**UNIVERSIDADE DE BRASÍLIA**
**FACULDADE DE TECNOLOGIA**
**DEPARTAMENTO DE ENGENHARIA ELÉTRICA**

# SISTEMA DE ACIONAMENTO DE DISPOSITIVOS COMANDADO POR VOZ

**PEDRO HENRIQUE DE OLIVERA RAMIRO**

**ORIENTADOR: ALEXANDRE RICARDO SOARES ROMARIZ**

**PROJETO FINAL DE GRADUAÇÃO**
**ENGENHARIA DE REDES DE COMUNICAÇÃO**

**BRASÍLIA – DF: 09/2010**

# UNIVERSIDADE DE BRASÍLIA
# FACULDADE DE TECNOLOGIA
# DEPARTAMENTO DE ENGENHARIA ELÉTRICA

## SISTEMA DE ACIONAMENTO DE DISPOSITIVOS COMANDADO POR VOZ


## PEDRO HENRIQUE DE OLIVEIRA RAMIRO


PROJETO FINAL DE GRADUAÇÃO SUBMETIDA AO DEPARTAMENTO DE ENGENHARIA ELÉTRICA DA FACULDADE DE TECNOLOGIA DA UNIVERSIDADE DE BRASÍLIA, COMO PARTE DOS REQUISITOS NECESSÁRIOS PARA A OBTENÇÃO DO GRAU DE ENGENHEIRO.

APROVADA POR:

---

ALEXANDRE RICARDO SOARES ROMARIZ, Doutor, UnB
(ORIENTADOR)

---

JANAÍNA GONÇALVES GUIMARÃES, Doutora, UnB
(EXAMINADOR)

---

RICARDO ZELENOVSKY, Doutor, UnB
(EXAMINADOR)

DATA: BRASÍLIA – DF, 1 DE SETEMBRO DE 2010.

Dedico este trabalho a minha família, que sempre esteve ao meu lado.

# Agradecimentos

Ao meu orientador Prof. Dr. Alexandre Ricardo Soares Romariz, pelo constante apoio, incentivo e dedicação essenciais para o desenvolvimento deste trabalho.

Aos meus amigos que conviveram comigo durante estes anos de graduação.

Aos professores, que, além de mentores do conhecimento técnico, foram importantes em minha formação pessoal.

## Resumo

O reconhecimento automático da fala tem sido objeto de estudo há mais de 50 anos e tem grande importância e espaço no mundo atual. Uma das grandes dificuldades encontradas nesta área é sua interdisciplinaridade.

A grande maioria dos sistemas implementados no estudo do reconhecimento automático da fala baseiam-se em modelos estatísticos, especialmente em modelos ocultos de Markov (HMM).

O objetivo deste trabalho é o desenvolvimento de um sistema de acionamento de dispositivos por comando de voz baseado em um sistema de reconhecimento de palavras isoladas que utiliza modelos ocultos de Markov em seu mecanismo. O sistema deve ser capaz de reconhecer dois dispositivos distintos (“Televisão” e “Lâmpada”) e duas ações (“Acionar” e “Desligar”). Uma vez realizado o reconhecimento, dados são enviados ao hardware responsável pelo acionamento por meio da porta USB. Todo o processo será detalhado no decorrer deste relatório.

# Sumário

# Lista de figuras

# Lista de tabelas

# 1. Introdução

O ser humano apresenta a linguagem oral como forma mais comum de comunicação. É de muito interesse que essa mesma forma de comunicação possa intermediar as relações entre homem e máquina. Esse desejo motiva o estudo de sistemas de reconhecimento e síntese de voz com o intuito de desenvolver uma interface homem-máquina baseada na linguagem oral. Diversas técnicas foram e têm sido desenvolvidas com a finalidade de implementar sistemas capazes de ser essa interface. Entretanto, ainda se está muito distante de um sistema capaz de compreender um discurso de qualquer teor, falado de forma natural, por qualquer pessoa, em qualquer lugar.

## 1.1. Definição do problema

O reconhecimento automático de voz consiste em capturar um sinal acústico produzido pelo homem, convertê-lo em um sinal digital e, a partir de uma base de dados, identificar um conjunto de palavras.

Uma das grandes dificuldades presente na área de reconhecimento de voz é a grande quantidade de temas envolvidos [1]. Dentre esses temas estão o processamento de sinais, reconhecimento de padrões, inteligência artificial, lingüística, fonética, acústica e outras. Ainda, os sistemas de reconhecimento de voz devem ser capazes de atuar em ambientes com presença de ruído.

O principal limitador de desempenho de um sistema de reconhecimento de voz é a variabilidade dos sinais de fala. Essa variabilidade pode ser dada por diversos fatores. Entre eles:

- Variabilidade de sons para um único locutor e entre locutores distintos.
- Variabilidade de ambientes em que um sistema é utilizado.
- Variabilidade de ruído de fundo.
- Variabilidade na produção da fala (diferente para cada locutor).

De forma geral, essas variabilidades não podem ser eliminadas, tendo o sistema a função de amenizá-las.

## 1.2. Onde o reconhecimento de voz está presente

Qualquer atividade em que haja interação homem-máquina pode ser potencialmente vista como uma aplicação do reconhecimento de voz. Hoje em dia, diversas atividades já apresentam o uso do reconhecimento de voz. Dentre essas atividades, estão:

- Sistemas de telefonia: comandos de voz para efetuar uma chamada; comandos de voz para acessar menus em centrais de atendimento ao cliente.
- Robótica: comunicação (fala e compreensão) com seres humanos.
- Sistemas de transcrição: conversão de textos falados por um usuário em texto.
- Sistemas de controle e comando: sistemas que utilizam a fala para a realização de determinadas funções.

## 1.3. Visão geral do trabalho

O objetivo deste trabalho é desenvolver um sistema de reconhecimento de voz com vocabulário limitado, que permita comando de voz para acionamento de sistemas elétricos via computador. Tendo esse objetivo em mente, um sistema de reconhecimento de palavras isoladas com um pequeno vocabulário é o ideal para o início de estudos nessa área. Desenvolveu-se, então, um sistema de reconhecimento baseado nos modelos ocultos de Markov (HMM).

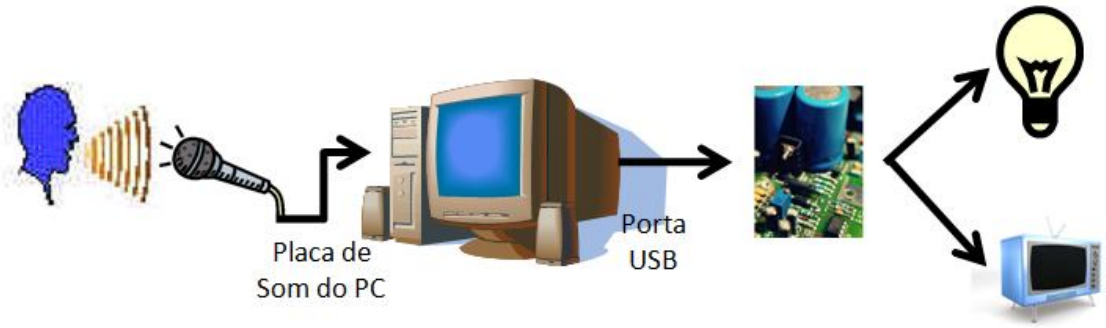


**Figura 1 - Visão geral do trabalho**

Este trabalho foi dividido nos seguintes capítulos:

- Capítulo 1: introdução sobre o tema.
- Capítulo 2: apresenta o histórico, características de sistemas de reconhecimento de fala e a teoria sobre modelos ocultos de Markov.
- Capítulo 3: descreve como o projeto foi implementado.
- Capítulo 4: apresenta e discute os resultados obtidos.

Capítulo 5: trata das conclusões e propõe trabalhos futuros.

# 2. Sistemas de reconhecimento de fala e modelos ocultos de Markov

Como dito anteriormente, um sistema de reconhecimento de voz converte o sinal acústico produzido pelo homem em um sinal digital de áudio por meio de um hardware associado e, a partir de uma base de dados, identifica o conjunto de palavras faladas.

## 2.1. Breve histórico

Há mais de 50 anos, o reconhecimento automático da fala tem sido objeto de pesquisas [1]. A literatura afirma que o primeiro estudo sobre reconhecimento de fala ocorreu em 1952 nos laboratórios Bell [7]. Esse estudo inicial deu origem a um sistema de reconhecimento de dígitos isolados dependente do locutor.

Nas décadas de 50 e 60, as estratégias mais usadas no reconhecimento de voz tinham como base a segmentação do sinal acústico em fonemas e, por meio de uma análise espectral, reconhecer o fonema em questão. Já no final da década de 60, pesquisadores do NTT Labs desenvolveram a técnica *Linear Predictive Coding* (LPC), que simplificou a análise da voz. Entretanto, inicialmente, o LPC era apenas utilizado na codificação da fala. Nos anos 70, Rabiner e Levinson (e outros) utilizaram o LPC no reconhecimento de fala. Nessa época, para um vocabulário pequeno, o paradigma predominante para o reconhecimento da fala era o *Dynamic Time Warping* (DTW) [8]. A técnica DTW apresentava bons resultados para o reconhecimento de palavras isoladas com vocabulário pequeno e deu origem aos primeiros sistemas de reconhecimento de voz comerciais.

Os primeiros métodos estatísticos para o reconhecimento da fala surgiram nos anos 80. Desde então, o mais utilizado desses métodos é o baseado em modelos ocultos de Markov ou *Hidden Markov Models* (HMM). Ainda na década de 80, foi introduzida a técnica de redes neurais aplicadas ao reconhecimento de voz.

Já a partir da década de 90, os estudos visam ao reconhecimento de fala contínua, com vocabulário ilimitado e independente do locutor. Para isso, muitas pesquisas se voltam para a solução de problemas como a robustez ao ruído, adaptação ao locutor, distorções introduzidas pelo canal de transmissão, etc.

## 2.2. Tipos de sistemas de reconhecimento de fala

Para caracterizarmos um sistema de reconhecimento de voz, diversos itens podem ser abordados. Dentre eles estão o tamanho do vocabulário, a dependência ou a independência do locutor, o modo de pronúncia, etc.

Quanto ao tamanho do vocabulário, o sistema pode ser:

- Vocabulário pequeno: até 20 palavras;
- Vocabulário médio: entre 20 e 100 palavras;
- Vocabulário grande: entre 100 e 1000 palavras;
- Vocabulário muito grande: acima de 1000 palavras.

Quanto à dependência ou à independência do locutor, pode ser:

- Dependente do locutor: reconhece a fala de pessoas cujas vozes foram utilizadas para treinar o sistema.
- Independente do locutor: procura reconhecer a fala de qualquer pessoa. Para a implementação de um sistema independente, este deve ser treinado com o maior número possível de pessoas com características distintas (sexo, idade, sotaque, etc.).

Quanto ao modo de pronúncia, o sistema pode ser:

- Reconhecedor de palavras isoladas: este tipo de sistema reconhece palavras faladas isoladamente, ou seja, com uma pausa mínima entre cada palavra para que sejam detectados o início e o fim de cada uma dessas palavras.
- Reconhecedor de palavras conectadas: sistema mais complexo que os reconhecedores de palavras isoladas e que utilizam palavras como unidade fonética padrão. São capazes de reconhecer sentenças completas pronunciadas sem pausa entre as palavras.
- Reconhecedor de fala contínua: é capaz reconhecer a fala expressa de forma natural, sem nenhuma peculiaridade quanto à pronúncia. São os sistemas mais complexos e difíceis de serem realizados, pois contêm uma enorme gama de peculiaridades da fala natural que devem ser tratadas.

## 2.3. Modelos ocultos de Markov

A teoria básica de modelos ocultos de Markov foi publicada por Baum, em conjunto com outros pesquisadores, e foi usada em sistemas de reconhecimento de voz pela primeira vez nos anos 70. Entretanto, somente nos últimos anos, os modelos ocultos de Markov têm se tornado a principal ferramenta utilizada em sistemas de reconhecimento de fala.

Os processos de Markov têm aplicações em diversas áreas e se caracterizam por não possuírem memória, isto é, toda a informação passada é resumida integralmente no estado atual.

Um modelo de Markov, também conhecido como cadeia de Markov, é um conjunto finito de estados ligados entre si por transições. Essas transições estão relacionadas a um processo estocástico. Existe ainda um outro processo estocástico associado a um modelo de Makov que envolve as observações de saída de cada estado. Se um observador externo ao processo enxergar apenas as observações de saída, pode-se afirmar que os estados estão ocultos. Portanto, o processo estocástico relacionado às transições não é visível e, daí, surge o nome Modelos Ocultos de Markov (*Hidden Markov Models*).

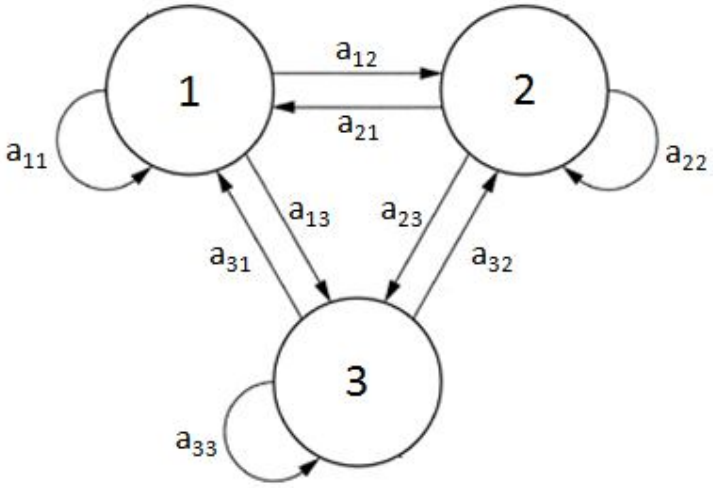


**Figura 2 - Cadeia de Markov de 3 símbolos**

Observando a Figura 2, nota-se que uma cadeia de Markov é também uma máquina de estados. Diversos fenômenos podem ser modelados por meio de máquinas de estados. Se algum desses fenômenos possuir características de processos estocásticos, ele pode ser modelado por um HMM.

### 2.3.1. Elementos de um HMM

Com o objetivo de definir completamente um modelo oculto de Markov, os seguintes elementos são necessários [10]:

- O número de estados do modelo, N. Os estados individuais são rotulados como $S = \{S_1, S_2, \dots, S_N\}$, e o estado em *t* como $q_t$.
- O número de símbolos observáveis distintos por estado, M. Os símbolos individuais são denotados como $V = \{v_1, v_2, \dots, v_M\}$.
- Um conjunto de transições de probabilidade do estado, $A = \{a_{ij}\}$, onde

$$a_{ij} = P(q_{t+1} = S_j | q_t = S_i),\ 1 \leq i, j \leq N \text{ e}$$

$$\sum_{j=1}^{N} a_{ij} = 1,$$

$$1 \leq i \leq N$$

- A distribuição de probabilidade de símbolos de observações no estado *j*, $B = \{b_j(k)\}$, onde

$$b_j(k) = P(O_t = v_k | q_t = S_j)$$

em que $v_k$ é o k-ésimo símbolo individual e $O_t$ é vetor de parâmetros atual.

- A distribuição de probabilidades inicial $\pi = \{\pi_i\}$, onde

$$\pi_i = P(q_1 = S_i),\ 1 \leq i \leq N.$$

Portanto, para uma definição completa de um HMM são necessárias as especificações dos parâmetros *N* e M, a sequência de observações ($O$ =

$O_1, O_2, \ldots, O_T$, onde *T* é o número de observações na sequência) e a especificação de três conjuntos de medidas de probabilidade *A*, *B* e $\pi$. De acordo com o padrão da literatura [1], será utilizada a notação compacta

$$\lambda = (A, B, \pi)$$

para indicar o conjunto de parâmetros completo do modelo.

### 2.3.2. Topologias dos modelos ocultos de Markov

Geralmente, usam-se duas topologias de HMM [10]:

- Modelo esquerda-direita.
- Modelo ergódico.

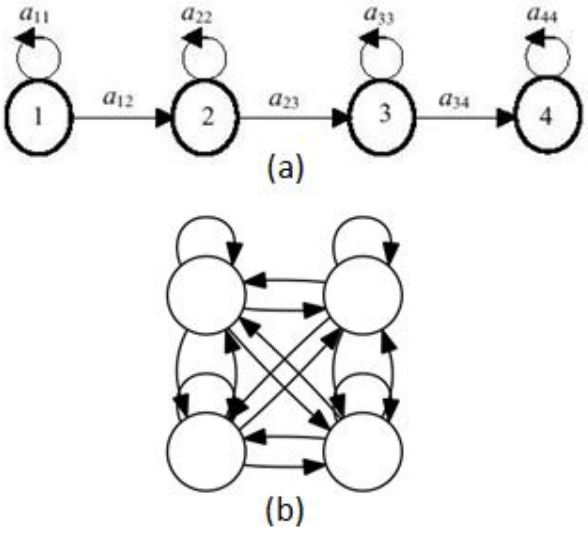


**Figura 3 - As duas topologias mais usadas de HMM. a) Modelo esquerda-direita. b) Modelo ergódico**

O modelo ergódico é um processo aleatório em que, analisando ao longo do tempo, a média temporal de uma longa realização observada tende à média das transições de estado na cadeia. Ou seja, as probabilidades de transição da cadeia podem ser obtidas ao observar médias de longos eventos do processo.

Já o modelo esquerda-direita possui este nome devido à propriedade de que, a medida que o tempo aumenta, o índice do estado aumenta ou permanece o

mesmo. Para um modelo esquerda-direita, a distribuição de probabilidades do estado inicial possui a seguinte propriedade:

$$\pi_i = \begin{cases} 0, i \neq 1 \\ 1, i = 1 \end{cases}$$

ou seja, a sequência de estados deve começar no estado $S_1$.

Ambos os modelos apresentados na Figura 3, podem ser generalizados de modo a possuir um número arbitrário de estados. Para um número muito grande de estados, determinar de forma ótima as matrizes A e B se torna muito difícil. Para a determinação do número de estados, não existem meios teóricos.

Geralmente, para o reconhecimento da fala, utiliza-se um modelo esquerda-direita simplificado - Figura 3 (b). Seguindo boa parte da literatura, optou-se por utilizar esse tipo de modelo neste trabalho. Assim, apenas transições para o mesmo estado ou estados mais à direita são permitidas.

### 2.3.3. Os três problemas básicos do HMM

No desenvolvimento de sistemas modelados por HMM's, existem três problemas básicos [10]:

- Problema da avaliação: dado um modelo $\lambda = (A, B, \pi)$ e uma sequência de observações $O = O_1, O_2, \dots, O_T$, como calcular eficientemente $P(O|\lambda)$, a probabilidade da sequência de observações, dado o modelo?
- Problema da decodificação: dado um modelo $\lambda = (A, B, \pi)$ e uma sequência de observações $O = O_1, O_2, \dots, O_T$, qual a melhor sequência dentro do modelo capaz de gerar essas observações?
- Problema do treinamento: dado um modelo $\lambda = (A, B, \pi)$ e uma sequência de observações $O = O_1, O_2, \dots, O_T$, como ajustar os parâmetros do modelo $\lambda = (A, B, \pi)$ de modo a maximizar o valor $P(O|\lambda)$?

O problema da avaliação ocorre ao tentar selecionar, dentre vários modelos, aquele que mais provavelmente gerou uma dada sequência de observações. Ao resolver este problema, obtém-se a solução, também, para o reconhecimento de

palavras isoladas onde cada palavra é representada por um modelo. Determina-se a palavra falada comparando-se as probabilidades de cada modelo ter gerado uma dada sequência de dados.

O problema da decodificação tem como objetivo descobrir, a partir de uma sequência de observações, qual foi a sequência de estados com maior probabilidade de ser a geradora daquela. Este tipo de problema é encontrado em sistemas de reconhecimento de fala conectada. Assim, cada palavra possuiria um modelo, porém todas as palavras são colocadas em conjunto formando um modelo global.

Assim, podemos afirmar que as soluções dos problemas da avaliação e da decodificação são maneiras de como obter resultados a partir de sequências de observações e modelos com parâmetros determinados. Entretanto, ainda há a necessidade de saber como criar um modelo de Markov para representar um dado fenômeno físico. A resposta vem com a solução do problema de treinamento, que é o mais complexo e importante. Com a solução desse problema pode-se, por exemplo, a partir de locuções de uma determinada palavra, criar um modelo a ser utilizado para o reconhecimento de outras locuções da mesma palavra.

### 2.3.4. Algoritmos para a solução dos problemas básicos

Em aplicações reais, é necessária a solução de cada um dos três problemas básicos dos modelos ocultos de Markov. Para isso, foram desenvolvidos alguns algoritmos dentre os quais estão o algoritmo *Forward* e o algoritmo *Backward* para o problema de avaliação, o algoritmo de *Viterbi* para o problema de decodificação e o algoritmo de *Baum-Welch* para o problema de treinamento.

#### 2.3.4.1 Algoritmo Forward

Dado um modelo $\lambda = (A, B, \pi)$ e uma sequência de observações $O = O_1, O_2, \dots, O_T$, o objetivo é encontrar a probabilidade de ocorrer aquela sequência de observações, $P(O|\lambda)$. Esse cálculo pode ser feito usando recursos simples de probabilidade, mas esse cálculo envolve um número de operações na ordem de $N^T$. Mesmo que $T$ não seja muito grande, o número de operações pode ser proibitivo.

Assim será apresentado um método de baixa complexidade e que faz uso de uma variável auxiliar, $\alpha_t(i)$, onde

$$\alpha_t(i) = P(O_1, O_2, \dots, O_t, q_t = S_i|\lambda)$$

ou seja, $\alpha_t(i)$ é definida como a probabilidade de uma sequência terminar em $S_i$.

Assim, recursivamente resolve-se $\alpha_t(i)$ da seguinte maneira:

1. Inicialização:

$$\alpha_1(i) = \pi_i b_i(O_1), \qquad 1 \leq i \leq N$$

2. Indução:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \qquad 1 \leq t \leq T-1 \qquad e \qquad 1 \leq j \leq N$$

3. Finalização:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i)$$

A probabilidade da sequência de observações, dado um modelo $P(O|\lambda)$ é calculada usando-se todas as variáveis $\alpha_t(i)$, para $t = T$ em todos os estados.

#### 2.3.4.2. Algoritmo Backward

Outra opção para resolver o problema da avaliação é o algoritmo *Backward*. De forma similar, usa-se uma variável auxiliar, $\beta_t(i)$, onde

$$\beta_t(i) = P(O_1, O_2, \dots, O_t, q_t = S_i|\lambda)$$

ou seja, $\beta_t(i)$ é definida como a probabilidade de uma sequência terminar em $S_i$.

Também recursivamente, $\beta_t(i)$ é resolvido:

1. Inicialização:

$$\beta_T(i) = 1, \qquad 1 \leq i \leq N$$

2. Indução:

$$\beta_T(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \qquad t = T-1, T-2, \ldots, 1 \qquad 1 \leq t \leq N$$

3. Finalização:

$$P(O|\lambda) = \sum_{i=1}^{N} \pi_i b_i(O_i) \beta_i(i)$$

Como pode ser observado, o algoritmo *Backward* possui recursão no sentido oposto ao do algoritmo *Forward*. Ambos são utilizados para a solução do problema da avaliação, mas apenas um deles é necessário para a realização da tarefa.

#### 2.3.4.3. Algoritmo de Viterbi

Com o objetivo de encontrar a melhor sequência de estados, $Q = \{q_1, q_2, \ldots, q_T\}$, para uma da sequência de observações $O = O_1, O_2, \ldots, O_T$, define-se a quantidade

$$\delta_t(i) = \max_{q_1, q_2, \ldots, q_{t-1}} P(q_1 q_2 \ldots q_t = i, O_1 O_2 \ldots O_T | \lambda)$$

ou seja, $\delta_t(i)$ é a maior probabilidade ao longo de um caminho no instante *t*, que considera as *t* primeiras observações e finaliza no estado $S_i$. Por indução tem-se:

$$\delta_{t+1}(j) = \left[\max\left(\delta_t(i) a_{ij}\right)\right] b_j(O_{t+1})$$

Para recuperar a sequência de estados, é necessário manter os argumentos que maximizam a expressão anterior, para cada *i* e *j*. Isto é realizado por meio de uma matriz $\Psi_t(j)$. A seguir, o procedimento completo para se obter a melhor sequência de estados:

1. Inicialização:

$$\delta_t(i) = \pi_i b_i(O_1), \qquad 1 \leq i \leq N$$

$$\Psi_1(i) = 0, \qquad 1 \leq i \leq N$$

2. Recursão:

$$\delta_t(j) = \max_{1 \leq i \leq N} \left[\delta_{t-1}(i) a_{ij}\right] b_j(O_t), \qquad 2 \leq t \leq T, \qquad 1 \leq j \leq N$$

$$\Psi_t(j) = \operatorname*{argmax}_{1 \leq i \leq N} \left[\delta_{t-1}(i) a_{ij}\right], \qquad 2 \leq t \leq T, \qquad 1 \leq j \leq N$$

3. Finalização:

$$q_T^* = \operatorname*{argmax}_{1 \leq i \leq N} [\delta_T(i)]$$

4. *Backtracking*:

$$q_T^* = \Psi_{t+1}(q_{t+1}^*), \qquad t = T-1, T-2, \ldots, 1$$

Portanto, o algoritmo de *Viterbi* tem seu início com o cálculo de $\delta_t(i)$ e, usando a recursão, mantém um ponteiro apontado ao estado com maior semelhança com aquele estágio do processo. Depois, na finalização, o estado $q_T^*$ é encontrado e a partir desse é feito um rastreamento recursivo para encontrar os estados que têm um ponteiro apontado para si. Esse conjunto de estados formado pelo estado $q_T^*$ e pelos estados apontados é o resultado do algoritmo de *Viterbi*.

#### 2.3.4.4. Algoritmo de Baum-Welch

Dada uma sequência finita de observações para se realizar o treinamento, não existe uma maneira ótima de estimar os parâmetros do modelo. Porém, pode-se escolher $\lambda = (A, B, \pi)$ tal que $P(O|\lambda)$ é localmente maximizada usando técnicas de gradiente ou um procedimento iterativo tal como o método de *Baum-Welch* (também conhecido como método *EM – expectation-maximization*).

Para a estimação dos parâmetros do HMM, o algoritmo *Baum-Welch* é o mais recomendado [10]. Esse algoritmo é apresentado em termos das variáveis $\alpha_t$ e $\beta_t$ e realiza a re-estimação dos parâmetros $a_{ij}$ e $b_{ij}$ [11].

Para uma única sequência de observações $O = O_1, O_2, \ldots, O_T$, a re-estimação da probabilidade de transição do estado *i* para o estado *j* da matriz de transição de estados *A* é dada por:

$$\bar{a}_{ij} = \frac{\sum_{t=1}^{T-1} a_t(i) a_{ij} b_j(O_{t+1}) \beta_{t+1}(j)}{\sum_{t=1}^{T-1} \alpha_t(i) \beta_t(j)}$$

Em HMM's discretos, a quantidade de símbolos de saída é finita. Também para uma única elocução, a re-estimação da função de probabilidade para que um estado *q*, emita um símbolo $O_t = v_k$ é obtida por

$$\bar{b}_i(k) = \frac{\sum_{t=1}^{T-1} a_t(i) \beta_t(j) \; em \; que \; O_t = v_k}{\sum_{t=1}^{T-1} \alpha_t(i) \beta_t(j)}$$

onde

$$\bar{b}_i(k) \geq 0, \qquad 1 \leq i \leq N, \qquad 1 \leq k \leq M$$

$$\sum_{k=1}^{M} \bar{b}_i(k) = 1, \qquad 1 \leq i \leq N$$

Resumidamente, o algoritmo de *Baum-Welch* trabalha atribuindo probabilidades iniciais a todos os parâmetros. Então, até que o treinamento convirja, ajusta as probabilidades dos parâmetros para aumentar a probabilidade que o modelo faça parte do conjunto treinado.

# 3. Sistema desenvolvido

Nos capítulos anteriores, foram apresentados alguns conceitos e definições úteis ao desenvolvimento do sistema proposto por este trabalho. Este capítulo apresentará o desenvolvimento e a implementação do sistema prático de acionamento de dispositivos elétricos por meio de comandos de voz.

O sistema de acionamento de dispositivos elétricos por comandos de voz foi implementado a partir do desenvolvimento de um software capaz de reconhecer certas palavras isoladas proferidas por um determinado locutor. Cada uma dessas palavras, as quais o sistema é capaz de reconhecer, faz parte de um vocabulário pré-definido.

Esse software responsável pelo reconhecimento das palavras foi desenvolvido utilizando a ferramenta matemática MATLAB, versão 7.6.0.324, tendo como base a teoria de Modelos Ocultos de Markov (HMM). O vocabulário base pré-definido para o sistema é composto de quatro palavras: “Televisão”, “Lâmpada”, “Acionar” e “Desativar”. Ou seja, duas palavras que identificam dispositivos elétricos (“Televisão” e “Lâmpada”) e duas que identificam ações (“Acionar” e “Desligar”) que podem ser efetuadas sobre esses dispositivos. Como dito anteriormente, o software de reconhecimento é capaz de tratar palavras isoladas. Assim, para gerar um comando, deve ser pronunciado pelo locutor o dispositivo a ser acionado e, após uma breve pausa, a ação a ser efetuada. O sistema só decidirá se alguma ação será realizada ao identificar uma palavra de comando (“Acionar” ou “Desligar”). Caso algum desses comandos seja identificado, o software de reconhecimento verifica se alguma palavra que identifica um dispositivo foi reconhecida anteriormente à palavra de comando. Caso positivo, o software executará uma nova rotina que realiza o comando. Caso negativo, o sistema apenas ignora a última palavra reconhecida.

Uma característica do MATLAB usada no sistema é o fato de se poderem executar aplicativos externos ao aplicativo desenvolvido nessa linguagem. Tirando proveito dessa qualidade, optou-se por criar um aplicativo em C++ para realizar o acesso e a transferência de dados ao hardware que faz o controle dos dispositivos elétricos. Assim, quando o software responsável pelo reconhecimento identifica um comando, esse aplicativo em C++ é executado. Para fazer a transferência do comando identificado no software de reconhecimento para o aplicativo em C++, é utili-

zado um arquivo de texto (*.txt*). Dessa forma, assim que um comando é identificado, o software de reconhecimento gera um arquivo de texto, que contém o comando a ser realizado, e aciona o aplicativo em C++. Esse aplicativo lê o comando no arquivo texto e realiza o acesso ao hardware e seguido da transferência do comando.

O hardware desenvolvido, por sua vez, tem como princípios básicos o recebimento de um comando, a interpretação desse comando e a realização de uma ação referente ao comando interpretado. Optou-se por desenvolver um hardware que se conectasse ao computador por meio de uma porta USB. Essa escolha foi feita porque, apesar de uma maior dificuldade de interfaceamento quando comparada a portas Serial ou Paralela, a porta USB está presente hoje em dia em praticamente todos os computadores pessoais, enquanto os outros tipos de portas estão em desuso.

Resumidamente, o hardware é composto por um chipset que recebe sinais gerados pelo aplicativo em C++, interpreta-os e disponibiliza como saída um conjunto de *strings* de comando. Essas *strings* serão enviadas a um microcontrolador que realiza a ação contida nas *strings*. Mais detalhes sobre o hardware serão apresentados posteriormente.

## 3.1. Software de reconhecimento de palavras isoladas

Visando ao desenvolvimento de um sistema prático de reconhecimento, onde o vocabulário é composto por dispositivos e ações, inicialmente foi implementado um software capaz de identificar os dígitos de 0-9 pronunciados por um único locutor. Assim que os resultados para esse sistema mostraram-se satisfatórios, o sistema prático foi desenvolvido.

Simplificadamente, o software desenvolvido tem dois modos de operação: o modo de treinamento (executado isoladamente) e o modo de reconhecimento (em tempo real). A Figura 4 mostra o sistema em forma de diagrama.

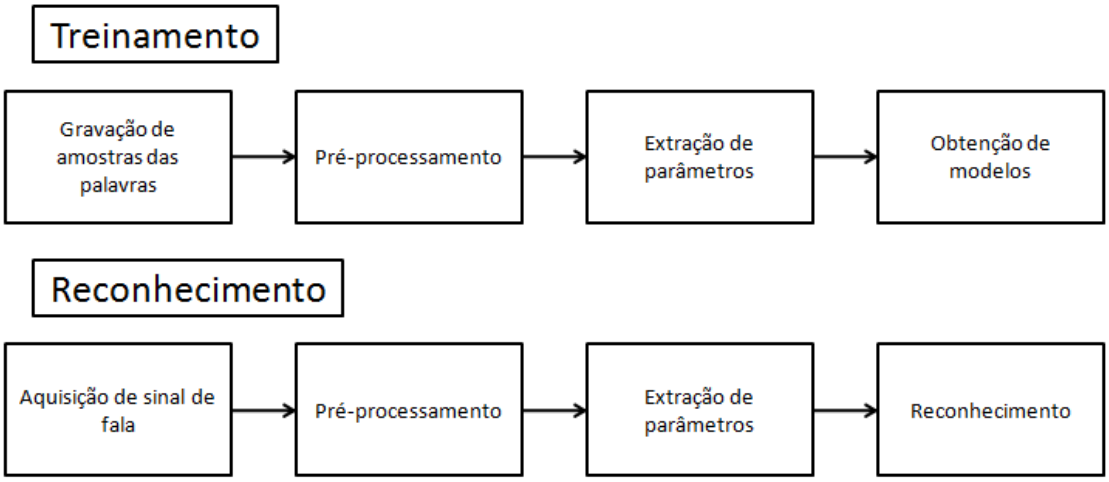


**Figura 4 - Representação em blocos do sistema de reconhecimento**

Como pode ser observado na Figura 4, existem estágios comuns aos dois modos de operação: o Pré-processamento e a Extração de parâmetros.

### 3.1.1. Pré-processamento

O estágio de pré-processamento é responsável por eliminar elementos, presentes nos sinais obtidos, que são indesejáveis e podem atrapalhar a tarefa de reconhecimento.

Os sinais adquiridos costumam possuir uma componente contínua que atrapalha a comparação em valores absolutos. Portanto, a remoção dessa componente DC é importante para melhores resultados.

Outro fator importante é o volume da voz captada pelo sistema. A amplitude dos sinais adquiridos é a característica que representa o volume com que a palavra foi proferida. Assim, para garantir que todas as palavras sejam processadas dentro de uma mesma faixa de valores de amplitude, é realizada a normalização do sinal. Com isso, a amplitude de todos os sinais tratados pelo sistema estará entre -1 e 1.

Finalmente, ao se adquirir um sinal de fala, possivelmente ocorrerão momentos de silêncio antes e após a palavra. Esses momentos não são interessantes para o sistema de reconhecimento uma vez que não contêm informação sobre a fala

propriamente dita e sua inclusão apenas aumenta a quantidade de dados a ser processada e aumenta o tempo total despendido pelo sistema para reconhecer uma palavra. O estágio completo do pré-processamento é representado na Figura 5.

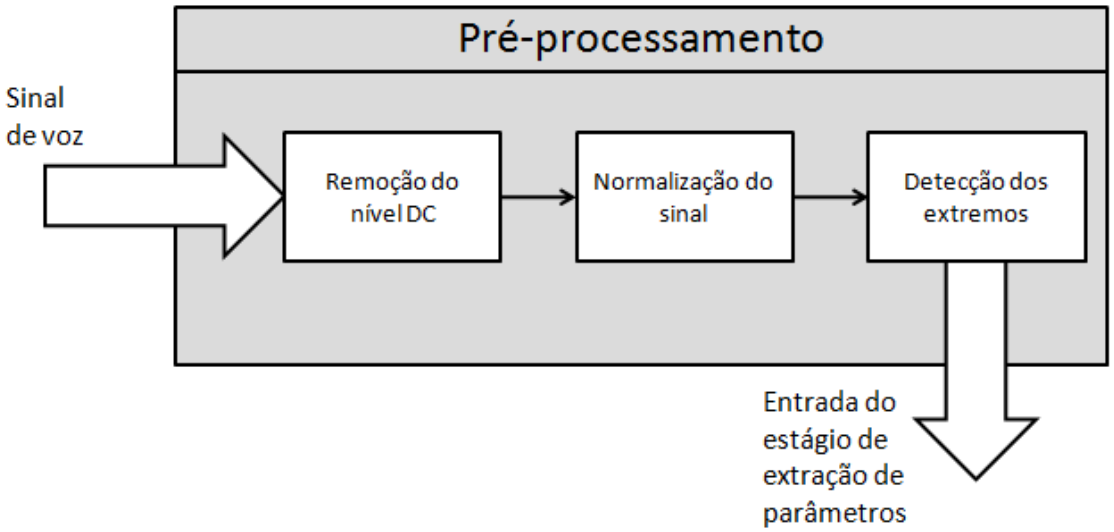


**Figura 5 - Esquema do estágio de pré-processamento**

### 3.1.2. Extração de parâmetros

Um sinal de fala apresenta uma grande quantidade de dados e sua análise direta exigiria muita capacidade e tempo de processamento. Muitos dados nos sinais de fala são redundantes e, portanto, com pouca importância na distinção das palavras. Com a extração de parâmetros, podemos representar unidades de fala com o menor número possível de parâmetros. Concluímos então que o estágio de extração de parâmetros é de grande importância na eficiência de um sistema de reconhecimento.

Neste trabalho, decidiu-se utilizar um modelo baseado na análise cepstral. Assim, os parâmetros usados para classificação e identificação das palavras foram os coeficientes *Mel-cepstrais*. Apesar de a literatura sobre o assunto propor o uso de coeficientes de energia, após experimentos com o sistema já desenvolvido, notou-se que a energia do sinal não influenciava os resultados.

O sinal pré-processado é dividido em segmentos de 25 milissegundos e de cada um desses segmentos são extraídos 13 coeficientes mel-cepstrais. Além dos coeficientes mel-cepstrais, mais 13 coeficientes (derivadas dos coeficientes mel-

cepstrais) chamados delta-cepstrais são utilizados. O conjunto desses 26 coeficientes é armazenado em um vetor. Dessa forma, cada amostra de fala é representada por um conjunto de vetores de tamanho 26. A quantidade de vetores varia para cada elocução de uma mesma palavra, dependendo de sua duração. Esse esquema é representado na Figura 6.

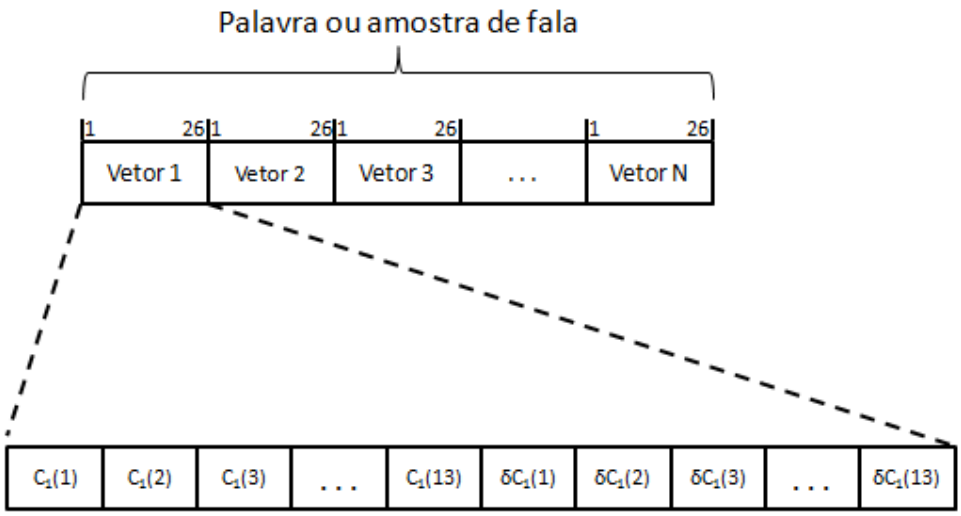


**Figura 6 - Segmentação e extração de parâmetros**

### 3.1.3. Gravação de amostras das palavras

Este é um estágio exclusivo do modo de treinamento. Neste estágio, o usuário grava amostras das palavras que serão utilizadas pelo sistema. Para o sistema desenvolvido para reconhecer os dígitos de 0-9, foram gravados 50 elocuções para cada dígito de um mesmo locutor. Muitas dessas elocuções possuem diferenças de pronúncia para que o sistema seja capaz de reconhecer o dígito independentemente da forma em que este é falado.

Após obter bons resultados com o reconhecimento dos dígitos de 0-9, também foram gravadas 50 elocuções de um mesmo locutor das palavras “Televisão”, “Lâmpada”, “Acionar” e “Desligar”.

Todas essas elocuções gravadas foram obtidas a uma taxa de amostragem de 8000 Hz e depois pré-processadas. Com isso, gerou-se um banco de dados com os sinais gravados. A partir desse banco de dados serão obtidos os coeficientes cepstrais no estágio de extração de parâmetros.

### 3.1.4. Obtenção de modelos

Este estágio é fundamental para o sucesso de um sistema de reconhecimento de voz. É neste estágio que são estimados os modelos HMM's referentes a cada palavra do vocabulário.

Como visto no capítulo 2, um modelo HMM é caracterizado por *N*, o número de estados do modelo; por *M*, o número de símbolos de observação distintos por estado; por *A*, distribuição de probabilidade de transição dos estados; por *B*, distribuição de probabilidade de símbolos de observações nos estados; e por $\pi$, distribuição de estado inicial.

Para determinar o número de estados, não existe uma regra. São necessárias a familiarização com modelos ocultos de Markov e a realização de testes com diferentes valores a fim de encontrar um número ótimo. Após a realização desses testes, o número N de estados escolhido foi 4. O número de observações por estado, M, foi 100. A distribuição de estado inicial foi definida como $\pi$ = [1 0 0 0]. Neste ponto é importante citar que os modelos HMM a serem obtidos são modelos esquerda-direita, ou seja, a medida que o tempo aumenta, o índice do estado aumenta ou permanece o mesmo, procedendo da esquerda para a direita.

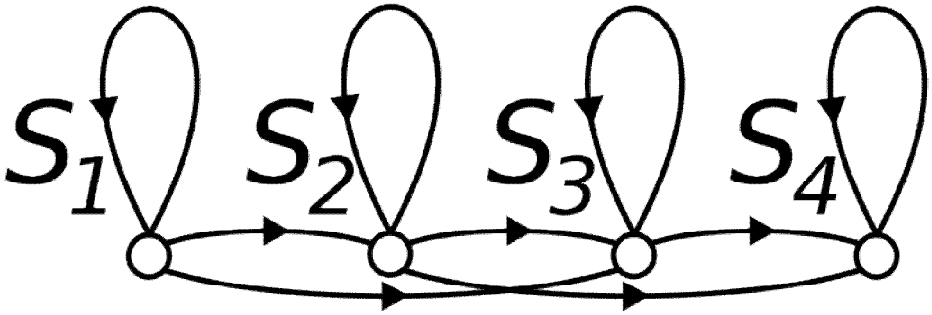


**Figura 7 - Modelo esquerda-direita de quatro estados**

Para a definição das distribuições de probabilidade A e B iniciais, foram usadas rotinas disponíveis em uma toolbox gratuita do MATLAB [2]. Essas rotinas são baseadas no algoritmo de Baum-Welch, descrito no capítulo 2.

Uma vez definidos os parâmetros acima, é usada mais uma rotina desenvolvida na toolbox citada anteriormente para efetuar o treinamento propriamente dito de

todas as palavras do vocabulário. Para cada elocução, é feito o cálculo da semelhança logarítmica de seus dados. Depois desse cálculo, por meio de iterações, é obtido um valor único para a semelhança logarítmica de uma palavra. A rotina foi configurada para que, após, no máximo, 50 iterações, seja definido o modelo HMM de cada uma das palavras do vocabulário.

### 3.1.5. Aquisição do sinal de fala

Ao executar o software de reconhecimento, um loop infinito é iniciado mantendo um canal aberto para captação de sinais de fala. Para que o sistema não capte qualquer ruído e o interprete como fala, foi configurado um valor de limiar que pode ser editado de acordo com o ambiente. Quando mais ruidoso for o ambiente, maior deve ser valor do limiar.

Para efetuar a detecção de um sinal foi utilizado um algoritmo que leva em consideração a energia das amostras do sinal de fala e a taxa de cruzamentos do zero. Esse mesmo algoritmo é utilizado para a determinação dos extremos de um sinal de fala captado. Mais detalhes sobre esse algoritmo pode ser obtido em [3].

Experimentalmente, notou-se que a determinação do início e do fim de uma palavra é de grande importância para o reconhecimento. Inicialmente, a rotina desenvolvida com base no algoritmo, além de suprimir os momentos de silêncio, acabava cortando pequenos pedaços da palavra. Assim, os resultados do reconhecimento não foram satisfatórios. Entretanto, após expandir os limites em poucas amostras, houve um grande aumento na eficiência do sistema.

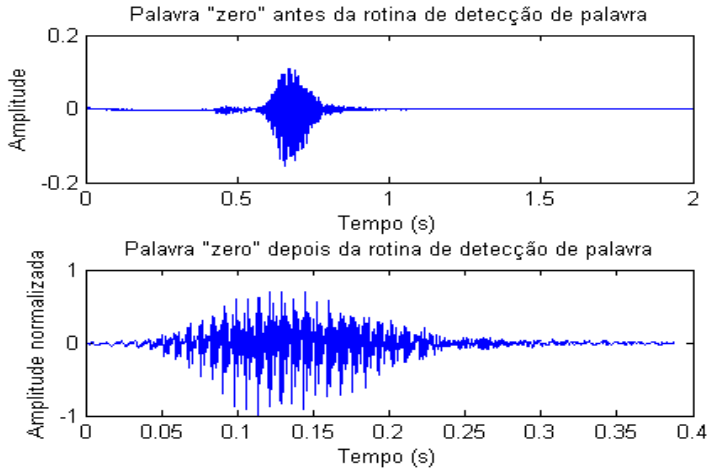


**Figura 8 - Dígito "zero" antes e depois da rotina de detecção do início e fim de uma uteração**

### 3.1.6. Reconhecimento

O estágio de reconhecimento é exclusivo do modo de operação de reconhecimento. Nesta fase, a palavra adquirida no estágio de aquisição passa pelo pré-processamento, tem seus parâmetros extraídos e, a partir desses parâmetros extraídos, é obtida a semelhança logarítmica dos dados e estes são comparados às semelhanças logarítmicas calculadas para os modelos HMM's obtidos no modo de operação de treinamento. O modelo que apresentar maior semelhança quando comparado à semelhança logarítmica da palavra capturada em tempo real é considerado o resultado do reconhecimento. Novamente, neste ponto foi utilizada um rotina já definida pela toolbox gratuita.

## 3.2. Hardware implementado

Como dito anteriormente, o hardware utilizado neste trabalho tem como princípios o recebimento de um comando, a interpretação de um comando e a realização de uma ação referente ao comando interpretado. Resumidamente, o hardware é um controlador de relés interfaciado pela porta USB.

Para isso, o hardware foi dividido em dois estágios: um estágio para o estabelecimento da comunicação por meio da porta USB e outro estágio responsável pela execução do comando.

### 3.2.1. Estabelecimento da comunicação

Neste estágio, o hardware deve determinar como deve ocorrer a comunicação entre ele e o micro-computador. Para a realização desta tarefa, foi utilizado o chipset da FT232BL da FTDI Ltd.. Esse chipset foi escolhido por já conter o protocolo USB embarcado. Outro ponto positivo desse dispositivo é o fato de o fabricante disponibilizar gratuitamente os drivers para o controle do chip nos mais variados sistemas operacionais [4].

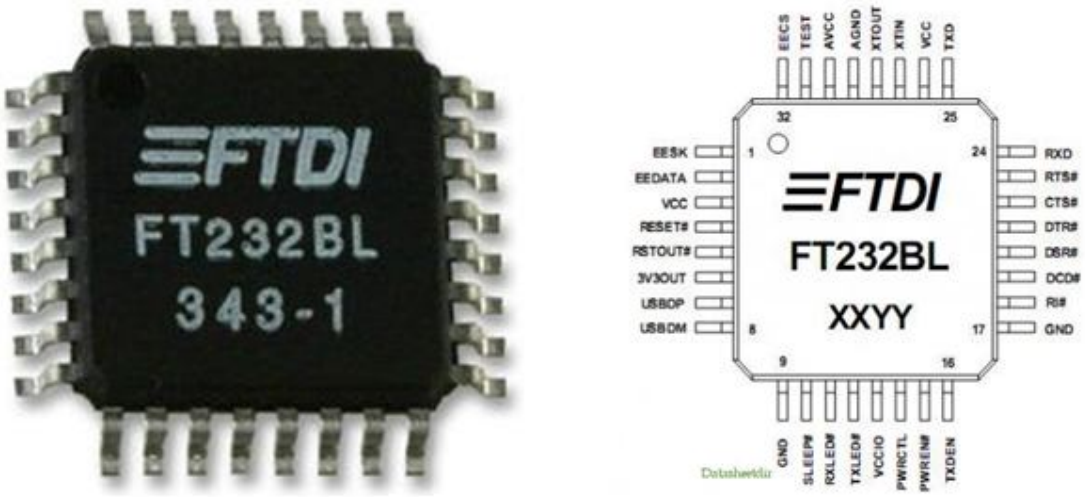


**Figura 9 - Foto e esquema de pinos do chipset FT232BL**

Este chipset é uma ótima solução para a construção de dispositivos que se comuniquem com o barramento USB, podendo atingir uma velocidade de comunicação de até 3 Mbps. Resumidamente, este chipset disponibiliza os dados USB em Serial.Tem como características relevantes ao sistema:

- Um único chip manipula tanto dados USB quanto dados Serial;
- Compatível com barramento USB 1.1 e 2.0;
- Taxa de transferência entre 300 e 3 MBaud em TTL;
- Taxa de transferência entre 300 e 1 MBaud utilizando drivers RS232;
- Taxa de transferência entre 300 e 3 MBaud utilizando drivers RS422 e RS485;
- Tensão de alimentação entre 4,35V a 5,25V;
- Timeout ajustável para buffer RX;

- Suporte a uma memória EEPROM externa opcional para personalização do produto;
- EEPROM pode ser programada pela própria placa USB;
- Suporte para alimentar dispositivos diretamente no barramento USB por meio do pino PWREN#;
- Suporte para os recursos USB (suspend/resume) através dos pinos SLEEP# e RI#.

Neste trabalho, optou-se por realizar o interfaceamento do dispositivo no modo *Bus-powered*, ou seja, esse dispositivo alimentado pelo próprio barramento USB. Essa escolha permite maior mobilidade e conveniência já que não há necessidade de uma fonte de tensão adicional ao sistema. A Figura 10 mostra o esquema do circuito montado.

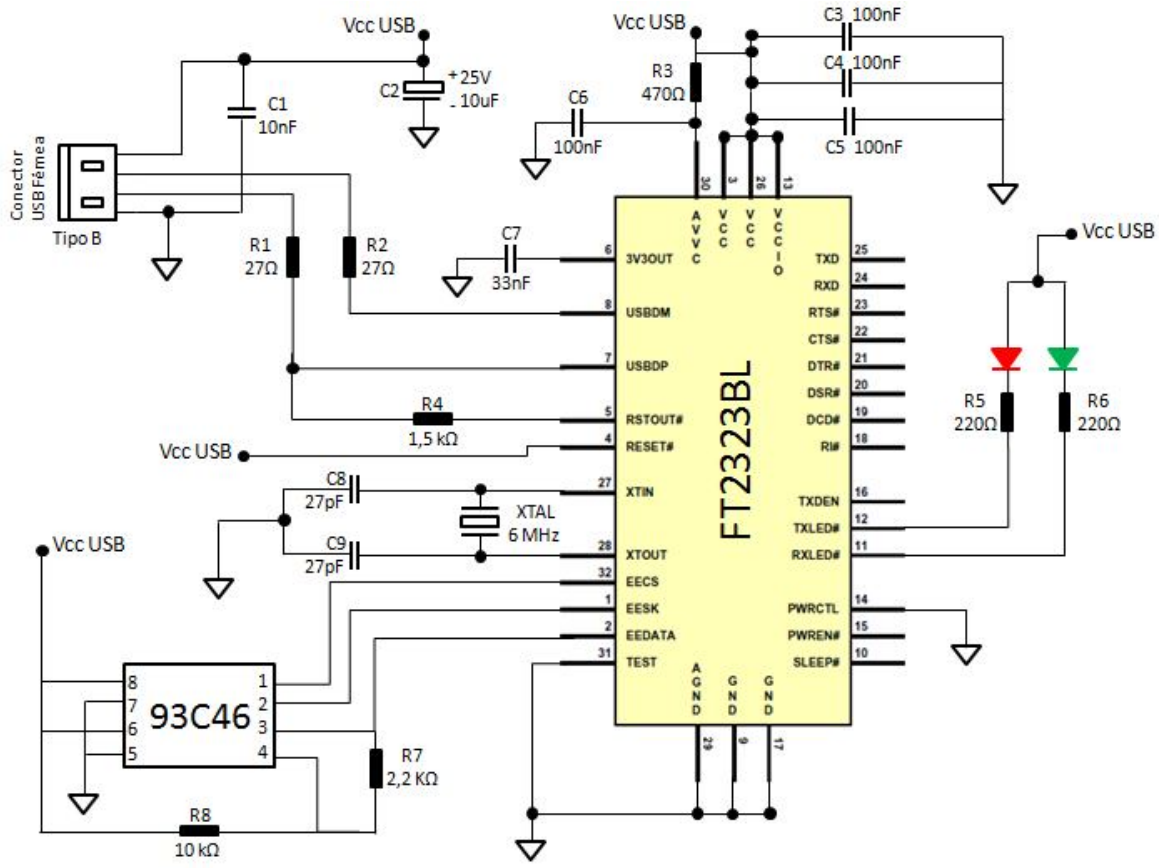


**Figura 10 - Esquema do circuito responsável pela comunicação entre o micro-computador e o hardware**

Desta forma, o circuito apresentado na Figura 10, recebe os comandos pela porta USB e devolve como saída uma String de comando. O CI 93C46 é a memória EEPROM. Apesar de estar presente no circuito, optou-se por não configurá-la. Assim, os dados presentes na memória se referem ao fabricante do chipset FT232BL, no caso a FTDI Ltd.. As tabelas a seguir mostram detalhadamente a função dos pinos divididos em grupos de atuação:

| Pin# | Sinal | Tipo | Descrição |
|---|---|---|---|
| 25 | TXD | Saída | Pino de transmissão |
| 24 | RXD | Entrada | Pino de recepção |
| 23 | RTS# | Saída | Request to send / Handshake |
| 22 | CTS# | Entrada | Clear to send / Handshake |
| 21 | DTR# | Saída | Terminal de dados pronto |
| 20 | DSR# | Entrada | Dados para envio pronto |
| 19 | DCD# | Entrada | Detecta a portadora de dados |
| 18 | RI# | Entrada | Indicador de Ring |
| 16 | TXDEN | Saída | Habilita a transmissão de dados para RS485 |

**Tabela 1 - Grupo de pinos da interface UART (Universal Asynchronous Receiver/Transmitter)**

| Pin# | Sinal | Tipo | Descrição |
|---|---|---|---|
| 7 | USBDP | Entrada/Saída | Sinal positivo de dados (D+) USB. Requer um resistor de pull-up de 1,5k conectado ao pino RSTOUT#. |
| 8 | USBDM | Entrada/Saída | Sinal negativo de dados (D-) USB. |

**Tabela 2 - Grupo de pinos da interface USB**

| Pin# | Sinal | Tipo | Descrição |
|---|---|---|---|
| 32 | EECS | Entrada/Saída | EEPROM-Chip select (seleciona o chip). |
| 1 | EESK | Saída | Sinal de clock para a EEPROM. |
| 2 | EEDATA | Entrada/Saída | Conexão de dados direta com a EEPROM. |

**Tabela 3 - Grupo de pinos da interface da memória EEPROM**

| Pin# | Sinal | Tipo | Descrição |
|---|---|---|---|
| 10 | SLEEP# | Saída | Vai ao nível baixo quando está no modo USB *suspend*. |
| 15 | PWREN# | Saída | Está em nível baixo quando se tem configurado o FT232BM no modo *Buspowered*. Está em nível alto durante o período de suspensão do bus USB. Pode-se usar este pino para controlar a alimentação de dispositivos externos, alimentados diretamente através do bus USB, mediante a utilização de um MOSFET Canal-P. |
| 14 | PWRCTL | Entrada | Em nível baixo, o FT232BM é alimentado através do bus USB (*Buspowered*). Em nível alto é alimentado mediante conexão externa (*Selfpowered*). |

**Tabela 4 - Grupo de pinos de controle de energia**

| Pin# | Sinal | Tipo | Descrição |
|---|---|---|---|
| 4 | RESET# | Entrada | Através deste pino podemos realizar um reset a partir do exterior. Se não for usado, deve ser conectado ao VCC. |
| 5 | RSTOUT# | Saída | Saída do gerador interno de Reset. Este pino não é afetado no caso de um reset no Bus USB. |
| 12 | TXLED# | Saída | LED indicador de transmissão de dados. Este pino quando está em nível baixo indica transmissão de dados. |
| 11 | RXLED# | Saída | LED indicador de recepção de dados. Este pino quando está em nível baixo indica recepção de dados. |
| 27 | XTIN | Entrada | Entrada do oscilador 6 MHz. |
| 28 | XTOUT | Saída | Saída do oscilador 6 MHz. |
| 31 | TEST | Entrada | Põe o chipset no modo teste. Para o funcionamento normal, deve-se concetá-lo ao ground. |

**Tabela 5 - Grupo de pinos de sinais diversos**

| Pin# | Sinal | Tipo | Descrição |
|---|---|---|---|
| 6 | 3V3OUT | Saída | Saída do regulador LDO (*Low Drop Out*) de 3,3V. Este pino deve ser conectado a um capacitor cerâmico de 33nF. Uma pequena quantidade de corrente (menor ou igual a 5mA) pode ser obtida deste pino, para alimentar um circuito a 3.3v se caso necessário. |
| 3,26 | VCC | Alimentação | Tensão de alimentação (+4,35V a +5,25V). |
| 13 | VCCIO | Alimentação | Especifica os níveis de tensão utilizados na interface UART (3.0V - 5,25V). |
| 9,17 | GND | Alimentação | Sinal negativo (massa). |
| 30 | AVCC | Alimentação | VCC analógico para o multiplicador x8 do Clock interno. |
| 29 | AGND | Alimentação | Gnd analógico para o multiplicador x8 do Clock interno. |

**Tabela 6 - Grupo de pinos de alimentação**

### 3.2.2. Execução dos comandos

Nesta etapa, um microcontrolador interpreta sinais e compara as “strings” enviadas pelo circuito responsável pelo estabelecimento da comunicação. É a etapa responsável pelo acionamento dos dispositivos propriamente ditos.

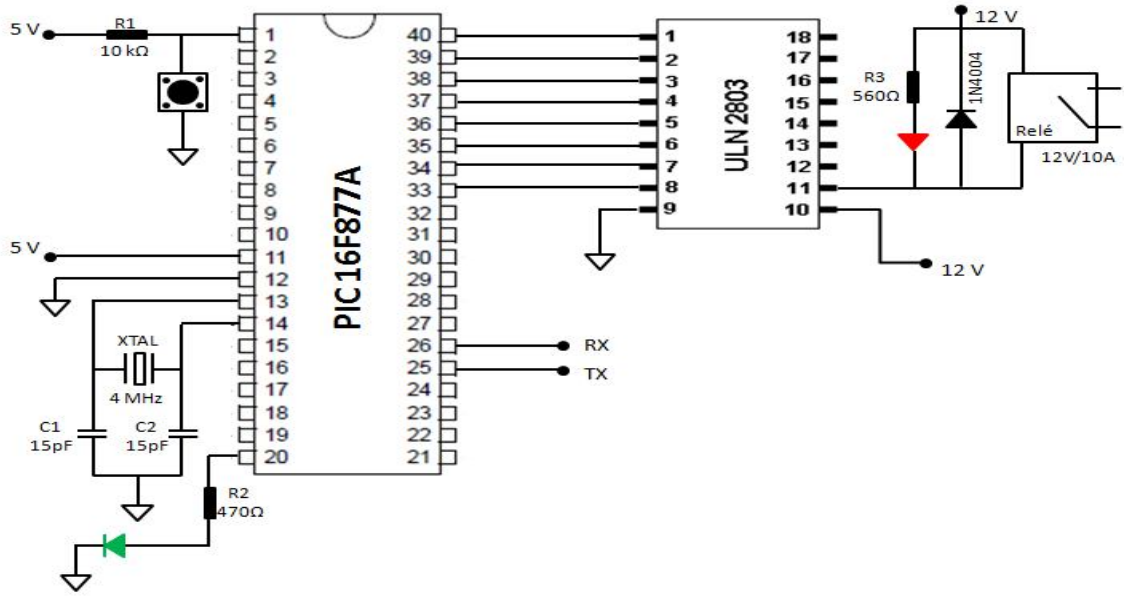


**Figura 11 - Circuito responsável pela execução dos comandos**

O PIC é responsável pela comparação de strings e o driver ULN 2803 é responsável pelo controle dos relés. Este driver possui 8 entradas TTL e oito saídas que podem controlar até 45V/500mA. Neste projeto, uma fonte de alimentação de 12 V é utilizada no controle dos relés. Um relé consome aproximadamente 50mA, possibilitando a ligação de mais 7 relés. Com isso, a corrente total utilizada é de 400mA, deixando uma margem de 100mA. Mais detalhes sobre o PIC 16F877A podem ser encontrados em [5] e sobre o driver ULN 2803 em [6].

# 4. Experimentos e resultados

## 4.1. Software de reconhecimento

Inicialmente foram efetuados testes para o sistema treinado para reconhecer os dígitos de 0-9. O primeiro teste foi efetuado no ambiente em que o sistema foi treinado, mantendo as mesmas condições de ruído e acústica e cada dígito foi pronunciado 100 vezes.

| | | Dígito reconhecido | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | Média (%) |
| Dígito apresentado | 0 | 100 | - | - | - | - | - | - | - | - | - | 100 |
| | 1 | - | 98 | - | - | - | 2 | - | - | - | - | 98 |
| | 2 | - | - | 100 | - | - | - | - | - | - | - | 100 |
| | 3 | - | - | - | 100 | - | - | - | - | - | - | 100 |
| | 4 | - | - | - | - | 100 | - | - | - | - | - | 100 |
| | 5 | - | - | - | - | - | 100 | - | - | - | - | 100 |
| | 6 | - | - | - | 3 | - | - | 97 | - | - | - | 97 |
| | 7 | - | - | - | - | - | - | - | 100 | - | - | 100 |
| | 8 | - | - | - | - | - | - | - | - | 100 | - | 100 |
| | 9 | - | - | - | - | - | - | - | - | - | 100 | 100 |
| | | | | | Média total de sucesso do sistema | | | | | | | 99,5 |

**Tabela 7 - Índice de reconhecimento nas mesmas condições do treinamento**

Também foram realizados testes no ambiente em que o sistema foi treinado, mas ruídos proveniente do ambiente externo e de dispositivos eletrônicos que produzem som acionados próximos ao microfone de captação foram adicionados. Os resultados podem ser observados na Tabela 8.

| | | Dígito reconhecido | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | Média (%) |
| Dígito apresentado | 0 | 96 | - | - | - | 1 | - | - | 3 | - | - | 96 |
| | 1 | - | 85 | - | - | - | 12 | - | - | 3 | - | 85 |
| | 2 | - | - | 98 | - | - | 2 | - | - | - | - | 98 |
| | 3 | - | - | - | 95 | - | 3 | 2 | - | - | - | 95 |
| | 4 | - | - | - | - | 100 | - | - | - | - | - | 100 |
| | 5 | - | - | - | - | - | 100 | - | - | - | - | 100 |
| | 6 | - | - | - | 7 | - | 6 | 87 | - | - | - | 87 |
| | 7 | - | - | - | - | - | - | - | 100 | - | - | 100 |
| | 8 | 1 | - | - | - | - | - | - | - | 99 | - | 99 |
| | 9 | - | - | - | - | - | 1 | - | - | - | 99 | 99 |
| | | | | | Média total de sucesso do sistema | | | | | | | 95,9 |

**Tabela 8 - Índice de reconhecimento em ambiente com ruído**

Os mesmos testes foram realizados para o sistema prático (vocabulário contendo apenas as palavras “Televisão”, “Lâmpada”, “Acionar” e “Televisão”). Os resultados estão presentes nas tabelas a seguir.

| | | Palavra reconhecida | | | | |
|---|---|---|---|---|---|---|
| | | Televisão | Lâmpada | Acionar | Desligar | Média (%) |
| Palavra apresentada | Televisão | 100 | - | - | - | 100 |
| | Lâmpada | - | 100 | - | - | 100 |
| | Acionar | - | - | 100 | - | 100 |
| | Desligar | - | - | - | 100 | 100 |
| | | | Média total de sucesso do sistema | | | 100 |

**Tabela 9 - Índice de reconhecimento do sistema prático nas mesmas condições do treinamento**

| | | Palavra reconhecida | | | | |
|---|---|---|---|---|---|---|
| | | Televisão | Lâmpada | Acionar | Desligar | Média (%) |
| Palavra apresentada | Televisão | 100 | - | - | - | 100 |
| | Lâmpada | - | 100 | - | - | 100 |
| | Acionar | - | - | 100 | - | 100 |
| | Desligar | - | - | - | 100 | 100 |
| | | | Média total de sucesso do sistema | | | 100 |

**Tabela 10 - Índice de reconhecimento do sistema prático em ambiente com ruído**

O sistema de reconhecimento de dígitos de 0-9, por possuir um maior número de palavras em seu vocabulário que o sistema prático, está mais propenso a erros no reconhecimento. Já no mesmo ambiente em que o sistema foi treinado e sem ruído, o sistema apresentou erros de identificação para os dígitos 1 e 6, que apresentaram, respectivamente, taxas de acerto de 98% e 97%. Entretanto, a média de acertos total do sistema foi de 99,5%.

Comparando as Tabelas 7 e 8, nota-se que o ruído interfere na eficiência do sistema. O reconhecimento de dígitos que não haviam apresentado erros de reconhecimento, na presença de ruído, teve seu desempenho degradado. Entretanto, mesmo com a presença de ruído, o sistema apresentou uma média total de 95,9%.

Já o sistema prático, com um menor número de palavras em seu vocabulário, apresentou uma maior robustez quanto ao ruído quando comparado ao sistema de reconhecimento de dígitos de 0-9. Observando as Tabelas 9 e 10, nota-se que a média total do sistema prático é 100%, independente da presença ou ausência de ruído.

## 4.2. Software integrado ao hardware

Os esquemas dos circuitos apresentados na seção 3.2 foram montados e integrados em uma *protoboard* para que fosse possível a realização de testes. Os testes para o sistema seguiram o mesmo padrão dos realizados para o software de

reconhecimento. Cada comando, composto pelo dispositivo (“Televisão” ou “Lâmpada”) e a ação a ser realizada (“Acionar” ou “Desligar”), foi repetido 100 vezes e, para cada repetição, foi verificado se o relé correspondente ao dispositivo em questão realizava a ação proposta. Nesta etapa, foram realizados testes apenas no ambiente de treinamento sem a presença de ruído uma vez que o ruído interfere apenas na identificação de comandos. O envio, interpretação e realização da ação pelo hardware independem da presença ou ausência de ruído no ambiente. Os resultados seguem apontados na Tabela 11.

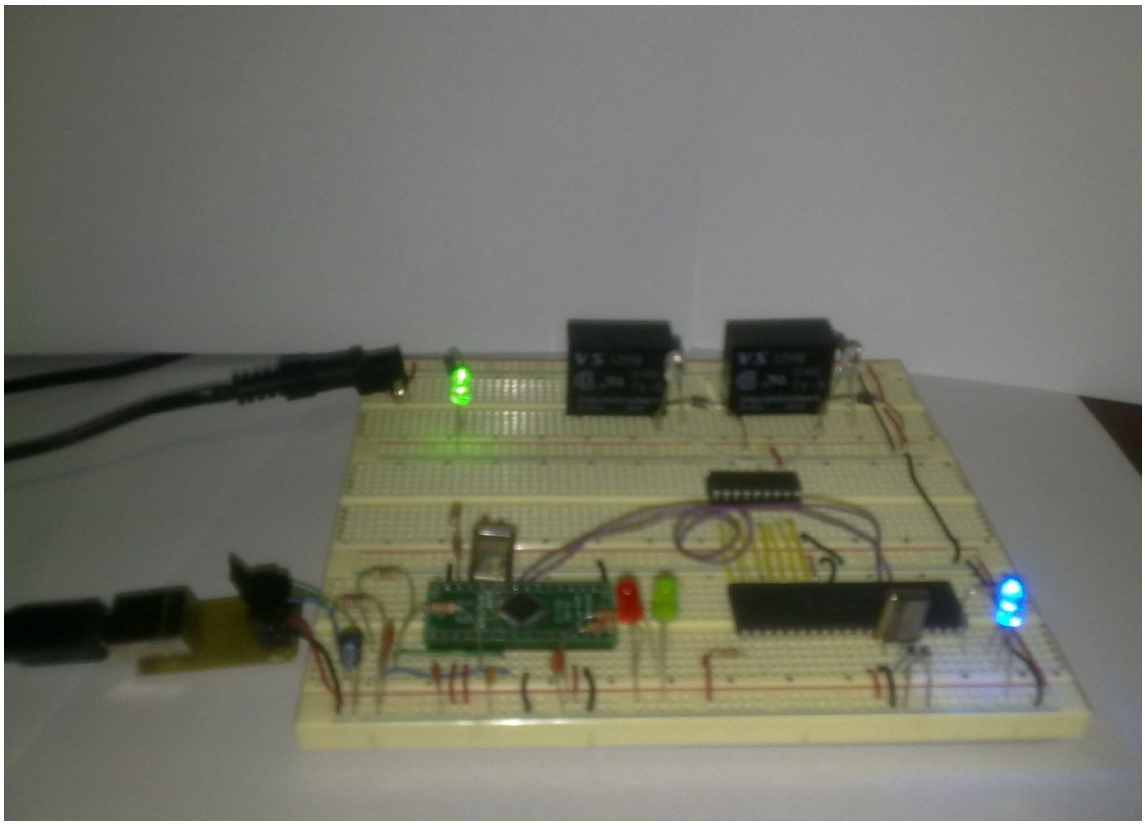

**Figura 12 - Foto do hardware montado em uma *protoboard***

Assim como nos testes para o sistema prático, o sistema integrado (software e hardware) apresentou uma média de sucesso de 100% no ambiente para qual o sistema foi treinado.

| | | Ação | | | |
|---|---|---|---|---|---|
| | | Executada | Não executada | Outro comando executado | Média de sucesso (%) |
| Comando apresentado | Televisão Acionar | 100 | - | - | 100 |
| | Televisão Desligar | 100 | - | - | 100 |
| | Lâmpada Acionar | 100 | - | - | 100 |
| | Lâmpada Desligar | 100 | - | - | 100 |
| | | Média total de sucesso do sistema integrado (%) | | | 100 |

**Tabela 11- Resultados para o teste do software integrado ao hardware no mesmo ambiente de treinamento**

# 5. Conclusão e trabalhos futuros

O trabalho teve como objetivo pesquisar sobre a tecnologia de reconhecimento de voz e, por meio desse estudo, desenvolver um sistema prático de controle de dispositivos através de comandos de voz. A estratégia adotada para o desenvolvimento, além de proporcionar um bom embasamento teórico e prático na área de reconhecimento de voz, possibilitou o desenvolvimento de um sistema inicial para estudos futuros.

Este trabalho teve início apresentando uma introdução à área de reconhecimento de voz, suas aplicações e um breve histórico. Além disso, discorreu sobre etapas do sistema com o intuito de otimizar o processo de reconhecimento.

Tratou também de conceitos sobre modelos ocultos de Markov (HMM), discutindo os seus elementos, os problemas básicos e os algoritmos responsáveis pela resolução desses problemas.

Ainda foram discutidos aspectos sobre o barramento USB, utilizado neste trabalho para realizar a interface entre o software de reconhecimento e o hardware responsável pelo controle dos dispositivos.

Foi apresentado o desenvolvimento de um sistema em MATLAB capaz de reconhecer palavras isoladas dependente do locutor utilizando HMM's discretos responsável por enviar comandos a um hardware de controle.

Por fim, foram apresentados resultados que mostram a eficiência do sistema capaz de reconhecer os dígitos de 0-9 em ambiente sem e com ruído e resultados que mostram a eficiência do sistema capaz de reconhecer as palavras "Televisão", "Lâmpada", "Acionar" e "Desligar".

Analisando os resultados obtidos e o desempenho do hardware, podem ser feitas algumas considerações:

- O sistema de reconhecimento de dígitos apresentou, quando na ausência de ruído, uma alta taxa de acertos (99,5%). Entretanto, os erros ficaram concentrados em apenas dois desses dígitos: 1 e 6. O dígito 1 foi reconhecido duas vezes como 5 e o dígito 6 três vezes como 3. Uma solução para os erros apresentados nessa situação é a implementação de um pós-processamento

que poderia, por exemplo, utilizar a taxa de cruzamentos por zero para fazer resolver possíveis equívocos.

- O algoritmo de detecção de extremos foi um fator fundamental na eficiência do sistema. Utilizando apenas o algoritmo de Rabiner e Sambur [3], alguns detalhes, principalmente no fim das palavras, eram perdidos. Muitos erros estavam ligados a um clique existente no final da palavra que acontece no fechamento dos lábios quando se encerra a pronúncia da palavra.
- Outro importante fator que pode ser considerado é a eficiência do sistema em ambientes diferentes daqueles para o qual foi treinado. A acústica do ambiente é fundamental para o sistema e, portanto, este deve ser usado no local onde as amostras para o treinamento foram obtidas.
- Os dados foram transferidos pela porta USB e o controle dos dispositivos ocorreu conforme esperado. Apesar de, no sistema prático, terem sido utilizados duas linguagens e aplicativos diferentes para o reconhecimento e o envio de dados pela porta USB, a tarefa é feita com rapidez e eficiência.

O sistema como um todo apresentou resultados satisfatórios. Este realizou as tarefas a que se propôs com uma alta taxa de eficiência. Entretanto, algumas ações podem ser tomadas para melhorar ainda mais o sistema. A seguir, temos algumas dessas ações:

- Na implementação de um sistema usual, usar uma linguagem de um sistema não fechado como o MATLAB. O sistema atual só pode ser usado por usuários que possuírem a ferramenta.
- Utilizar outras técnicas de pré-processamento e extração de parâmetros.
- Para o reconhecimento das palavras, foram usadas apenas componentes relacionadas à freqüência. Poderiam ser usados coeficientes temporais para melhorar a taxa de reconhecimento.
- Implementar uma rotina de pós-processamento para a resolução de conflitos.
- Em vez de reconhecer palavras, desenvolver sistema que passe a tratar fonemas.

Portanto, de acordo com o que foi mostrado, este trabalho apresenta um estudo básico da tecnologia de reconhecimento de voz aplicada a uma situação prá-

tica próxima do real. Para a utilização deste sistema em uma escala maior e por mais usuários, são necessárias mais pesquisas e alterações.

## Referências


[1] Rabiner, L. R., Juang B. H. Fundamentals of speech recognition. Prentice Hall, 1993.

[2] Hidden Markov Model (HMM) Toolbox for MATLAB disponível em http://www.cs.ubc.ca/~murphyk/Software/HMM/hmm.html

[3] Rabiner, L. R., Sambur, M. R. An algorithm for determing the endpoints of isolated utterances, Bell System Technical Journal, vol. 54, pp. 297-315, 1975.

[4] Sítio da FTDI Ltd. com os drivers para os sistemas operacionais: http://www.ftdichip.com/FTDrivers.htm

[5] Datasheet do microcontrolador PIC16F877A da Microship disponível em: http://ww1.microchip.com/downloads/en/devicedoc/39582b.pdf

[6] Datasheet do ULN 2803 disponível em: http://www.rentron.com/Files/uln2803.pdf

[7] Davis, K. H., Biddulph, R., Balashek, S. Automatic recognition of spoken digits. The Journal of the Acoustical Society of America, 1952.

[8] Rabiner, L. R., Juang, B. H. Automatic speech recognition – a brief history of the technology development, Elsevier Encyclopedia of Language and Linguistics, 2005.

[9] Lathi, B. P. Modern Digital and Analog Communication Systems, 3. ed. New York: Oxford University Press, 1998, 253 p.

[10] Rabiner, L. R., A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition Proceedings of th IEEE, Vol. 77, nº2, 1989.

[11] Gonçalves, J. V. Estudo e implementação de um sistema de reconhecimento de dígitos conectados usando HMM contínuos. Tese de Mestrado. UNICAMP, SP, 2005.