pinos 2, 4 e 5 e os outros pinos utilizados para comandos. O uso da serial consome tempo de processamento do Arduino, portanto seu uso está condicionado estritamente à necessidade de uso, contralada pelo ‘void loop’.

Um vetor armazena até 128 amostras do sinal sonoro que entra pela entrada analógica do Arduino. Sobre esse vetor, é feito o janelamento do sinal por meio da função "windowing" chamada por meio do objeto criado para a biblioteca, ficando então, "FFT.windowing". Os valores armazendos no vetor ‘vReal’ são subescritos durante todo o código. A função “compute”, analisa o vetor já com os valores do sinal janelado, atribuindo a ele o valor da FFT e a função “complexToMagnitude” retorna as magnitudes da FFT. Então é realizada a rotina para o controle do limite de captação dos picos e para a identificação das frequências associadas.

O código completo, onde é possível analisar as funções e rotinas descritas, se encontra no Apêndice A.

5. RESULTADOS

Para demonstração dos testes, realizados no Audacity, foram feitas gravações de cada palavra duas vezes e, em seguida, foi plotado seus respectivos espectros de frequência (FFT) utilizando as ferramentas do próprio Audacity, como demonstrado nas Figuras 26, 27, 28, 29, 30 e 31. Pode ser observado que seus picos de magnitudes ocorrem aproximadamente nas mesmas frequências (eixo das abcissas), podendo assim a palavra ser reconhecida pela decodificação.

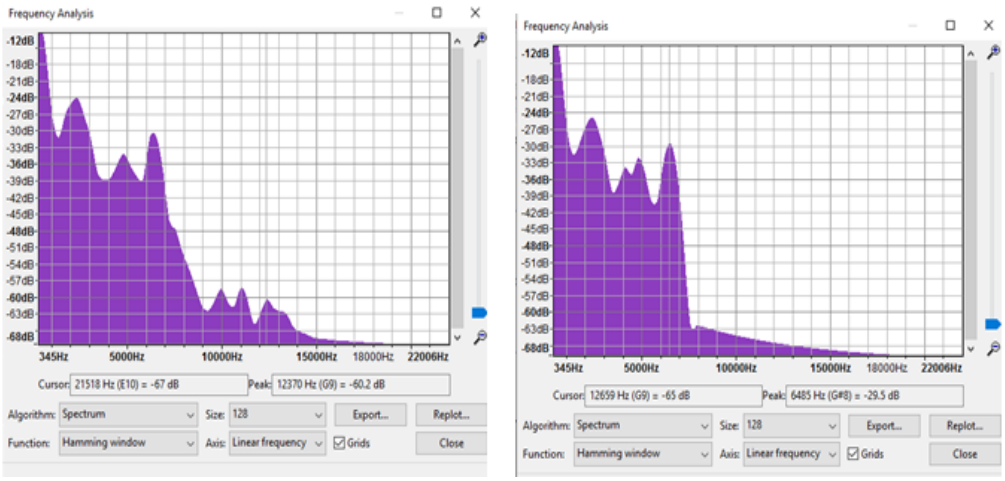

Figura 26 – Espectros de frequência da palavra “direita”
Fonte: Própria

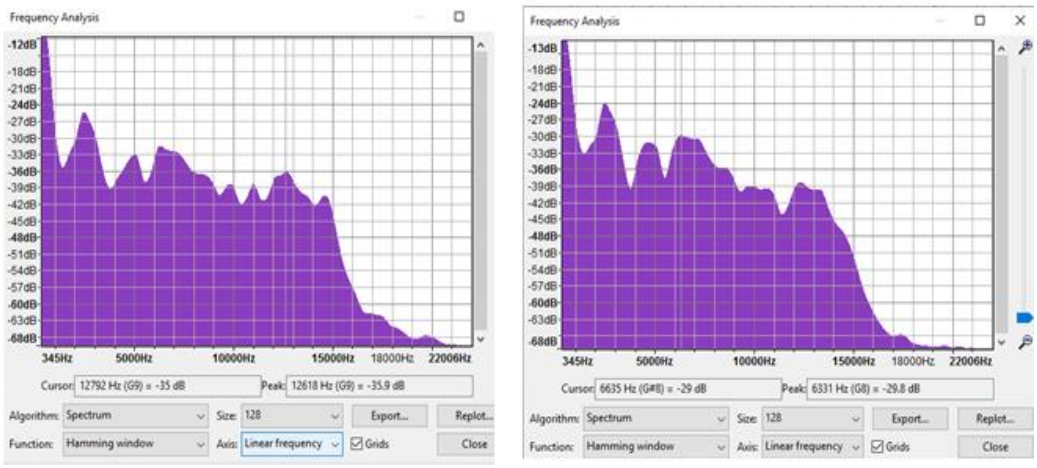

Figura 27 – Espectros de frequência da palavra “esquerda”
Fonte: Própria

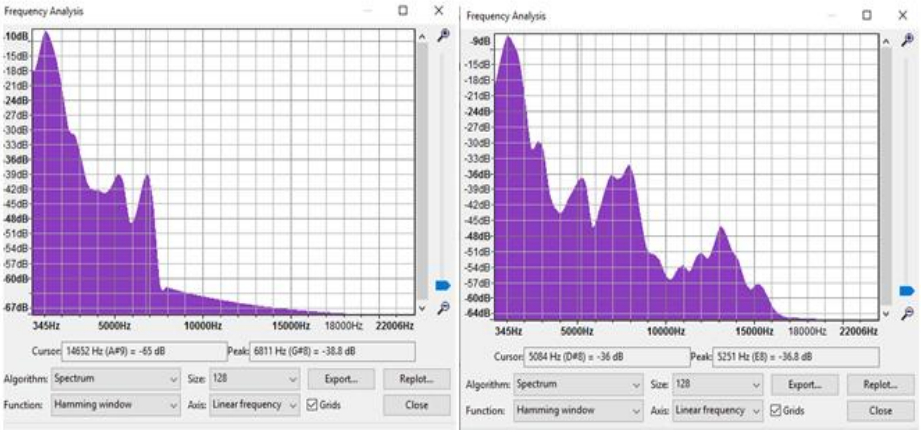

Figura 28 – Espectros de frequência da palavra “abra”
Fonte: Própria

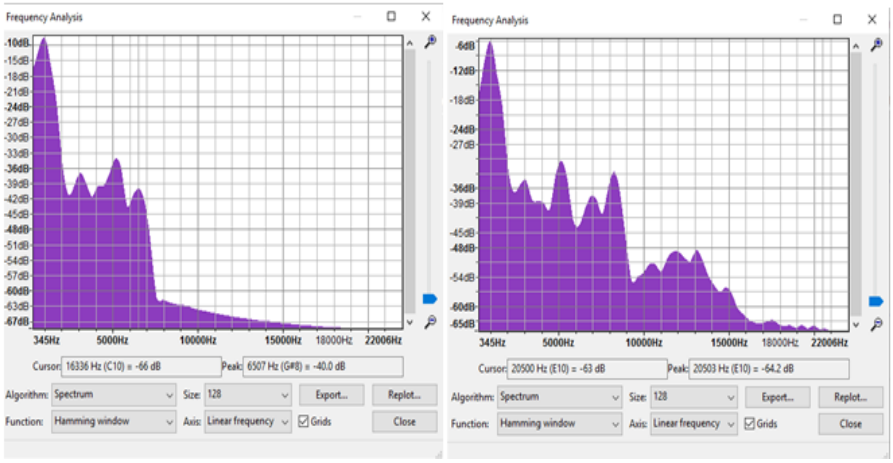

Figura 29 – Espectros de frequência da palavra “alto”
Fonte: Própria

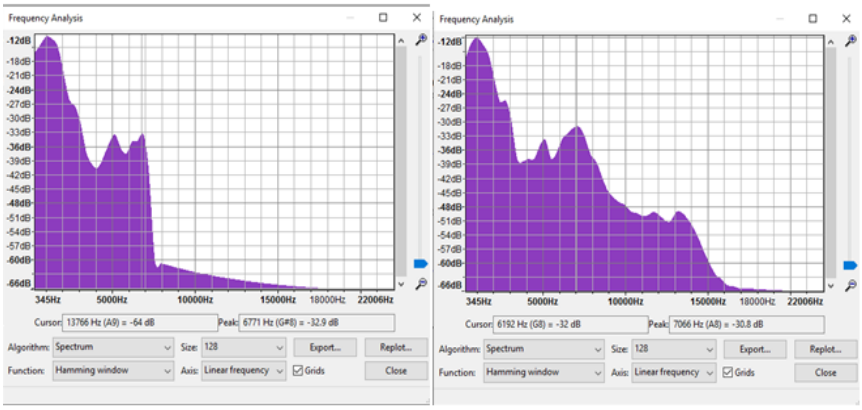

Figura 30 – Espectros de frequência da palavra “baixo”
Fonte: Própria

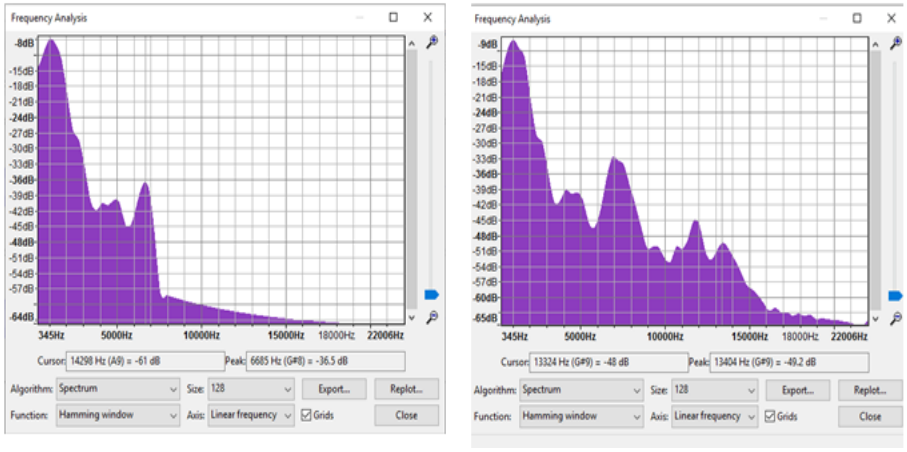

Figura 31 – Espectros de frequência da palavra “para”
Fonte: Própria

Também foram realizados testes com palavras diferentes, mas que possuem as mesmas componentes vocálicas, no caso, foram utilizadas as palavras “abra” e “para”, sendo os “a’s” as componentes vocálicas em comum. Como pode ser observado nas Figuras 28 e 31, os espectros de frequências das palavras “abra” e “para”, respectivamente, se assemelham. Visualmente percebe-se uma aproximação nos valores das magnitudes, o que poderia causar uma confusão no sistema e uma palavra ser interpretada como a outra, dependendo do número de frequências analisadas. No caso deste projeto a condição para que uma palavra seja reconhecida é que, no mínimo, 5 frequências se repitam na etapa de gravação, considerando que 7 frequências estão sendo armazenadas e comparadas.

Cada pessoa tem uma frequência de voz diferente, por isso o reconhecimento de voz é ideal para utilização em segurança, já que assim o sistema só irá reconhecer a voz do usuário.

Também para a exposição dos resultados, foram utilizadas as palavras gravadas no Audacity, utilizando o microfone do computador, com a entrada do sinal na entrada analógica do Arduino através do plug P2.

Os LED’s são utilizados no circuito para sinalização, podendo ser dispensado o uso da serial. Para todas as palavras, individualmente, a gravação é repetida algumas vezes para análise do espectro de frequência, enquanto o LED vermelho está aceso, como observado na Figura 32. Para auxílio na análise do sinal é utilizada a Serial do Arduino, durante os testes, de forma que pelo monitor serial, pôde-se acompanhar as frequências obtidas a cada pronúncia da palavra na etapa de gravação, bem como quantas vezes uma mesma frequência se repetiu. É esperado, pelo menos, 5 ocorrências de cada frequência para sair da etapa de gravação, caracterizando a

condição de saída. Uma sexta frequência precisa se repetir pelo menos uma vez também. O ideal seria que todas as frequências se repetissem em todas as reproduções da palavra, porém a condição dada é menor, devido a presença de ruído e à limitação do tamanho da amostra em apenas 128 amostras, pela biblioteca que executa a FFT. Ainda na etapa de gravação, em um vetor denominado ‘medPico’, é feita a soma cumulativa das frequências semelhantes que se repetem (cada frequência somada em uma posição do vetor) e em seguida, quando a condição de saída é satisfeita, cada soma é dividida pela quantidade de vezes em que esta frequência se repetiu, obtendo-se como resposta a média das frequências.

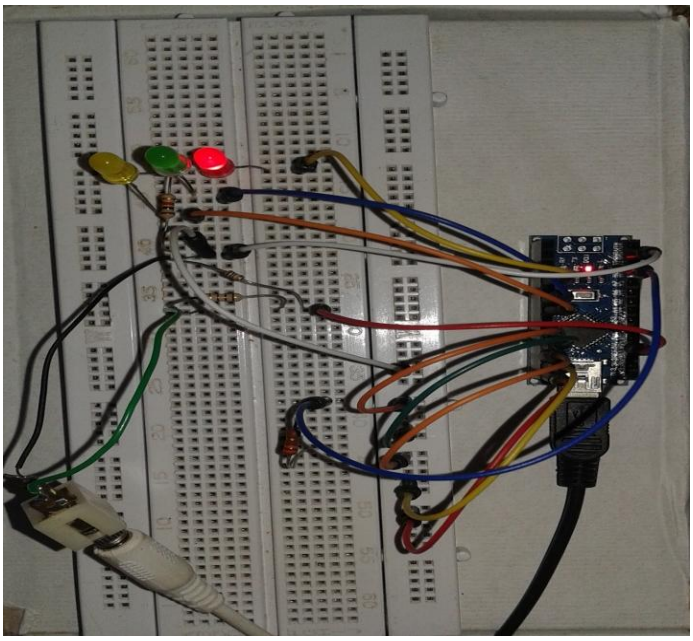

Figura 32 – LED vermelho acionado para sinalização de reprodução da palavra
Fonte: Própria

Neste ponto têm-se duas opções, jogar nível alto no pino 9 para gravar uma nova palavra ou jogar nível alto no pino 8 para executar a palavra gravada e acionar o LED amarelo, que é a carga. Nesse momento o LED verde está aceso, para demonstrar que a gravação foi um sucesso e que as frequências já foram obtidas, como ilustrado na Figura 33.

Quando é dado nível alto no pino 8 para poder pronunciar a palavra, o LED verde vai apagar e o vermelho vai começar a piscar de novo, enquanto ele estiver aceso é pronunciada ou reproduzida a palavra, sendo comparada as frequências obtidas nessa reprodução com as frequências gravadas para a palavra. É necessário o reconhecimento de, ao menos cinco, para

o acionamento da carga, com o acionamento do LED amarelo, como demonstrada na Figura 34. A partir deste ponto é possível escolher opções para gravar outra palavra, com o pino 11 em nível alto, ou desativar a carga e reproduzir novamente a palavra em análise, com o pino 10. A todo o momento é dada a opção de gravar uma nova palavra colocando-se o pino 12 em nível alto.

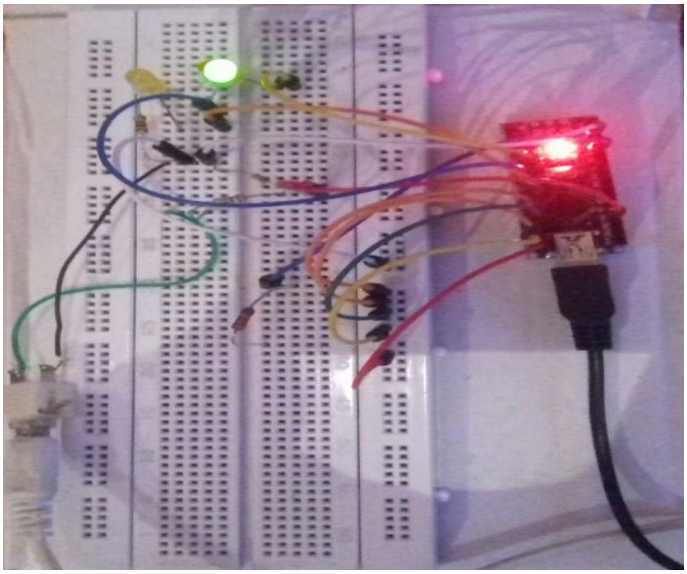

Figura 33 – LED verde acionado para sinalização de gravação realizada
Fonte: Própria

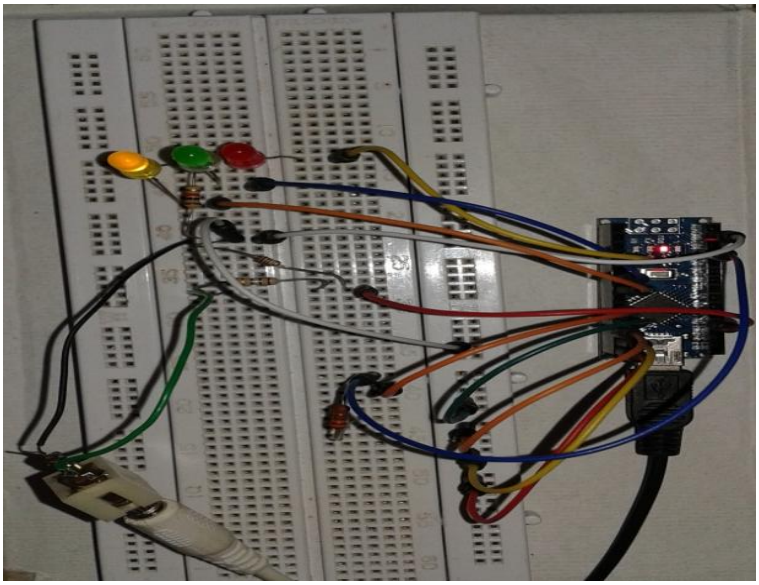

Figura 34 – LED amarelo acionado para sinalização de reconhecimento da palavra
Fonte: Própria

Os testes realizados com as palavras “direita”, “esquerda”, “abra”, “alto”, “baixo” e “para”, revelaram o número necessário de repetições de cada palavra para sua gravação e o número de repetições necessárias para realizar o acionamento da carga, após a gravação da palavra, como pode ser obervado na Tabela 3.

Palavra	Repetições para gravação	Repetições para acionamento
Direita	7	3
Esquerda	8	1
Abra	5	1
Alto	6	4
Baixo	5	1
Para	6	1

Tabela 3 – Repetições para gravação e acionamento da carga
Fonte: Própria

Outro teste foi realizado com palavras que possuem mesmas componentes vocálicas, “abra” e “para”, nesse caso a letra “a”, já que palavras que possuem as mesmas componentes vocálicas podem ser confundidas pelo sistema. Inicialmente, foram comparados, seus respectivos espectros de frequência através do Audacity e verificou-se que suas frequências eram próximas, podendo haver engano pelo sistema. A palavra “abra” foi gravada para o acionamento da carga, enquanto a palavra “para”, com as mesmas componentes vocálicas, foi reproduzida após a gravação da palavra “abra”, para verificar se o sistema consegue identificá-la como sendo a palavra correta para o acionamento da carga.

Foi então observado que algumas frequências não foram equivalentes às da palavra gravada para reconhecimento, e, portanto, o sistema não a reconheceu como a palavra de acionamento. Em seguida, foram feitas várias repetições da gravação da palavra “para” e, após algumas repetições, o sistema identificou frequências similares e a identificou como a palavra correta para o acionamento da carga, sendo esta acionada. Apesar de o ideal ser a palavra ser

reconhecida com apenas uma reprodução, o sistema, com algumas repetições, reconheceu uma palavra diferente como sendo a palavra gravada, o que ocasiona um erro do sistema. Tal erro deve-se à presença de mesmas componentes vocálicas nas duas palavras testadas.

6. CONCLUSÕES E TRABALHOS FUTUROS

Com o intuito de fazer o reconhecimento de voz para o acionamento de dispositivos foi proposto um sistema utilizando a Transformada Rápida de Fourier com um microcontrolador. A solução proposta consiste na decodificação de palavras através da FFT, reconhecendo os picos em frequências característicos de determinada palavra.

Foi verificado que palavras que possuem as mesmas componentes vocálicas podem ser confundidas pelo sistema. Para solucionar essa questão foi aumentada a quantidade de frequências a serem utilizadas na decodificação, porém isso aumenta o tempo de gravação devido à quantidade maior de frequências repetidas semelhantes que precisam ocorrer.

Outra forma de melhorar a precisão do sistema é aumentando sua resolução em frequência com o aumento do tamanho da quantidade de amostras, dos atuais 128 para 256 ou mais. Para a implementação dessa solução será necessário fazer alterações em toda a biblioteca que executa a FFT.

Foram observadas algumas frequências negativas durante a identificação no Arduino, que podem ser atribuídas à variável “delta” presente no código, que é utilizada como fator de correção para a frequência identificada pelo algoritmo. Uma vez que o fator de correção é necessário, a solução adotada foi desconsiderar essas frequências e trabalhar apenas com valores positivos.

Para a gravação da palavra a ser utilizada para acionamento da carga, foram necessárias repetições para um balanceamento e redução de erros na identificação das frequências características. Para algumas palavras o número de repetições foi maior que em outras, provavelmente por possuírem um maior número de frequências características dentro do limite de magnitude estabelecido.

Foi utilizado um total de 30 varreduras no vetor de amostras a cada reprodução da palavra gravada, o que está ligado diretamente à precisão do sistema. Outra forma de melhorar a precisão do sistema na identificação de palavras é a diminuição da margem de erro aceitável nas comparações das frequências semelhantes que atualmente se encontra em 30 Hz. Medidas como o aumento no número de correspondências utilizadas como condição de parada na etapa de gravação e o aumento do número de correspondências de frequências na etapa de identificação da voz no acionamento da carga, também aumentam a precisão na identificação,

mas, com o atual sistema, acabam por aumentar muito a quantidade de vezes que a palavra precisa ser pronunciada.

Para a continuação desse trabalho, sugere-se aperfeiçoar a precisão do sistema para uma identificação mais direta e imediata, sem a necessidade de muitas repetições da palavra. Aumento da quantidade de palavras identificadas, trabalhando com matrizes no código, para armazenar e manipular as frequências identificadas. Testes com palavras pronunciadas por mais de uma pessoa, a fim de verificar se o sistema distingue os comandos. Uso de um cartão de memória para armazenar as frequências e colocá-las em uso para decodificação assim que o sistema seja ligado, sem a necessidade de regravar as palavras sempre que o sistema é desligado. Sugere-se também aprimoramento do sistema para utilização remota, como por exemplo, a emissão de tons que auxiliem o usuário na realização de uma gravação remota ou identificação da ocorrência correta do acionamento de uma carga.

REFERÊNCIAS BIBLIOGRÁFICAS

AMORIM F. C.; ALVES, F. S.; LOPES, M. R. Séries de Fourier. Disponível em: < https://metodosmatematicosuff.files.wordpress.com/2011/03/sc3a9ries-de-fourier_trabalho.pdf >. Acesso em: 06 mar.2017.

ARDUINO Nano. Disponível em: < https://www.arduino.cc/en/Main/arduinoBoardNano >. Acesso em: 19 mar.2017.

COOLEY, J. W.; TURKEY, J. W. "An algorithm for the machine calculation of complex Fourier series." *Mathematics of computation* 19.90 (1965): 297-301.

ELECHOUSE. *Voice Recognition Module V3: Speak to Control (Arduino compatible)*. Disponível em: < https://www.elechouse.com/elechouse/images/product/VR3/VR3_ manual .pdf >. Acesso em: 26 mar.2017. 38p.

FECHINE, J. M. A Transformada de Fourier e suas aplicações. Grupo PET de Computação. Universidade Federal de Campina Grande – PB. **Ciclo de Seminários Técnicos**. 2010.

FELIPPETTO DE CASTRO, M. C. Capítulo 3 – Fundamentos de Comunicação de Dados. Teleprocessamento I. Disponível em: < http://www.feng.pucrs.br/~decastro/TPI/TPI_Cap3_parte2.pdf >. Acesso em: 26 mar. 2017.

FILHO, D.O.B. Curso de Arduino. Disponível em: < http://www.robotizando.com.br/curso_arduino_o_que_e_arduino_pg1.php >. Acesso em: 19 mar. 2017.

HAYES, T. apud LONGUEVILLE, D. 2011. FFT library**.** Disponível em < https://github.com/t3db0t/Deconspectrum/tree/master/software/PlainFFT >. Acesso em: 21 jul. 2016.

KLEINA, N. Reconhecimento de voz da Google tem só 8% de erro e não para de melhorar. 2015. Disponível em: < https://www.tecmundo.com.br/google-i-o-2015/80678-reconhecimento-voz-google-tem-so-8-erro-nao-de-melhorar.htm > 2011. Acesso em: 26 mar.2017.

LEE, A**. The Julius Book.** SourceForge: 2010.

LIMA, J.B.; CAMPELLO DE SOUZA; R. M.; OLIVEIRA, H. M. de; CAMPELLO DE SOUZA; M. M. Decodificação de Sinais DTMF via Transformada Aritmética de Fourier. In: **XXII SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES**, 2004, Belém. Recife: Universidade Federal de Pernambuco, 2004.

MACHADO, P. A.; MOECKE, M. Estudos Iniciais do Sistema didático para análise de sinais no domínio da frequência DSA-PC: tipos de janelas temporais. Disponível em: < http://wiki.sj.ifsc.edu.br/wiki/images/7/7f/Estudos-DSA.pdf > 2011. Acesso em: 19 mar.2017.

MARTIN, K .D.; KIM, Y.E. Musical instrument identification: a pattern recognition approach. 136th Meeting of the Acoustical Society of America, Norfolk, VA, October, 1998.

MELO, C. Fundamentos de Radio Frequência. Disponível em: < https://pt.slideshare.net/carlosvmelo/fundamentos-de-radio-freqncia >. Acesso em: 06 mar. 2017.

NEYRA-ARAOZ, J. H. Transformada de Fourier: fundamentos matemáticos, implementação e aplicações musicais. Disponível em: < https://www.ime.usp.br/~kon/MAC5900/seminarios/seminario_Jorge.pdf >. Acesso em: 06 mar.2017.

ORFANIDIS, S. J. *Introduction to Signal Processing*, Prentice-Hall, 1996.

PERICO, A.; SHINOHARA, C. S.; SARMENTO, C.D. **Sistema de reconhecimento de voz para automatização de uma plataforma elevatória.** 2014, 97f. Trabalho de Conclusão de Curso (Graduação em Engenharia Industrial Elétrica – Ênfase em Automação do Departamento Acadêmico de Eletrotécnica) – Universidade Tecnológica Federal do Paraná, Curitiba, 2014.

PETRY, A.; ZANUZ, A.; BARONE, D. A. C. Utilização de técnicas de processamento digital de sinais para a identificação automática de pessoas pela voz. **Simpósio sobre Segurança em Informática, São José dos Campos, SP**, 1999.

SANTANA, A. C. Projeto de Tese II. Programa de Pós-Graduação em Engenharia Elétrica. Universidade Federal de Minas Gerais. 2016.

SEARA da Ciência. Fourier e suas Séries Maravilhosas. Disponível em: < http://www.seara.ufc.br/tintim/matematica/fourier/ >. Acesso em: 21 jan. 2017.

SILVA MELO, M. C. **Trajetória Tecnológica Do Setor De Telecomunicações No Brasil: A Tecnologia VoIP.** 2008, 231f. Dissertação (Mestrado em Economia) – Universidade Federal de Santa Catarina, Florianópolis, 2008.

SOUZA, D. F. de; SOBRAL CINTRA, R. J. de ; OLIVEIRA , H. M. de. Uma Ferramenta para Análise de Sons Musicais: A Série Quantizada de Fourier. In: XXII **SIMPÓSIO BRASILEIRO DE TELECOMUNICAÇÕES**, 2005, Campinas. Campinas: Universidade Federal de Pernambuco, 2005.

THE FOURIER Transform.com. Fourier Transforms. Disponível em: < http://www.thefouriertransform.com/ >. Acesso em: 06 mar. 2017.

VERDAN, D. B. **Estudo e montagem de sistemas para acionamento remoto via sinais DTMF do celular.** 2016, 104f. Trabalho de Conclusão de Curso (Graduação em Engenharia de Controle e Automação) – Universidade Federal de Ouro Preto, Ouro Preto, 2016.

VIEIRA, V. Saiba Tudo Sobre o Arduino. Disponível em: < http://sejalivre.org/saiba-tudo-sobre-arduino/#disqus_thread > 2011. Acesso em: 19 mar.2017.

APÊNDICE A - CÓDIGO FONTE PARA ARDUINO NANO UTILIZANDO A TRANSFORMADA DE FOURIER

```
#include "PlainFFT.h"
PlainFFT FFT = PlainFFT(); // cria objeto FFT
const uint16_t samples = 128; // Quantidade de amostras tomadas antes de executar FFT
double samplingFrequency = 8000; //para voz
// Vetores de entrada e saída
double vReal[samples];
double vImag[samples];
uint16_t idx=0;
double picos[7]={0.0,0.0,0.0,0.0,0.0,0.0,0.0};
double medPicos[7]={0.0,0.0,0.0,0.0,0.0,0.0,0.0};
int medCont[7]={0,0,0,0,0,0,0};
int medCont2[7]={0,0,0,0,0,0,0};
int k=0, f=0, iteracao=0, w=0, g=0, comp=0;
//med é a quantidade de itens para a média e var é a quantidade de varreduras para cada execução do audio
float limite=0.2, var=30.0, idxCont=0.0, med=5.0, erro=30.0;

void setup(){
 Serial.begin(115200);
  pinMode(2,OUTPUT);
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);
  pinMode(8,INPUT);//A
  pinMode(9,INPUT);//B
  pinMode(10,INPUT);//C
  pinMode(11,INPUT);//D
  pinMode(12,INPUT);//E
}

void loop() {
```

```
 k=0;
 w=0;
 g=0;
 idxCont=0.0;
 while (k<=0){
  f=0;
  if(Serial.available()){
  Serial.println("Fale a palavra enquanto o LED estiver acesso.");
  }
  double frequencias[7]={0.0,0.0,0.0,0.0,0.0,0.0,0.0};
  idx=0;
  for(uint8_t j=1; j<=var; j++){
  digitalWrite(2,HIGH);
    for(uint16_t p=1; p<samples; p++){
    vReal[p] = analogRead(A0);
    }
  FFT.windowing(vReal, samples, FFT_WIN_TYP_HAMMING, FFT_FORWARD); //Redução de vazamento espectral (frequências parasitas/ruído no domínio da freq. devido a sinal não periódico)
  FFT.compute(vReal, vImag, samples, FFT_FORWARD); // Compute FFT
  FFT.complexToMagnitude(vReal, vImag, samples); // Compute magnitudes, Eixo y do espectro do sinal. O eixo x são as frequências.

for (uint16_t i = 1; i < ((samples >> 1)); i++) {      //samples>>1 divide samples por 2 ao deslocar um bit para a direita
     if(vReal[i]>limite){
       idxCont=idxCont+1.0;    //armazena quantidade de picos
       if (i+6>idx && i-6>idx && i+6<(samples >> 1)){    //estabelece distância entre picos
       idx=i;

       double delta = 0.5 * ((vReal[idx-1] - vReal[idx+1]) / (vReal[idx-1] - (2.0 * vReal[idx]) + vReal[idx+1]));
          double interpolated = ((idx+delta) /samples) * samplingFrequency;
```

```
    if(interpolated<4000 && interpolated>0){
    if (frequencias[1]==0.0){
    frequencias[1]=interpolated;
    }
    else if (frequencias[2]==0.0){
    frequencias[2]=interpolated;
    }
    else if (frequencias[3]==0.0){
    frequencias[3]=interpolated;
    }
    else if (frequencias[4]==0.0){
    frequencias[4]=interpolated;
    }
    else if (frequencias[5]==0.0){
    frequencias[5]=interpolated;
    }
    else if (frequencias[6]==0.0){
    frequencias[6]=interpolated;
    }
    else if (frequencias[7]==0.0){
    frequencias[7]=interpolated;
    }
   }
  }
 }
}//for do vReal
 }//for das varreduras
 idxCont=idxCont/var;
 //faz o controle do limite de captação de picos
 if(idxCont>8.0){//pode estar pegando ruído
  limite=limite+0.01;
  for(uint8_t i=1; i<=6; i++){
   frequencias[i]=0.0;
  }
```

```
    }
    else if(idxCont<4.0 && idxCont!=0.0){//se pegar menos picos e se não pegar pico, diminui
limite
      limite=limite-0.01;
    }

    digitalWrite(2,LOW);
    if(Serial.available()){
       Serial.println("medPicos frequencias");
        Serial.print(medPicos[1],2);
        Serial.print("\t");
        Serial.println(frequencias[1],2);
        Serial.print(medPicos[2],2);
        Serial.print("\t");
        Serial.println(frequencias[2],2);
        Serial.print(medPicos[3],2);
        Serial.print("\t");
        Serial.println(frequencias[3],2);
        Serial.print(medPicos[4],2);
        Serial.print("\t");
        Serial.println(frequencias[4],2);
        Serial.print(medPicos[5],2);
        Serial.print("\t");
        Serial.println(frequencias[5],2);
        Serial.print(medPicos[6],2);
        Serial.print("\t");
        Serial.println(frequencias[6],2);
        Serial.println("medCont:");
        Serial.println(medCont[1]);
        Serial.println(medCont[2]);
        Serial.println(medCont[3]);
        Serial.println(medCont[4]);
        Serial.println(medCont[5]);
        Serial.println(medCont[6]);
```

```
      Serial.print("idxCont:");
      Serial.println(idxCont);
      Serial.print("limite:");
      Serial.println(limite,2);
   }
   delay(2000);
   idxCont=0.0; //zera contador de picos
 ////////////////////////////////////////////////////////////////////////////////////
   if(frequencias[6]>30.0){ //Se até 4 frequências forem obtidas
     for (uint8_t i=1; i<=6; i++){

     if(frequencias[i]>erro){
     if(picos[1]+picos[2]+picos[3]+picos[4]+picos[5]+picos[6]==0.0){
       picos[i]=frequencias[i];
       medPicos[i]=picos[i];
     }
     else if ((frequencias[i]+erro)>=picos[1] && (frequencias[i]-erro)<=picos[1] &&
medCont[1]<=(med-1)){
     medPicos[1]=medPicos[1]+frequencias[i];
     picos[1]=frequencias[1];
     medCont[1]=medCont[1]+1;
     }
     else if ((frequencias[i]+erro)>=picos[2] && (frequencias[i]-erro)<=picos[2] &&
medCont[2]<=(med-1)){
     medPicos[2]=medPicos[2]+frequencias[i];
     picos[2]=frequencias[i];
     medCont[2]=medCont[2]+1;
     }
     else if ((frequencias[i]+erro)>=picos[3] && (frequencias[i]-erro)<=picos[3] &&
medCont[3]<=(med-1)){
     medPicos[3]=medPicos[3]+frequencias[i];
     picos[3]=frequencias[i];
     medCont[3]=medCont[3]+1;
     }
```

```
      else if ((frequencias[i]+erro)>=picos[4] && (frequencias[i]-erro)<=picos[4] && medCont[4]<(med-1)){
      medPicos[4]=medPicos[4]+frequencias[1];
      picos[4]=frequencias[i];
      medCont[4]=medCont[4]+1; // contador para garantir que pelo menos quatro valores de frequencia serão usados na média
      }
      else if ((frequencias[i]+erro)>=picos[5] && (frequencias[i]-erro)<=picos[5] && medCont[5]<=(med-1)){
      medPicos[5]=medPicos[5]+frequencias[i];
      picos[5]=frequencias[i];
      medCont[5]=medCont[5]+1;
      }
      else if ((frequencias[i]+erro)>=picos[6] && (frequencias[i]-erro)<=picos[6] && medCont[6]<=(med-1)){
      medPicos[6]=medPicos[6]+frequencias[i];
      picos[6]=frequencias[i];
      medCont[6]=medCont[6]+1;
      }
      if(picos[i]==0.0 && frequencias[i]>0){
       if(frequencias[i]!=picos[1] && frequencias[i]!=picos[2] && frequencias[i]!=picos[3] && frequencias[i]!=picos[4] && frequencias[i]!=picos[5] && frequencias[i]!=picos[6]){
        picos[i]=frequencias[i];
        medPicos[i]=picos[i];
        }
      }
     }
   }//for
   //se depois de 5 rodadas a frequência em picos não se repetir, zera ele e atribui a frequencia nova da próxima rodada
   iteracao=iteracao+1;
   if (iteracao==2){
    for(uint8_t i=1; i<=6; i++){
   medCont2[i]=medCont[i]; //armazena o contador depois de algumas iterações no inicio
```

```
    }
   }
   if (iteracao==5){ //se após 5 entradas no loop acima o medPicos não se alterar, de acordo com o seu valor nas primeiras iterações (medCont2) zera o valor de comparação "picos" para substituir pelo próximo
    iteracao=0;
    for(uint8_t c=1; c<=6; c++){
    if(medCont[c]==medCont2[c] && medCont[c]!=med){
    picos[c]=0.0;
    medPicos[c]=0.0;
    medCont[c]=0;
    }
   }
   }
   if (medCont[1]+medCont[2]+medCont[3]+medCont[4]+medCont[5]+medCont[6]>=(5*med)+1){ //quando pelo menos 5 frequencias já tiverem uma quantidade "med" de valores para a média
    k=1;//qualquer valor maior que zero
   }
   }//if
  }//while
  for (uint8_t i=1; i<=6; i++){
   if(medCont[i]!=0){
  medPicos[i]=medPicos[i]/(medCont[i]+1); //faz a média de "med" valores de picos acumulados.
   }
  medCont[i]=0;
  medCont2[i]=0;
  }
  if(Serial.available()){
   //Serial.println("Frequencias armazenadas.");
   Serial.println(medPicos[1],2);
   Serial.println(medPicos[2],2);
```

```
  Serial.println(medPicos[3],2);
  Serial.println(medPicos[4],2);
  Serial.println(medPicos[5],2);
  Serial.println(medPicos[6],2);

  Serial.println("Gravacao efetuada com sucesso.");
  Serial.println("Ative pino 8 para pronunciar a palavra e fazer o acionamento.");
  Serial.println("Ative pino 9 para gravar outra palavra.");
 }
 digitalWrite(4,HIGH);
  while(w==0){ //se byte pronto para leitura
  //opção para pronunciar palavra gravada e acionar uma carga
  if (digitalRead(8)==HIGH){
    digitalWrite(4,LOW);
      w=1;
  }
  //opção para gravar nova palavra
  if(digitalRead(9)==HIGH){
    digitalWrite(4,LOW);
      f=4;
      w=1;
   }
 }//while
 w=0;
////////////////////////////////////////////////////////////////////////////////////////

 while(f<=3 && g==0){
  if(Serial.available()){
   Serial.println("Repita a palavra gravada enquanto o LED estiver aceso, para entrar.");

   Serial.println("A qualquer momento acione o pino 12 para gravar nova palavra.");
  }
   if (digitalRead(12)==HIGH){
      digitalWrite(5,LOW); //inverte estado do LED
```

```
      g=1; //sai da condição do while e pede novamente para fazer a gravação
      f=4;
   }

   double frequencias[7]={0.0,0.0,0.0,0.0,0.0,0.0,0.0};
   idx=0;
   idxCont=0.0;
   for(uint8_t j=1; j<=var; j++){
   digitalWrite(2,HIGH);
     for(uint16_t i=1; i<samples; i++){
     vReal[i] = analogRead(A0);
     }
   FFT.windowing(vReal, samples, FFT_WIN_TYP_HAMMING, FFT_FORWARD); //Redução de vazamento espectral (frequências parasitas/ruído no domínio da freq. devido a sinal não periódico)
   FFT.compute(vReal, vImag, samples, FFT_FORWARD); // Compute FFT
   FFT.complexToMagnitude(vReal, vImag, samples); // Compute magnitudes, Eixo y do espectro do sinal. O eixo x são as frequências.

for (uint16_t i = 1; i < ((samples >> 1)); i++) {//samples>>1 divide samples por 2 ao deslocar um bit para a direita
     if(vReal[i]>limite){
      idxCont=idxCont+1.0;
      if (i+6>idx && i-6>idx && i+6<(samples >> 1)){
      idx=i;

      double delta = 0.5 * ((vReal[idx-1] - vReal[idx+1]) / (vReal[idx-1] - (2.0 * vReal[idx]) + vReal[idx+1]));
          double interpolated = ((idx+delta) /samples) * samplingFrequency;

      if (interpolated<4000 && interpolated>0){
      if (frequencias[1]==0.0){
      frequencias[1]=interpolated;
      }
```

```
   else if  (frequencias[2]==0.0){
   frequencias[2]=interpolated;
   }
   else if (frequencias[3]==0.0){
   frequencias[3]=interpolated;
   }
   else if (frequencias[4]==0.0){
   frequencias[4]=interpolated;
   }
   else if (frequencias[5]==0.0){
   frequencias[5]=interpolated;
   }
   else if (frequencias[6]==0.0){
   frequencias[6]=interpolated;
   }
   else if (frequencias[7]==0.0){
   frequencias[7]=interpolated;
   }
  }
 }
}
}//for
idxCont=idxCont/var;
//faz o controle do limite de captação de picos
if(idxCont>8.0){
 limite=limite+0.01;
 for(uint8_t i=1; i<=6; i++){
  frequencias[i]=0.0;
 }
}
else if(idxCont<5.0 && idxCont!=0.0){
 limite=limite-0.01;
}
```

```
digitalWrite(2,LOW);
delay(2000);
if(Serial.available()){
    Serial.println("Gravadas  Recem-coletadas");
    Serial.print(medPicos[1],2);
    Serial.print("\t");
    Serial.println(frequencias[1],2);
    Serial.print(medPicos[2],2);
    Serial.print("\t");
    Serial.println(frequencias[2],2);
    Serial.print(medPicos[3],2);
    Serial.print("\t");
    Serial.println(frequencias[3],2);
    Serial.print(medPicos[4],2);
    Serial.print("\t");
    Serial.println(frequencias[4],2);
    Serial.print(medPicos[5],2);
    Serial.print("\t");
    Serial.println(frequencias[5],2);
    Serial.print(medPicos[6],2);
    Serial.print("\t");
    Serial.println(frequencias[6],2);
}
    comp=0;
if(frequencias[6]>30){ //quando o vetor frequências receber quatro leituras
  for (uint8_t i=1; i<=6; i++){
  if ((frequencias[i]+erro)>=medPicos[1] && (frequencias[i]-erro)<=medPicos[1]){
    comp=comp+1;
  }
  else if ((frequencias[i]+erro)>=medPicos[2] && (frequencias[i]-erro)<=medPicos[2]){
    comp=comp+1;
  }
  else if ((frequencias[i]+erro)>=medPicos[3] && (frequencias[i]-erro)<=medPicos[3]){
```

```
      comp=comp+1;
     }
     else if ((frequencias[i]+erro)>=medPicos[4] && (frequencias[i]-erro)<=medPicos[4]){
      comp=comp+1;
     }
     else if ((frequencias[i]+erro)>=medPicos[5] && (frequencias[i]-erro)<=medPicos[5]){
      comp=comp+1;
     }
     else if ((frequencias[i]+erro)>=medPicos[6] && (frequencias[i]-erro)<=medPicos[6]){
      comp=comp+1;
     }
     }//for
   }//if
   if(Serial.available()){
   Serial.println(comp);
   }
   if (comp>=5){//Se pelo menos três frequencias baterem medPicos com margem de erro de
+/- 20Hz
   comp=0;
   digitalWrite(5,HIGH);
   w=0;
  if(Serial.available()){
   Serial.println("Acione o pino 10 para desativar carga e reativar senha por fala ou o pino 11
para gravar nova palavra.");
  }
      while(w==0){
      if(digitalRead(10)==HIGH){
        digitalWrite(4,LOW);
        digitalWrite(5,LOW); //inverte estado do LED
       f=0; //não sai da condição do while e pede novamente para fazer a pronúncia
       w=1;
      }
      if (digitalRead(11)==HIGH){
        digitalWrite(4,LOW);
```

```
        digitalWrite(5,LOW);
       f=4;  //sai da condição do while e reinicia o void loop
       w=1;
     }
  }//while
}//if

}//while
}//void loop
```