UNIVERSIDADE FEDERAL DE SERGIPE
CAMPUS UNIVERSITÁRIO PROF. ALBERTO CARVALHO
DEPARTAMENTO DE SISTEMAS DE INFORMAÇÃO (DSI)

GUILHERME BRUNO VITURINO ALVES

SINTRALIB – Sistema Integrado Tradutor de LIBRAS: Comunicação para todos

ITABAIANA
MAIO DE 2017

UNIVERSIDADE FEDERAL DE SERGIPE
CAMPUS UNIVERSITÁRIO PROF. ALBERTO CARVALHO
DEPARTAMENTO DE SISTEMAS DE INFORMAÇÃO (DSI)

GUILHERME BRUNO VITURINO ALVES

SINTRALIB – Sistema Integrado Tradutor de LIBRAS: Comunicação para todos

Trabalho de Conclusão de Curso submetido ao Departamento de Sistemas de Informação da Universidade Federal de Sergipe como requisito parcial para a obtenção do título de Bacharel em Sistemas de Informação.

Orientadora: Profª. Ma. Mai-ly Vanessa Almeida Saucedo Faro
Coorientador: Prof. Dr. Alcides Xavier Benicasa

ITABAIANA
MAIO DE 2017

Bruno Viturino Alves, Guilherme.

SINTRALIB – Sistema Integrado Tradutor de LIBRAS: Comunicação para todos / Guilherme Bruno Viturino Alves – Itabaiana: UFS, 2017.
96f.

Trabalho de Conclusão de Curso em Bacharel em Sistemas de Informação – Universidade Federal de Sergipe, Curso de Sistemas de Informação, 2017.

1. LIBRAS. 2. Inteligência Artificial. 3. Sistemas de Informação. 4. Tradutor

GUILHERME BRUNO VITURINO ALVES

SINTRALIB - COMUNICAÇÃO PARA TODOS

Trabalho de Conclusão de Curso submetido ao Departamento de Sistemas de Informação da Universidade Federal de Sergipe (DSI-ITA/UFS) como parte dos requisitos para obtenção do grau de Bacharel em Sistemas de Informação.

Itabaiana, 29 de maio de 2017.

BANCA EXAMINADORA:

_____

Prof(a) Mai-ly Vanessa Almeida Saucedo Faro, Mestre.
Orientador
DSI-ITA/UFS

_____

Prof(a) Alcides Xavier Benicasa, Doutor.
Coorientador
DSI-ITA/UFS

_____

Prof(a) Methanias Colaço Rodrigues Júnior, Doutor.
DSI-ITA/UFS

A minha mãe e meus avós maternos que me deram todo apoio necessário para a conquista deste objetivo.

AGRADECIMENTOS

Parece até que foi ontem o dia em que prestei vestibular para o curso de Sistemas de Informação do Campus de Itabaiana. Não tinha noção de quão grandiosa seria aquela escolha tomada numa tarde de estágio no Ministério da Fazenda em Sergipe.

Desde já, agradeço de coração a todos que contribuíram de forma direta ou indireta na minha formação.

Aos meus avós, Adelson Viturino e dona Therezinha, grandes responsáveis por este acontecimento. Espelho-me em vocês, minha fortaleza. Esta conquista é a minha retribuição pela preocupação que vocês sempre mantiveram com a minha educação, meus amores.

À minha mãe, Rosângela Viturino, mulher guerreira que criou bem os seus três filhos com todo esforço para nos presentear com o melhor que tinha a oferecer. Com bastante esforço comprou o nosso primeiro computador, no qual obtive o meu primeiro contato íntimo com a computação. Eu a amo muito. Meu amor por ti é incondicional!!!

Ao meu pai, Adelmo Alves, que por muitas vezes distante, ainda assim contribuiu em vários momentos felizes da minha vida. Lembro-me bem dos dias em que saíamos para alugar fitas de vídeo game, muitas vezes na chuva, somente para me ver feliz. Te amo pai!

Aos meus irmãos, João Paulo e João Victor (bastante chatos, mas amo muito!), por existirem e serem os meus eternos amigos.

À minha noiva, Kerolay Nikson, que aturou (e aturará) a minha pessoa durante todo esse tempo. Saiba que sou bastante grato por todos os momentos contigo. Obrigado por me escutar e contribuir com as minhas loucuras. Grande parte deste projeto eu devo a você. A sua pessoa me fez enxergar, aprender que eu poderia ir além. Jamais me arrependerei de ter mudado de escola e te conhecido. Te Amo!!!

Ao meu notebook, Sony Vaio VPCE-B4B4E, companheiro de muitas noites de aprendizado e insônia. Sou bastante grato a você por me acompanhar do curso técnico até o meio da graduação.

Aos meus tios, José Messias, Maria do Carmo, José Carlos, Marta, Selson “SKOL”, Gilson e Rosemary. Em especial, à minha tia Rosemary, que me ensinou a ler e escrever antes de eu entrar na escola e sempre cuidou de mim. Não poderia me esquecer de que foi a senhora quem me deu o meu primeiro vídeo game, tornando-se a principal responsável por eu gostar de tecnologia.

Aos meus primos, Anderson, Andreia, Adrianinha, Alison, Elaine, Renata, Neilma, Clayton, Lucas, Ana Carla, Amanda, Lília, Manuela, Diego, Tatiane, Cibely, Ariane, Beatriz, Gabriel, Américo e Francielly, com um salve bem grande a minha prima Adrianinha, kkkkkkk!!! Não poderia deixar de citar Lurdinha, que sempre me apoiou, nem também deixar de falar do gordinho Wanderson, que me "enjoa" pedindo para desenvolver jogos o tempo todo.

À minha orientadora, professora Mai-Ly Faro. Sua contribuição e ensinamentos foram primordiais para o meu crescimento. Muito obrigado por ter me acolhido e confiado em mim como seu orientando. Sempre disponível na medida do possível, esclarecedora e companheira. Não poderia ter escolhido pessoa melhor para me orientar, sem a senhora este projeto não teria iniciado.

Ao grandiosíssimo prof. Alcides Benicasa. Sou seu fã de carteirinha. O conhecimento adquirido com as suas aulas e metodologia foi fundamental para execução deste trabalho. Fico totalmente honrado em tê-lo como coorientador deste trabalho. Muito obrigado pela preocupação, atenção, esclarecimento de dúvidas e ideias para o projeto SINTRALIB!

Ao meu grande amigo Adilton Sales. Sinceramente não sei nem como te agradecer bicho. Seu companheirismo e amizade valem mais que ouro. Você é surreal, faltam atributos qualitativos para te desenhar. Muito obrigado pelos momentos confortáveis com a sua pessoa, por toda ajuda no departamento, pelos momentos de desabafo e ajudas pessoais. E claro, não poderia faltar, obrigado por ser meu companheiro de cerveja.

Ao professor André Vinicius Nascimento! Ser seu aluno é muito gratificante. Ter participado do Prodap sob a sua coordenação foi umas das coisas mais gratificantes em termo de aprendizado profissional que eu poderia obter nesta graduação.

Ao professor Eugenio Rubens pelo aprendizado e passeios de moto. O senhor é show!!!

Ao professor Methanias Colaço. Infelizmente, não tive mais que uma oportunidade de tê-lo como professor. Sua aula é magnifica, sensacional e espero ter outras oportunidades para absorver mais conhecimentos do "monstro" Methanias Colaço.

A todos os professores e técnicos do DSI por todo conhecimento e preparação para a vida. Digo com toda certeza do mundo, NÃO EXISTE CORPO DOCENTE MELHOR!

A Brenno Santana e Plinio Cardozo por terem entrado comigo no curso Técnico em Informática da Pio X. Fico lisonjeado com a amizade de vocês, são meus irmãos do peito.

Brenno Santana, muito obrigado por me auxiliar até o final do curso técnico e no decorrer da minha graduação. Plinio Cardozo (IRMÃO SAFADO!!!), mesmo querendo me deixar dormindo no ônibus no dia do vestibular (Sei que não faria isso, ou faria? Kkkkk), agradeço-lhe por ter entrado na minha vida e ser leal a mim sempre. Contem comigo para tudo, seus sacanas!!!

Ao pessoal da república, Daniel Augusto, Tarcísio Oliveira, Fabrício Oliveira, Wallas Moura, Humberto pinhão e Jovino Filho. Não consigo descrever a experiência que tive ao lado de vocês. Com toda certeza, foi uma das melhores experiências da minha vida. Lembro-me dos ensinamentos, da comida, conversas, estudo, diversão, brigas e várias coisas legais que só uma família como vocês poderiam proporcionar. Espero encontrá-los em breve, e que nos reunamos para tomar aquela velha cerveja, jogar o velho *poker* e ter aquela velha e boa conversa. Também ao grande amigo Robson Santa Rosa (*In Memory*), sem palavras...!!!

Aos amigos companheiros de casa, Ednilson Messias e Felipe Santos. O que aprendi com vocês não tem tamanho, Ótimos foram os dias que passei ao lado de vocês, nossa amizade será mantida para sempre.

Ao pessoal da Itatech Jr. Em especial, Nathanael Vasconcelos, Breno Santana, Éuder Costa, Ednilson Castro, Felipe Santos, Kaic Barros, Daniel Lima, Amanda Rezende, Lucielma Andrade, Anderson Farias, Everton Lima e Leomir Silva. Vocês representam bastante na minha vida acadêmica.

À minha sogra e cunhadas (Imperatriz, Fernanda, Kerollyn e Karoline) por fazerem meu marketing na área (Bruno 10D, eu não esqueço kkkk).

A Maria e Toinho por terem me levado para fazer o vestibular em Itabaiana, por me acolherem como um sobrinho e me presentearem com meu grande amigo/irmão Plinio Cardozo!

Ao pessoal do Ministério da Fazenda em Sergipe (Wanderlan, Marcinha, Anderson, Paulinho, seu Isaías, dona Carmem, dona Mara, dona Cleide, Francisco, Torres e seu Edelson) pelo início da minha formação profissional. Vocês foram de suma importância no meu aprendizado. Um salve especial para dona Carmem, que me ajudou muito em conselhos e na conciliação do estágio com a escola.

A todos os amigos do ensino fundamental e médio (Bruno Santana, Albert, Erica, Naisa, Felipe Rato, Gildo, Luan Porco, Felipe Aquino, Jean, Luizinho, Glauber, boneco, Monise, Plinio, Michel, etc.) pela amizade, momentos de descontração e aprendizado.

Aos professores do ensino médio e fundamental, em especial a turma da Mágica do Saber e Pio X, Magno, Lana, Katia, Mary, Lídia, Candido, Formiga, Wel, Dorinha,

Michel, Odara, Reginaldo, etc., por terem me lançado para o mundo. Vocês foram a base de tudo que sou e merecem ser lembrados a todo momento. Não poderia me esquecer de Wilton (*in memory*). Obrigado por tudo! Eterno coordenador.

Aos professores do curso Técnico em Informática da Pio X, Marcio Araújo, Ariel, Sandro e Adriana Fontes. Vocês foram fundamentais na minha base para esta graduação. Um salve também para a coordenadora pedagógica, tia Edna, que sempre foi amorosa e sempre nos ajudava da melhor forma.

Espero não ter esquecido de ninguém. Desculpem-me, se esqueci.

A todos, o meu muito obrigado!

"NUNCA DESISTA NA PRIMEIRA TENTATIVA, A PERSISTENCIA É AMIGA DA CONQUISTA; SE QUISER CHEGAR AONDE A MAIORIA NÃO CHEGA, FAÇA O QUE A MAIORIA NÃO FAZ."

(Bill Gates)

ALVES, Guilherme Bruno Viturino. **SINTRALIB – Sistema Integrado Tradutor de LIBRAS: Comunicação para todos**. 2017. Trabalho de Conclusão de Curso – Curso de Sistemas de Informação, Departamento de Sistemas de Informação, Universidade Federal de Sergipe, Itabaiana, 2017.

RESUMO

Segundo o censo demográfico realizado em 2010, pelo Instituto Brasileiro de Geografia e Estatística (IBGE), cerca de dois milhões de pessoas possuem deficiência auditiva severa, sendo notória a baixa inclusão social desses indivíduos. A lei Nº 10.436, de 24 de abril de 2002, que torna a Linguagem Brasileira de Sinais (LIBRAS) parte integrante dos Parâmetros Curriculares Nacionais (PCNs), amenizou de forma mínima tal problema. Diante destas circunstancias, objetivou-se, com este trabalho, contribuir para a inclusão social dos surdos por meio do desenvolvimento de um sistema computacional, que, utilizando técnicas de Inteligência Artificial (IA), como as Redes Neurais Artificiais (RNAs), efetue a tradução de letras do alfabeto, cujo gesto é estático, da LIBRAS para o português, e também, traduzir letras do alfabeto, expressas oralmente ou via textual, do português para a LIBRAS. Com isso serão abordadas as técnicas e ferramentas utilizadas, em conjunto com os hardwares selecionados para este trabalho.

Palavras-chave: Inteligência Artificial. LIBRAS. Redes Neurais Artificiais e Tradutor.

ABSTRACT

According to the 2010 population census, by the Brazilian Institute of Geography and Statistics (IBGE), about two million people have severe hearing loss. The low social inclusion of these individuals is notorious. The law number 10,436, of April 24, 2002, which makes the Brazilian Sign Language (LSB) an integral part of the National Curricular Parameters (PCNs), minimized this problem. In view of these circumstances, the objective of this work was to contribute to the social inclusion of the deaf through the development of a computer system, using Artificial Intelligence (AI) techniques, such as Artificial Neural Networks (RNAs). Translation of letters of the alphabet, whose gesture is static, from LSB to Portuguese, and also, translate letters of the alphabet, expressed orally or via textual, from Portuguese to LSB. This will address the techniques and tools used, together with the hardware selected for this work.

Key-words: *Artificial intelligence. LSB. Artificial Neural Networks and Translator.*

LISTA DE FIGURAS

LISTA DE FÓRMULAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

AI	*Artificial Intelligence*
API	*Application Programming Interface*
ASL	Língua de Sinais Americana
BSL	Língua de Sinais Britânica
CLI	Interface de Linha de Comando
DF	Distrito Federal
EAP	Estrutura Analítica de Projetos
EI	Extração da Informação
FF	*Feedforward*
FN	*False Negative*
FP	*False Positive*
FPS	Frames Por Segundo
GB	Gigabyte
GUI	Interface Gráfica do Usuário
HTML	*Hyper Text Markup Language*
IA	Inteligência Artificial
IBGE	Instituto Brasileiro de Geografia e Estatística
IBM	*International Business Machines*
IDE	Ambiente de Desenvolvimento Integrado
IHC	Interface Homem-Computador
JSAPI	*Java Speech API*
LED	Ligth Emitter Diode
LIBRAS	Língua Brasileira de Sinais
LSB	*Brazilian Sign Language*
LSF	Língua de Sinais Francesa
ME	Micro Edition
MLP	Redes Neurais de Múltiplas Camadas
MP	Ministério do Planejamento, Desenvolvimento e Gestão
NUI	Interface Natural do Usuário
OMS	Organização Mundial de Saúde
PCN	Parâmetros Curriculares Nacionais
PDA	*Personal Digital Assistant*
PT-BR	português-brasileiro
RI	Recuperação da Informação
RGB	Sistema de Cores Aditivas
RNA	Rede Neural Artificial
RSSF	Rede de Sensores Sem Fio
SO	Sistema Operacional
SPOT	*Small Programmable Object*
SVM	*Support Vector Machine*
UFPB	Universidade Federal da Paraíba
UFS	Universidade Federal de Sergipe
USB	Universal Serial Bus
TB	Terabyte
TME	Tempo Médio de Execução
TN	*True Negative*
TP	*True Positive*

XML *Extensible Markup Language*

SUMÁRIO

1. INTRODUÇÃO

O alto índice de surdez no Brasil chega a ser preocupante. É caracterizada como surdez a perda parcial ou total da audição, os indivíduos acometidos por essa patologia tendem a ser prejudicados no tocante à inserção social. ANDRADE e LEWIS (2007) ratificam essas informações quando dizem que nos últimos anos houve um aumento significativo na quantidade de pessoas com deficiência auditiva, mas que, também, houve uma maior preocupação na inserção delas na vida social.

Segundo o censo demográfico 2010, realizado pelo Instituto Brasileiro de Geografia e Estatística (IBGE), cerca de 5% da população brasileira possui alguma deficiência auditiva. Desse percentual, aproximadamente 1% possui deficiência auditiva severa, o que representa um universo de mais ou menos dois milhões de pessoas (IBGE, 2010). É interessante esclarecer que esses números correspondem às pessoas que não conseguem ouvir de modo algum e às pessoas que, mesmo com aparelho auditivo, possuem grande dificuldade permanente de ouvir. Para realçar essa problemática, pesquisas da Organização Mundial de Saúde (OMS) afirmam que a quantidade de deficientes auditivos tende a aumentar ao longo dos anos (ANDRADE e LEWIS, 2007).

Grande parte dos surdos brasileiros não consegue se comunicar de forma eficiente em português escrito. Entretanto, comunicam-se, fluentemente, em linguagem de sinais. Para esse universo de pessoas, a Língua Brasileira de Sinais (LIBRAS) é considerada a Língua Materna, sendo a aquisição da capacidade de se comunicar em português escrito um desafio semelhante a aprender um segundo idioma (Veloso, 2007). Mesmo entre aqueles deficientes auditivos bilíngues, vê-se que a comunicação em LIBRAS é muito mais compreensível e confortável, como bem atesta Veloso (2007) quando diz que "... LIBRAS como toda língua materna, é a língua que desperta a subjetividade e a capacidade de compreensão do indivíduo".

Apesar da lei nº 10.436, de 24 de abril de 2002, tornar LIBRAS parte integrante dos Parâmetros Curriculares Nacionais – PCNs (PRESIDÊNCIA DA REPUBLICA, 2002), nota-se que essa língua ainda não é muito difundida entre os "falantes" da língua portuguesa, o que causa sérios problemas de interação entre surdos e ouvintes. Isso implica numa possível exclusão social, bem como num provável isolamento comunitário de pessoas com deficiência auditiva, já que ficam restritas a se comunicar apenas com indivíduos que dominam esta

linguagem. Segundo Lacerda (2006), algumas pesquisas desenvolvidas no Brasil e no exterior afirmam que alunos surdos, que já passaram alguns anos em escola não apropriada, possuem um rendimento bem baixo se comparado com alunos ouvintes que possuem capacidades cognitivas semelhantes.

Para facilitar a inclusão social dos deficientes auditivos, esta pesquisa contribuirá, através do desenvolvimento de um sistema computacional, cujo objetivo é a tradução de letras do alfabeto, pertencentes à língua portuguesa, para LIBRAS e a tradução de letras do alfabeto em LIBRAS, cujo gesto é elaborado de forma estática, para o português. Este trabalho é uma versão inicial e, com ele, pretende-se prosseguir com a pesquisa de forma a tentar desenvolver um sistema que permita a tradução de palavras e frases expressas em LIBRAS para o português e do português para LIBRAS.

1.1. Motivação e Justificativa

DAMASCENO et al. (2010) ressalta que as pessoas consideradas diferentes, sejam elas com necessidades especiais ou pertencentes a grupos e culturas diferentes, possuem um histórico de exclusão social e que tais fatores contribuíram para o surgimento de legislações que promovem as condições necessárias para esses grupos se incluírem na sociedade.

Andrade e Lewis (2007) consideram baixos os investimentos destinados a promoção e prevenção da saúde auditiva para a diminuição das perdas auditivas parciais e permanentes. Diante dessas constatações, faz-se necessário maiores contribuições para auxiliar na inclusão social dos deficientes auditivos.

1.2. Objetivos do Trabalho

1.2.1. Objetivo Geral

Contribuir no aumento da inclusão social e diminuição do preconceito em torno da população deficiente auditiva brasileira por meio do desenvolvimento de um software capaz de fazer a tradução LIBRAS-Português e Português-LIBRAS, que no presente trabalho fará a tradução de letras do alfabeto cujos gestos são estáticos.

1.2.2. Objetivos Específicos

- Analisar os hardwares de captura de movimento disponíveis no mercado;
- Estudar o funcionamento do sensor de movimentos escolhido;
- Definir os dados a serem servidos como entrada da RNA;
- Definir e treinar uma RNA capaz de classificar um gesto cujos dados foram capturados pelo sensor de profundidade escolhido;
- Definir e construir uma interface capaz de comunicar ao usuário ouvinte a informação passada pelo deficiênte auditivo;
- Definir e construir uma interface capaz de comunicar ao usuário deficiênte auditivo a informação passada pelo ouvinte;
- Realizar testes para avaliar e validar os resultados obtidos pela ferramenta desenvolvida.

1.3. Organização da Monografia

Este trabalho se estrutura da seguinte forma: no Capítulo 2, são apresentados a revisão bibliográfica e os conceitos necessários para a sua realização. Inicia-se com uma breve explanação a respeito de LIBRAS e, em seguida, vislumbram-se os conceitos das Redes Neurais Artificiais, Coeficientes de Similaridade, Captura de Movimentos, Interface Homem-Computador, Métricas de Avaliação, finalizando com os trabalhos relacionados; o Capítulo 3 apresenta uma visão geral da solução para tradução de LIBRAS para português e de português para LIBRAS. São abordadas informações sobre a plataforma de desenvolvimento e funcionalidades das bibliotecas utilizadas para a realização do trabalho, tais como Java Speech, IBM Via Voice, Neuroph, e também sobre o hardware selecionado (Leap Motion), além de partes importantes do algoritmo implementado; o Capítulo 4 apresenta o teste elaborado, assim como seu objetivo, seleção de participantes e os resultados obtidos a partir do teste realizado; e, finalmente, no Capítulo 5, serão expostas as conclusões sobre o trabalho realizado e os possíveis trabalhos futuros relacionados.

2. REVISÃO BIBLIOGRÁFICA

Este capítulo tem como objetivo explanar os conceitos necessários para o entendimento deste trabalho, principalmente os conceitos relacionados a LIBRAS, Redes Neurais Artificiais, Coeficientes de Similaridade, Captura de Movimentos, Interface Homem-Computador, Métricas de Avaliação e Trabalhos Relacionados.

2.1. LIBRAS

Segundo Carvalho (2007), LIBRAS é utilizada por grande parte dos surdos nos centros urbanos brasileiros. Sua origem vem tanto de uma língua de sinais nativa quanto da gestual francesa. Por isso, há uma grande similaridade com as outras línguas de sinais, tanto da Europa quanto da América (STROBEL e FERNANDES, 1998). Rosa (2005) ainda identifica que a LIBRAS não é composta somente de gestos da língua portuguesa. Há movimentos específicos e naturais dela própria, o que a caracteriza como uma língua a parte.

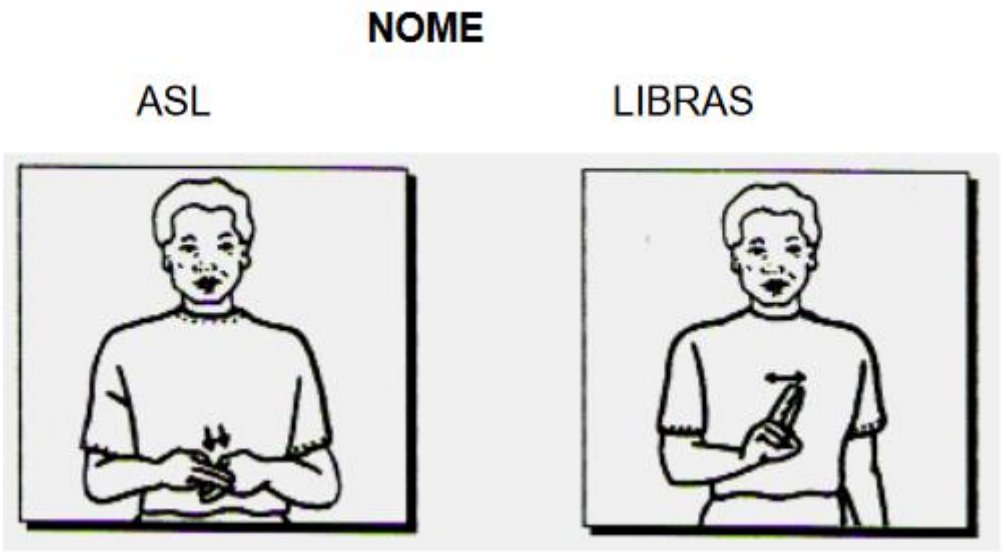
Segundo Damasceno et al. (2010), para se comunicar em LIBRAS, não basta apenas conhecer os sinais, é necessário conhecer a sua gramática para assim combinar os gestos e estabelecer comunicação. Os sinais surgem da combinação de configurações de mão, movimentos e de pontos de articulação, que são locais no espaço ou no corpo onde os sinais são feitos. Existem combinações de expressões faciais e corporais que transmitem os sentimentos, o que para os ouvintes são transmitidos pela entonação da voz, os quais, juntos compõem as unidades básicas dessa língua (ROSA, 2005). Assim, a LIBRAS se apresenta como um sistema linguístico de transmissão de ideias e fatos oriundos de comunidades de pessoas surdas do Brasil.

A composição da LIBRAS em nível linguístico é tão estruturada quanto qualquer outra linguagem, seja ela oral ou gestual. Sua estrutura é composta de fonologia, morfologia, sintaxe e semântica. Nas linguagens de sinais também existem itens lexicais. A diferença cai sobre a forma de articulação, chamada de visual-espacial ou cinético-visual. Com isso, a linguagem gestual tem complexidade e expressividade tão robusta como qualquer outra, pois consegue expressar ideias sutis, complexas e abstratas (ROSA, 2005).

STROBEL e FERNANDES (1998) enfatizam que a LIBRAS é totalmente diferente da Língua de Sinais Americana (ASL), que também difere da Língua de Sinais

Britânica (BSL), Francesa (LSF) e outras linguas de sinais. O mesmo autor afirma haver variações da língua de sinais em diferentes regiões de um mesmo país. A Figura 01 demonstra a palavra “nome” sendo executada na ASL e LIBRAS. Já a Figura 02 permite a visualização da execução da palavra “verde” em diferentes regiões do Brasil.

Figura 01 – Execução da palavra “nome” em ASL e LIBRAS.



Fonte: STROBEL e FERNANDES (1998).

Figura 02 – Diferença do sinal em LIBRAS nas diferentes regiões.

Fonte: STROBEL e FERNANDES (1998).

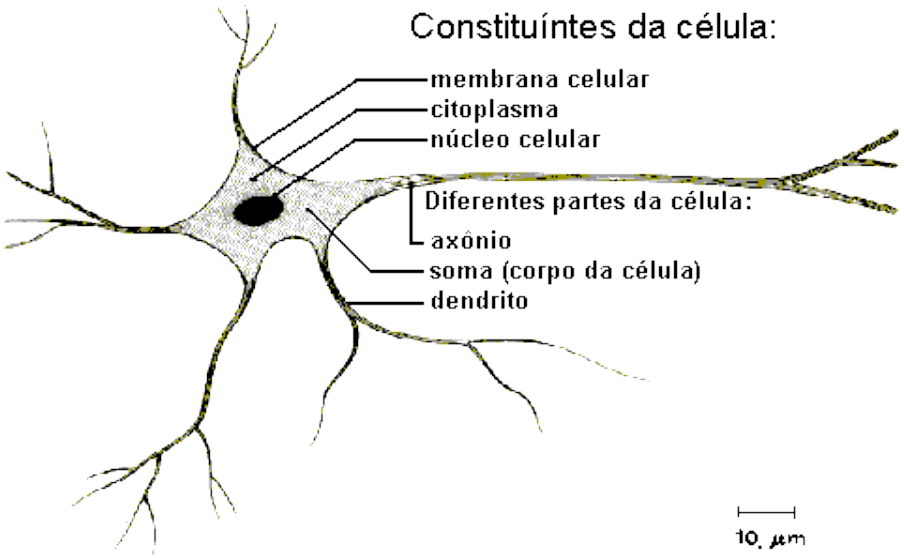
Damasceno et al. (2010) afirma existir diferentes níveis de surdez, os quais são mensurados através da perda auditiva em decibéis (db):

- Surdez leve: Caracterizada pela perda auditiva aos niveis de 25db a 40db;
- Surdez moderada: Pertencem a esse grupo os individuos cuja perda auditiva está entre 40db e 55db;
- Surdez acentuada: Esse nível de surdez é definido para uma perda auditiva entre 56db e 70db;
- Surdez severa: Acompanha pessoas com perda auditiva definida entre 71db e 90db;
- Surdez profunda: Toda perda auditiva acima de 90db;

2.2. Redes Neurais Artificiais

As Redes Neurais Artificiais (RNAs) são sistemas computacionais que apresentam modelos matemáticos inspirados nas estruturas neurais biológicas capazes de aprender através de experiências (BENICASA, 2013). Uma RNA pode chegar a ter centenas ou milhares de unidades de processamento, enquanto o cérebro de um mamífero chega a ter bilhões de neurônios (CARVALHO, 2009). O neurônio artificial pode se encontrar em um estado de atividade ou inatividade. O primeiro irá acontecer quando sua saída ultrapassar um valor dado como limite, caso contrário o mesmo se manterá em repouso (CARDON e MULLER, 1994). A Figura 03 permite a visualização do formato e componentes de um neurônio biológico.

Figura 03 – Visão de um neurônio biológico.

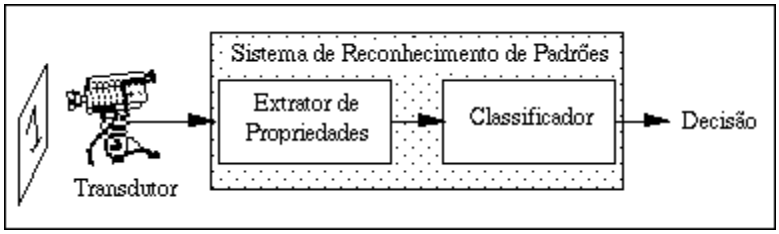


Fonte: CARVALHO, 2009.

As RNAs podem ser um modo de resolução de problemas característicos da Inteligência Artificial – IA (BARRETO, 2002). Modelos de RNAs são utilizados para a resolução de diversos problemas. O próprio sistema biológico pode ser simulado através da neurociência computacional (BENICASA, 2013). Para obter resultados mais precisos no reconhecimento de padrões, previsões, classificações e outros diagnósticos com um alto grau de complexidade, a RNA deverá ser composta por várias unidades de “neurônios”, os quais estão interligados entre si, formando um arranjo complexo de ligação (CAMPOS, 2010).

Segundo Barreto (2002), é bem comum usar o paradigma supervisionado para reconhecimento de padrões com associação direta em uma rede multicamadas. No entanto, o mesmo deixa claro que, se conseguem, também, bons resultados utilizando o aprendizado competitivo, tais quais as redes de Kohonen. A Figura 04 representa um sistema reconhecedor de padrões em que o transdutor é um sensor que envia informações do objeto, após as informações serem filtradas e servidas como entradas RNA. Depois do ajuste de pesos em função de uma saída desejada, tem-se uma decisão para a classificação.

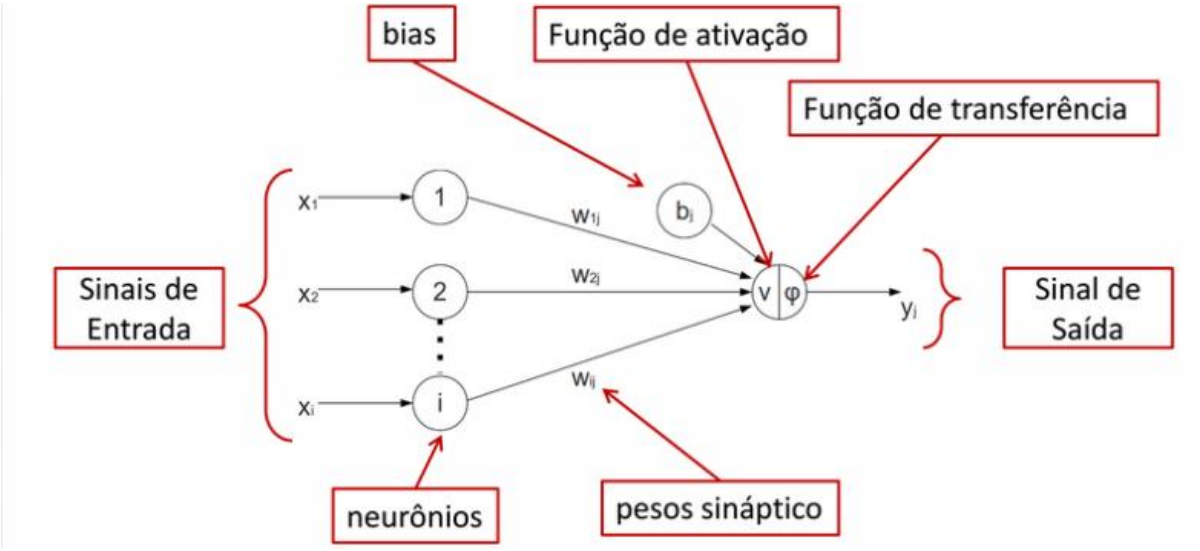
Figura 04 – Visão de um sistema de reconhecimento de padrões.



Fonte: BARRETO, 2002.

O primeiro modelo de RNA implementado foi o *perceptron*, por Frank Rosenblatt em 1958. Apesar de ser simples, a rede *perceptron* consegue representar toda função linearmente separável (CARDON e MULLER, 1994). Suas características envolvem uma camada de entrada e uma de saída. Cada entrada está relacionada a um peso, o valor de saída será o somatório dos produtos de cada entrada pelo peso relacionado, como demonstrado na Figura 05.

Figura 05 – Ilustração de uma Rede Neural *Perceptron*.



Fonte: VOLPI, 2016.

O RNA tem a capacidade de aprender principalmente por exemplos, fazendo referências com o que já aprendeu e sempre atualizando seus pesos, quando excitados (BARRETO, 2002). Os algoritmos de aprendizagem são utilizados para ajustar os valores dos pesos estabelecidos, sendo o *backpropagation* o mais comum entre eles. Contudo, esse algoritmo é bastante sensível nos mínimos locais e requer uma atenção maior na taxa de aprendizado. Para ter uma convergência correta e com o mínimo de falhas, além de bons

exemplos para serem aprendidos, é fundamental dar ênfase, em especial, a inicialização e ajuste de pesos (NIED, 2007).

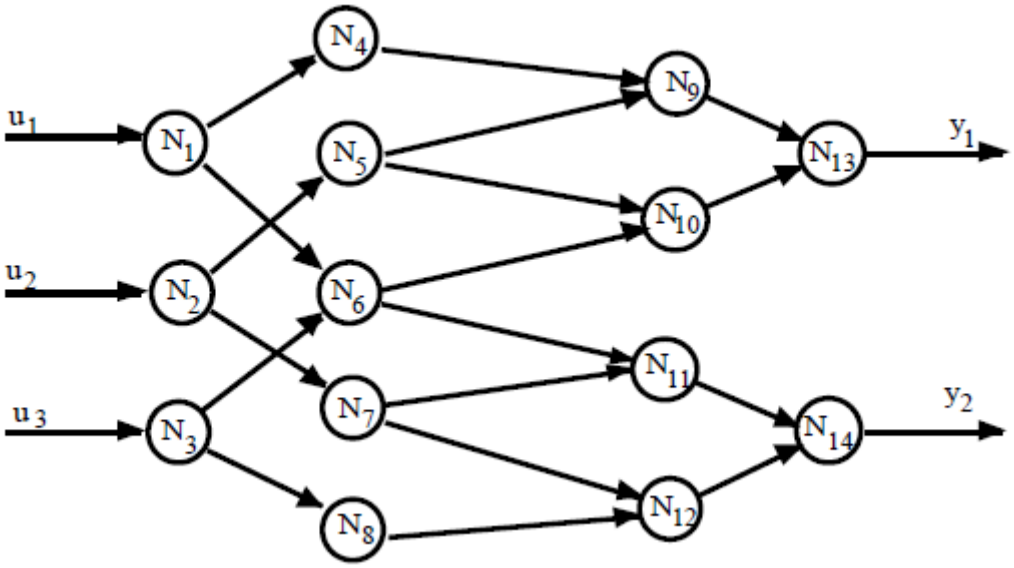
Benicasa (2013) diz que uma RNA pode ser dividida em três grupos distintos correspondentes ao tipo de saída da unidade de processamento simples:

- Saídas binárias: este grupo é formado por neurônios cuja saída produz apenas dados binários.
- Saídas contínuas: os neurônios que compõem este grupo têm suas saídas representadas por funções contínuas. As redes deste grupo podem processar entradas e saídas analógicas.
- Saídas pulsadas: este grupo tem a caracteristica de ser composto por redes neurais pulsadas, cujos neuronios utilizam pulsos como forma de processamento, ao invés de saídas binárias ou analógicas.

2.2.1. *Backpropagation*

É um procedimento de aprendizado para redes *feedforward* (FF) de múltiplas camadas (MLP), realizando ajustes de peso de forma iterativa a fim de encontrar a maior semelhança entre a saída atual e a saída desejada. Esse método foi proposto em 1986 por Rumelhart, Hinton e Willian (TISSOT, CAMARGO e POZO, 2002). Para Barreto (2002), o *backpropagation* é bastante usado, porém ineficiente, mesmo quando utilizam neurônios dinâmicos, é bastante limitado, pois não chega a representar todos os sistemas dinâmicos. A Figura 06 permite a visualização de um MLP.

Figura 06 – Rede neural direta com três camadas de neurônios.



Fonte: BARRETO, 2002.

Carvalho (2009) informa que as redes neurais que utilizam o método *backpropagation* podem ser vistas como "caixa preta" pelo fato da rede não apresentar um motivo pelo qual chegou a determinado resultado.

A função de ativação de uma RNA que utiliza *backpropagation* é do tipo *sigmoid*, pois é necessário o uso de uma função não linear para aumentar a sua capacidade de classificação. Essa função possibilita o aprendizado de representações mais complexas (CARDON e MULLER, 1994). Nesse contexto pode ser determinada pela Fórmula 01, onde S é a saída linear resultante da soma ponderada do nodo i e θ é o coeficiente linear.

$$sgm(S_i) = \frac{1}{1 + e^{-(S_i - \theta)}}$$

Fórmula 01: Fórmula para calcular a função *sigmoid*.

2.3. Coeficiente de Similaridade

Os coeficientes de similaridade baseiam-se em comparar dois objetos, a fim de encontrar características em comum com base em um número total de atributos envolvidos (MEYER, 2002). A princípio, o uso do coeficiente de similaridade se restringia apenas a medidas binárias. Entretanto, com o avanço do setor tecnológico, ele passou a ser utilizado em outros tipos de dados (VALENTIN, 1995).

Algumas fórmulas dos coeficientes de similaridade binários, bastante utilizadas,

são apresentadas na Tabela 1, onde: a é o número total de atributos em que Objeto$_1$ e Objeto$_2$ possuem, ambos, o valor 1, b é o número total de atributos em que Objeto$_1$ possui valor 1 e Objeto$_2$ possui valor 0, e c é o número total de atributos em que Objeto$_1$ possui valor 0 e Objeto$_2$ possui valor 1.

Tabela 1 – Coeficientes de similaridade que desconsideram a ausência conjunta.

Coeficiente	Formula*	Intervalo de ocorrência
Jaccard (1908)	$s = \frac{a}{a+b+c}$	[0,1]
Andeberg (1973)	$s = \frac{a}{a+2\,(b+c)}$	[0,1]
Czekanowsky (1913)	$s = \frac{2a}{2a+b+c}$	[0,1]
Kulenzynski I (1927)	$s = \frac{a}{b+c}$	[0,+∞]
Kulenzynski II (1927)	$s = \frac{a}{2}\left(\frac{1}{a+b} + \frac{1}{a+c}\right)$	[0,1]
Sorensen-Dice (1945)	$s = \frac{2a}{2a+b+c}$	[0,1]
Ochiai (1957)	$s = \frac{a}{\sqrt{(a+b)(a+c)}}$	[0,1]

Fonte: MEYER, 2002.

Uma métrica bastante utilizada é à distância cosseno. Onde se calcula então o cosseno do ângulo formado pelos dois vetores analisados, quanto mais próximo de 1, mais similares são os dois objetos, o contrário vale para quanto mais próximo de 0. À distância cosseno pode ser aplicada conforme demonstrado na Fórmula 02, onde A e B são vetores dimensionais. (FERNANDES, 2014).

$$\frac{\sum_{i=1}^{n}(A_i B_i)}{\sqrt{\sum_{i=1}^{n} {A_i}^2} + \sqrt{\sum_{i=1}^{n} {B_i}^2}}$$

Fórmula 02: Fórmula para calcular a distância cosseno entre dois vetores.

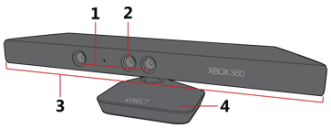
2.4. Captura de Movimento

Com o contínuo avanço tecnológico, a interação homem-máquina ganhou mais uma forma de expressão. A captura de movimentos abre espaço para diversas técnicas de visão computacional. Novos dispositivos foram criados com o intuito de acelerar e possibilitar o desenvolvimento de aplicações cada vez mais robustas nesta área (PIRES, 2012). Como consequência, surgem novos desafios e aplicações.

2.4.1. Sensor Kinect Xbox 360

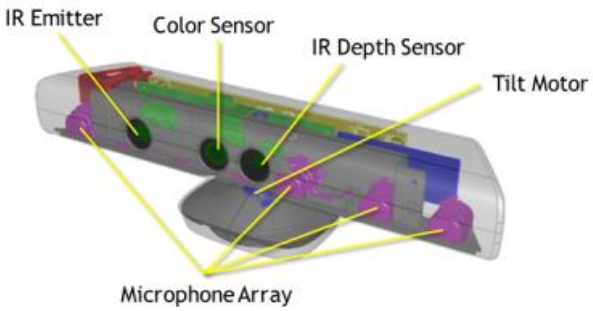
O sensor de profundidade kinect para Xbox 360 foi projetado com o intuito do jogador não precisar de um *joystick* para controlar personagens dentro do *game*. Não demorou muito e o dispositivo começou a ser utilizado em aplicações controladas via interação natural (MICROSOFT, 2016). O sensor kinect é composto por: 1– Sensores de Profundidade 3D, 2– câmera RGB (Sistema de Cores Aditivas), 3– microfones, 4– inclinação mecanizada. As Figuras 07 (A) e 07 (B) permitem a visualização completa dos componentes do kinect *for* Xbox 360.

Figura 07 (A) – Componentes do sensor kinect.



Fonte: MICROSOFT, 2014.

Figura 07 (B) – Imagem 3D do Sensor kinect.

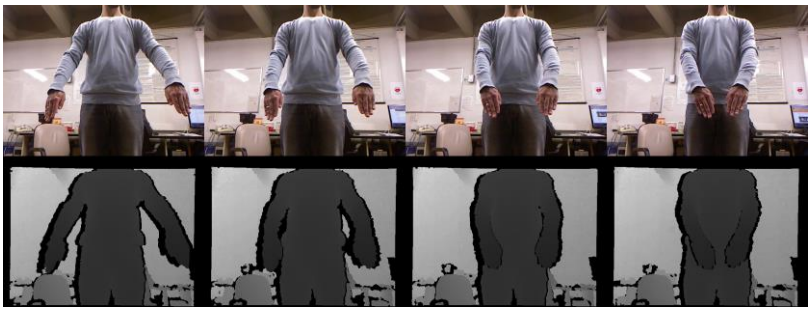


Fonte: MICROSOFT, 2014.

O kinect não consegue processar imagens do tipo textura e profundidade em paralelo, sendo assim, faz-se necessário a captura de todas as imagens referente a um quadro em especifico, para só então iniciar o processamento dos dados. (PIRES, 2012).

A Figura 08 mostra uma visão computacional da captura de movimentos do kinect para Xbox 360. Na parte superior é dada a entrada composta por uma sequência de vídeo. A imagem de textura fornece valores em RGB, na de profundidade é fornecido uma tripla (x,y,z).

Figura 08 – Visão da Captura de movimento do kinect para Xbox 360.



Fonte: PIRES, 2012.

O kinect possui um projetor de luz infravermelho, que em conjunto com a sua câmera especifica para capturar sinais infravermelhos, consegue obter noções de profundidade dos objetos em contato com a luz projetada. Sua eficiência em capturar movimentos é alta, entretanto, deixa bastante a desejar quando precisamos capturar movimentos mais detalhados, tais como os movimentos dos dedos, enxergando apenas o objeto mão, como se fosse um único ponto (MONTEIRO e ANTUNES, 2013).

Segundo Ferreira (2013), o sensor kinect possui duas versões diferentes, na versão feita para uso no console Xbox 360, traz a necessidade de um cabo especial para ser utilizado no computador, já na versão kinect *for* Windows, o mesmo cabo já vem integrado. Apesar dos aparelhos parecerem idênticos, O kinect *for* Windows tem vantagem sobre a versão kinect *for* Xbox 360 quando seu uso é direcionado para aplicações que requerem recursos mais sofisticados. Algumas das suas diferenças são:

- Near mode: habilidade de identificar objetos a uma distância mínima de 40 *cm* sem perder a precisão;
- Seated mode: recurso que possibilita identificar usuários que se encontram na posição sentada, com o kinect *for* Xbox 360 isso não é possivel, pois o mesmo

trabalha com o padrão de 20 pontos por esqueleto, neste caso o kinect *for* Windows reduz seu mapeamento para 10 pontos;

- *Universal Serial Bus* (USB): o kinect *for* Windows possui uma saída USB, caracteristica ausente no kinect *for* Xbox 360;
- Configurações Avançadas de câmera: configurações exclusivas para ajuste de luminosidade, exposição, foco;
- Kinect fusion: mapeamento do ambiente detectado na terceira dimensão;
- Movimento com as mãos: detecção de gestos manuais, como abrir e fechar as mãos, ou pressão sobre objetos.

A Figura 09 permite a visualização das duas versões do kinect, na parte superior da imagem se encontra o kinect *for* Xbox 360, logo abaixo o kinect *for* Windows.

Figura 09 – Visão das duas versões do sensor kinect.

Fonte: Elaborada pelo autor.

Para se capturar os movimentos dos dedos e mãos, é necessário combinar o kinect com alguns frameworks como OpenCV e OpenNI, além de middlewares como o NITE. Esses recursos permitem o reconhecimento de gestos por um conjunto de técnicas de processamento de imagens e análise de séries temporais, determinando o inicio e fim de cada gesto em relação aos quadros de movimentos.

2.4.2. Creative Senz3D

O sensor de profundidade *Creative Senz3D*, é bastante completo quando se trata de interação natural, sua capacidade de capturar imagens, gestos, voz e expressões faciais, possibilitando até reconhecimento facial, o tornam diferenciado (*Intel Corporation*, 2016). A Figura 10 permite a visualização dos componentes do sensor de profundidade *Creative Senz3D*, que possui dois conjuntos de microfones (um em cada extremidade lateral do sensor),

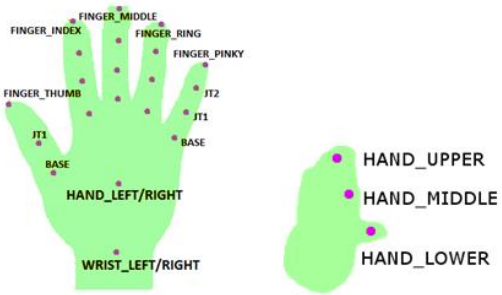
webcam HD 720p, e uma câmera de infravermelho, utilizada para capturar a profundidade dos objetos.

Figura 10 – Componentes do *Creative Senz3D*.

Fonte: DOSS et al., 2013.

A captura e reconhecimento de gestos no *Creative Senz3D*, se torna mais precisa pelo fato do sensor conseguir rastrear as mãos e os dedos, possibilitando assim, o reconhecimento de movimentos mais sensíveis (DOSS et al., 2013). A Figura 11 mostra a visualização gráfica da mão, e a precisão em reconhecer todas as articulações dos dedos e partes da mão.

Figura 11 – Visão computacional do sensor *Creative Senz3D* em função da mão.



Fonte: DOSS et al., 2013.

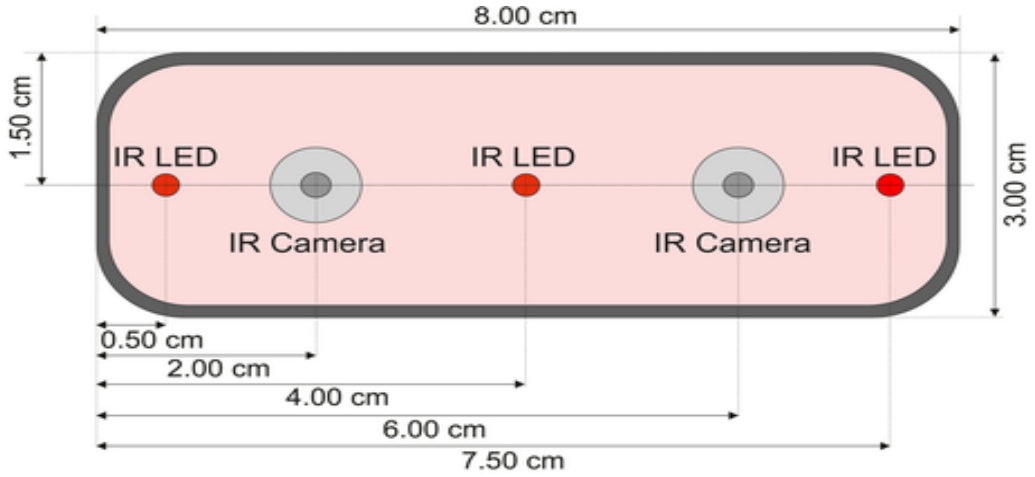
2.4.3. Leap Motion

O controle Leap Motion é um sensor de movimento limitado apenas aos movimentos dos braços e mãos. Consegue ter um alto desempenho no que se propõe, que é capturar movimentos dos membros superiores (LEAP MOTION, 2015). Suas dimensões são de oito por três centímetros (MELO, 2013). A sua velocidade de captura chega a até 200

frames por segundo (fps), o que implica em uma captura de movimentos em tempo real (LEAP MOTION, 2015).

O dispositivo é composto de três LEDs (*Light Emitter Diode)* infravermelhos, e dois sensores de profundidade, o que implica em um registro de imagem rápido, e que permite ao dispositivo ser usado em lugares com baixa luminosidade. A Figura 12 demonstra como é internamente a localização dos componentes do Leap Motion, e as suas dimensões.

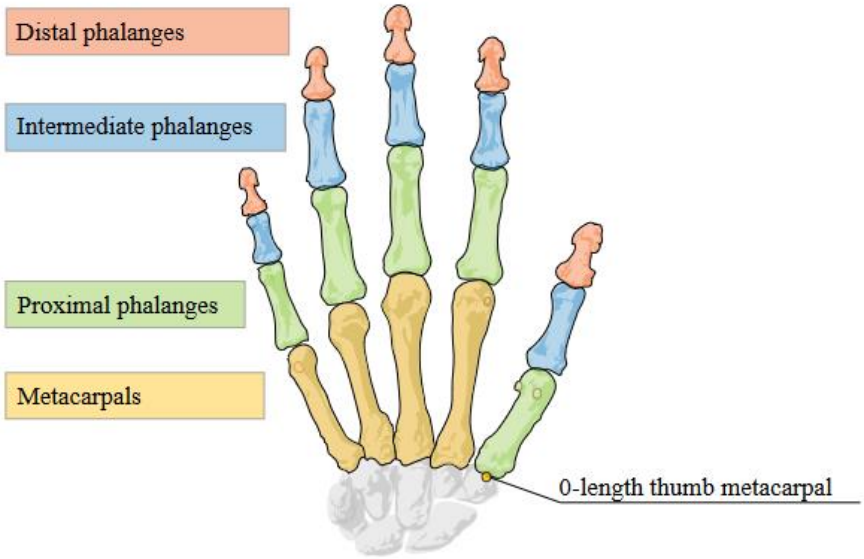
Figura 12 – Visualização Interna do dispositivo Leap Motion.



Fonte: LEAP MOTION, 2015.

Para cada quadro capturado, o dispositivo disponibiliza uma lista de objetos mapeados pelo controlador. Todos os objetos capturados pelos sensores infravermelhos sofrem ajustes para a melhoria da resolução e renderização (MELO, 2013). Uma das características do controlador Leap Motion é conseguir mapear os ossos das mãos e transforma-los em objeto, com isso, permite-se reconhecer gestos mais sofisticados (Leap Motion, 2015). A Figura 13 demonstra como o controlador consegue visualizar uma mão e monta-la como um objeto, que possui uma lista de dedos, e estes, uma lista de ossos, através do modelo *Skeletal tracking*, implementado na sua versão 2.0 da api do Leap Motion.

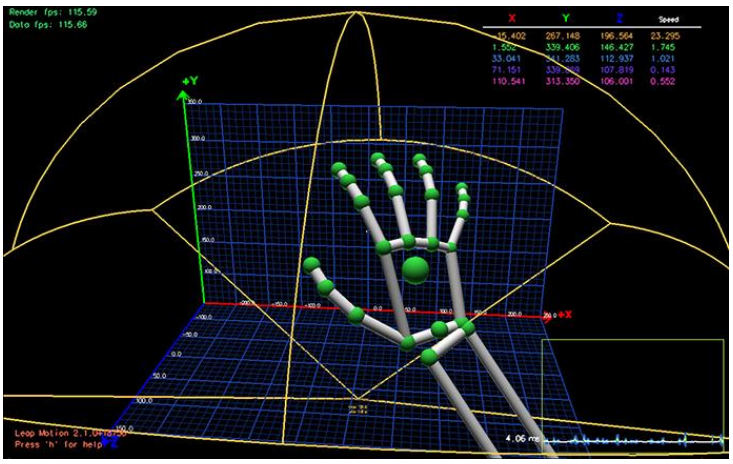
Figura 13 – Mapeamento da mão usado pelo modelo *Skeletal tracking* do Leap Motion.



Fonte: Leap Motion, 2015.

Os gestos no Leap Motion são reconhecidos por um software chamado *Leap Visualizer*, que permite ao usuário visualizar os objetos mapeados pelo dispositivo, além de disponibilizar as coordenadas dos objetos capturados pelo dispositivo, em um plano de coordenadas tridimensional (MELO, 2013). A Figura 14 demonstra a ferramenta *Leap Visualizer* em execução e como são exibidos os objetos e dados mapeados.

Figura 14 – Demonstração da Ferramenta *Leap Visualizer*.



Fonte: LEAP MOTION, 2015.

2.4.4. Comparação entre os hardwares sensores de movimento

A Tabela 2 apresenta um estudo comparativo entre os dispositivos capturadores de

movimento analisado.

Tabela 2 – Comparação entre os dispositivos analisados.

Dispositivo	Sensor facial	Sensor corporal	Mapeamento da mão	Microfone embutido	Valor em US$
Kinect *for* Windows	Não	Sim	Não	Sim	$149,99
Creative Senz3D	Sim	Sim	Sim	Sim	$199,99
Leap Motion	Não	Não	Sim	Não	$79,00

De acordo com o comparativo da Tabela 2, o *Creative Senz3D* se destaca de forma positiva sobre os demais dispositivos analisados, entretanto, o seu custo de aquisição é maior que os demais. O dispositivo kinect por si só não consegue reconhecer os dedos, precisando de frameworks elaborados por terceiros para executar de forma não tão precisa esta tarefa, também, o seu custo ainda é considerado alto se comparado com o valor de aquisição do Leap Motion. Pela precisão no mapeamento das mãos e dedos, e pelo custo menor que os demais dispositivos analisados, ficou decido de utilizar o Leap Motion para este trabalho.

2.5. Interface Homem-Computador

Com o grande avanço tecnológico, a Interação Homem-Computador (IHC) está cada vez mais desafiadora, complexa e robusta, dependendo de melhores dispositivos de hardware, integrados com softwares que consigam acompanhar a mesma proporção de inovação (UZAI e DUARTE, ac. 2010).

Para Marques, Advíncula e Almeida (2013), a IHC está em crescimento e tende a ser assimilada com bastante força pelos desenvolvedores, além de ter como objetivo tornar máquinas sofisticadas cada vez mais acessíveis no que se diz respeito à interatividade com seus usuários.

Segundo Maragoni e Precipito (2006), apesar do grande investimento em pesquisas de reconhecimento de voz nas ultimas quatro décadas, as tecnologias referentes à sintetização e reconhecimento de fala ainda possuem limitações significativas. A sintetização

de voz ainda não corresponde às expectativas dos usuários, os quais estão acostumados com uma voz humano-natural.

Toda parte do software com a qual usuário pode interagir, é denominada de interface de usuário. A interface pode ser descrita por meio de software, ou hardware (periféricos de entrada e saída: mouse, teclado, microfone ou outros dispositivos) (MARQUES, ADVÍNCULA e ALMDEIDA, 2013).

Para Braga (2007), se faz necessário ensinar máquinas a falar e a escrever, pois as máquinas facilitam e aceleram as tarefas diárias a ponto de se tornarem indispensáveis. As tecnologias da fala tendem a facilitar a IHC, ao mesmo tempo em que liberam a necessidade de se utilizar um teclado, ou um mouse, possibilita o usuário utilizar os olhos e as mãos para outras tarefas.

Segundo Oliveira, Ferreira e Furst (2013), o estudo do amadurecimento das IHCs se torna bastante importante como conhecimento complementar, servindo de base teórica para o lançamento de novas tendências.

Braga (2007) cita algumas vantagens em utilizar a voz como interface entre o usuário e a máquina, são elas: A possibilidade de executar tarefas em paralelo, acesso a outros tipos de informações custosas pela dependência para com a interface, outra vantagem é com relação a associação destes sistemas a smartphones e *PDAs* (*Personal Digital Assistant*), permitindo a mobilidade tão importante nos dias atuais. A tecnologia da fala a aplicações e produtos, possibilita a acessibilidade de informações para pessoas com deficiências visuais, ou necessidades especiais, que precisem do auxilio de um reconhecedor ou sintetizador de voz.

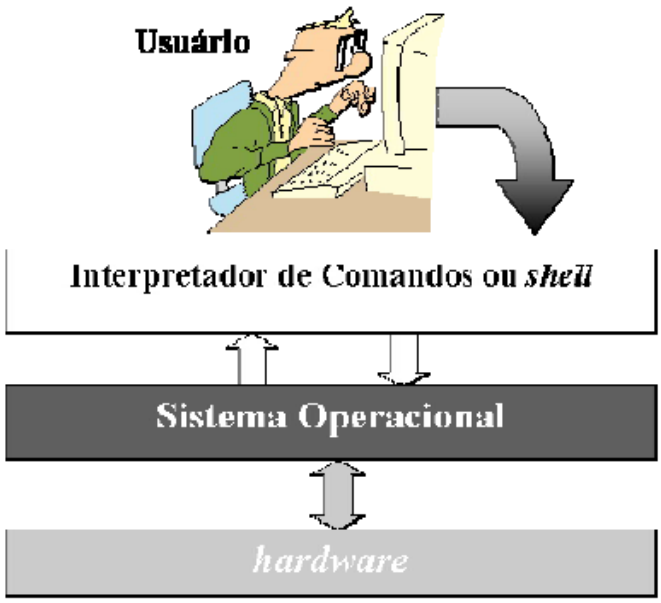
Maragoni e Precipito (2006) explicam as principais etapas para fazer uma conversão texto para voz:

- Análise da estrutura: é nessa étapa que todo o texto de entrada é processado, determinando o inicio e fim dos paragráfos, sentenças e estruturas;
- Pré-processamento: analisa o texto de entrada de forma a detectar tratamentos especiais da língua, tais como data, email, moeda, número, dentre outros;
- Conversão do Texto ao Fonena: faz a conversão de cada palavra do texto para a unidade básica do som em uma língua;
- Análise de *prosody*: processa as estruturas, sentenças, palavras e os fonemas, para determinar o melhor *prosody* a ser utilizado;

- Produção do *waveform*: nesta étapa, os fonemas e as informações *prosody* são utilizados para produzir o *waveform* de cada sentença. A maioria dos sistemas concatenam pedaços da fala humana gravada;

Oliveira, Ferreira e Furst (2013) explicam que durante a história da computação manifestaram-se diversas formas de IHC, dentre as mais utilizadas, podemos classifica-las em três tipos distintos: CLI (Interface de Linha de Comando), GUI (Interface Gráfica do Usuário) e NUI (Interface Natural do Usuário). O mesmo autor fala que as interfaces tem uma tendência de evoluir para interações cada vez mais avançadas, no entanto, as características de baixo consumo de recursos, agilidade no processamento e controle sobre o SO (Sistema Operacional), determinaram para a CLI evoluir paralelamente com as IHCs mais avançadas, permitindo o seu trabalho conjunto com as mesmas. A Figura 15 permite a visualização do funcionamento da interface CLI facilitando o acesso do usuário para se comunicar com um hardware.

Figura 15 – Funcionamento do CLI.



Fonte: OLIVEIRA; FERREIRA e FURST, 2013.

Para Maragoni e Precipito (2006), o reconhecimento de fala é o inverso da sintetização, ou seja, converte a língua falada em texto. Existem algumas características básicas para um reconhecedor de voz, são elas:

- Deve Suportar apenas uma língua especificada;

- Analisa apenas uma entrada de áudio;
- Pode de forma opcional aprender a voz dos seus usuário;
- Sua gramática pode ser dinamicamente atualizada;
- Tem suas propriedades de aplicação bem definidas.

Existem várias limitações além das tecnológicas no reconhecimento de voz, ou seja, o próprio ambiente pode ajudar a contaminar a frequência da voz com ruídos, isso pode dificultar ou até deixar impraticável qualquer tipo de tratamento para filtrar a poluição do sinal. Muitas das vezes é necessário fazer o treinamento da voz dentro do ambiente a ser utilizado o reconhecedor, assim o sistema se familiarizará com os ruídos a serem eliminados (BRAGA. 2007).

As principais etapas de um reconhecedor de voz, de acordo com Maragoni e Precipito (2006) são:

- Projetar gramática: a gramática tem o papel de definir o que o identificador deve ficar aguardando a escutar, somente as palavras que estejam contidas na gramática pode ser reconhecidas pelo identificador;
- Processador de sinal: tem a função de analisar as caracteristicas da frequência emitida pelo audio de entrada;
- Reconhecimento do fonema: sua responsabilidade é comparar os resultados padrões da frequência emitida com os resultados padrões da língua que está sendo reconhecida;
- Reconhecimento de palavras: tem o papel de comparar a sequência de fonemas resultantes em relação a estrutura da gramática;
- Geração de resultado: fornece a aplicação de forma textual a informação sobre as palavras detectadas no audio de entrada.

2.6. Métricas de Avaliação

De acordo com PERROCA e GAIDZINSKI (2003), a confiabilidade é um dos principais critérios para atestar a qualidade de ferramentas, caso obtenha uma variação baixa mensurando um atributo múltiplas vezes, se obtém uma maior confiabilidade na ferramenta.

A matriz de confusão é uma das formas mais simples de se avaliar a eficiência de um classificador em relação a sua predição (SANTOS et al., 2015). Para Prina e Trentin (2015), uma matriz de confusão é uma forma de representação de qualidade, sendo explanada

por meio de um cruzamento dos dados tidos como esperados e os dados classificados. Para Matos et al. (2009), uma matriz de confusão explana uma proporção concreta do modelo de classificação. No momento que existem apenas duas classes, uma *Positive* e a outra *Negative*, se têm apenas quatro resultados possíveis (SANTOS et al., 2015):

- *True Positive* (*TP*)*:* uma instância da classe *Positive* é classificada corretamente como *Positive*;
- *False Positive* (*FP*): uma instância da classe *Negative* é classificada erroneamente como *Positive*;
- *True Negative* (*TN*): uma instância da classe *Negative* é classificada corretamente como *Negative*;
- *False Negative* (*FN*): uma instância da classe *Positive* é classificada erroneamente como *Negativa*.

A Tabela 3 permite a visualização em forma de matriz, das predições corretas e incorretas referente às duas classes.

Tabela 3 – Matriz de confusão.

	Predicado	
Actual Class	***Positive***	***Negative***
Positive	*True Positive (TP)*	*False Negative (FN)*
Negative	*False Positive (FP)*	*True Negative (TN)*

Fonte: SANTOS et al., 2015.

Com base nos resultados obtidos na matriz de confusão é possível enxergar às medidas de precisão e revocação, nas quais são bastante utilizadas para testar a qualidade dos resultados na Recuperação da Informação (RI), Extração da Informação (EI) e IA (Aprendizado por Máquina e Processamento de Língua Natural), enquanto a primeira se enquadra como uma medida de fidelidade, a segunda é uma medida de completude (MATOS et al., 2009).

Segundo Santos et al. (2015), através da matriz de confusão se pode utilizar as métricas de desempenho e qualidade que possuem maior ênfase. Algumas delas são:

- Precisão: percentual na qual todas as amostras são realmente positivas, não inclui-se amostras negativas (MATOS et al., 2009). Neste contexto, pode ser determinada pela Fórmula 03;

$$Precissão = \frac{TP}{TP + FP}$$

Fórmula 03: Fórmula da Precisão.

- Revocação: percentual na qual é classificado como positivo todas as amostras que realmente são positivas. Tende a apresentar um índice do total de informação recuperada (MATOS et al., 2009). Neste contexto, pode ser determinada pela Fórmula 04;

$$Revocação = \frac{TP}{TP + FN}$$

Fórmula 04: Fórmula da Revocação.

- Medida-F: média harmônica da precisão e revocação (SANTOS et al., 2015). Baseada na medida de eficiência, a Medida-F foi derivada por Van Rijsbergen em 1979 (MATOS et al., 2009). Neste contexto, pode ser determinada pela Fórmula 05;

$$MedidaF = \frac{2 \times PRECISÃO \times REVOCAÇÃO}{PRECISÃO + REVOCAÇÃO}$$

Fórmula 05: Fórmula da Medida-F.

- Acurácia: taxa de instâncias classificadas normalmente (SANTOS et al., 2015), bastante utilizada para avaiação de problemas na classificação em aprendizado por máquinas (MATOS et al., 2009). Neste contexto, pode ser determinada pela Fórmula 06;

$$Acurácia = \frac{TP + TN}{TP + TN + FP + FN}$$

Fórmula 06: Fórmula da Acurácia.

- Especificidade: percentual na qual uma instância realmente negativa é classificada como negativa (MATOS et al., 2009). Neste contexto, pode ser determinada pela Fórmula 07 ;

$$Especifidade = \frac{TN}{TN + FP}$$

Fórmula 07: Fórmula para calcular a distância cosseno entre dois vetores.

- Tempo Médio de Execução (TME): Média aritimética da realização de atividades de um determinado algorítimo. Esse *timing* se da pela diferença entre *timing* final (T_f)e o *timing* inicial (T_i) da classificação (SANTOS et al., 2015). Neste contexto, pode ser determinada pela Fórmula 08.

$$TME = \frac{\sum_{k=1}^{n}\left(T_{fk} - T_{ik}\right)}{n}$$

Fórmula 08: Fórmula do Tempo Médio de Execução.

Uma medida de associação bastante utilizada para avaliar o grau de confiabilidade e precisão é o coeficiente de Kappa (PERROCA e GAIDZINSKI, 2003). Pelo fato de ser um índice de concordância ajustada, o coeficiente de Kappa constitui progresso em relação ao percentual geral de aprovação (PRINA e TRENTIN, 2015). Entretanto, o coeficiente de Kappa apresenta limitações perante sua métrica, pois muitas vezes não considera aspectos de alto valor nos dados, também não fornece informações significativas a respeito da base de concordância e discordância.

Segundo Prina e Trentin (2015), o valor de retorno obtido pelo coeficiente Kappa, tende a variar entre os lapsos numéricos 0 e 1, quanto mais próximo este valor estiver de 1, melhor a qualidade dos dados classificados. Existem várias maneiras de transformar dados quantitativos em qualitativos. Pode-se levar o exemplo utilizado por FONSECA (2000), conforme é dado ênfase na Tabela 4.

Tabela 4 – Agrupamento Qualificativo do Coeficiente Kappa.

Índice Kappa	Desempenho
< 0	Péssimo
$0 < k \leq 0.2$	Ruim
$0.2 < k \leq 0.4$	Razoável
$0.4 < k \leq 0.6$	Bom
$0.6 < k \leq 0.8$	Muito Bom
$0.8 < k \leq 1$	Excelente

Fonte: FONSECA, 2000.

2.7. Trabalhos Relacionados

2.7.1. Tradução português - LIBRAS

Diante dos diversos trabalhos encontrados, que se propõem a fazer a tradução de português para LIBRAS, seja por entrada textual, ou oral, destacam-se os trabalhos realizados a seguir (HANDTALK, 2012), (PRODEAF, 2014), (VLIBRAS, 2016), (MARTINS; PELIZZONI e HASEGAWA, 2005), (FALIBRAS, 2001) e (RYBENÁ, 2009).

2.7.1.1.Hand Talk

Fundada em 2012, a HANDTALK (2012) é uma empresa que possui uma aplicação que é capaz de traduzir de forma automática, textos em português para LIBRAS. A aplicação possui um personagem virtual que se encarrega de transmitir ao surdo à tradução da entrada recebida, que pode ser dada de forma textual, imputando palavras, ou frases em um campo. A aplicação também pode receber entradas de forma oral ou realizar a tradução automática de uma página web. Vale ressaltar que o HANDTALK (2012) é premiado internacionalmente, além de ser referência no seguimento de aplicações tradutoras em LIBRAS. A Figura 16 permite a visualização do avatar Hugo gesticulando a palavra “OI” no aplicativo Hand Talk.

Figura 16 – Avatar Hugo gesticulando a palavra “OI” no aplicativo Hand Talk.

Fonte: HANDTALK, 2012.

2.7.1.2. ProDeaf

O PRODEAF (2014) teve origem na Universidade Federal de Pernambuco, onde alunos do curso de ciência da computação deveriam criar um projeto e apresentar algumas dificuldades de comunicação que poderiam ocorrer. Surgiu então à possibilidade de desenvolver um projeto para amenizar as dificuldades encontradas. Depois de várias discursões, tiveram a ideia de desenvolver uma solução global para pessoas deficientes auditivas.

O PRODEAF (2014) está disponível das plataformas web e como aplicativo para smartphones Android, IOS e Windows Phone. O software consegue traduzir frases expressas de forma oral, ou textual, além de também possuir um tradutor automático de páginas web para LIBRAS. A Figura 17 demonstra a versão para smartphones do ProDeaf, e A Figura 18 demonstra a ferramenta ProDeaf na versão Web traduzindo um texto selecionado pelo usuário.

Figura 17 – Versão Móvel do ProDeaf.

Fonte: PRODEAF, 2014.

Figura 18 – Versão web do ProDeaf.

Fonte: PRODEAF, 2014.

2.7.1.3. VLIBRAS

O VLIBRAS (2016) é fruto de uma parceria entre o Ministério do Planejamento, Desenvolvimento e Gestão (MP), a Secretaria de Tecnologia da Informação e a Universidade Federal da Paraíba (UFPB). Sua plataforma consiste em um conjunto de ferramentas computacionais de código aberto, que juntas são capazes de traduzir áudio e texto para

LIBRAS. Uma das suas ferramentas permite, também, a tradução de vídeos. A sua interface é simples e amigável, o usuário pode configurar algumas variáveis, tais como: velocidade da animação, tamanho da tela, e exibição de legenda. A Figura 19 permite a visualização do VLIBRAS traduzindo um texto selecionado pelo usuário.

Figura 19 – Software VLIBRAS traduzindo um texto selecionado.

Fonte: VLIBRAS, 2016.

2.7.1.4. PULØ

O trabalho de Martins, Pelizzoni e Hasegawa (2005) tem como objetivo descrever o desenvolvimento da ferramenta PULØ – *Portuguese-UNL-LIST deOralizer*. O PULØ é a versão experimental de um sistema de tradução automática de português para LIBRAS, sua base é exclusiva de conhecimento linguístico que utiliza a estratégia de tradução indireta por interlíngua. Tais características limitam o sistema a prever apenas o desenvolvimento de dicionários e gramáticas, dispensando outros repositórios de informação. Este sistema pode receber como entrada textos em português-brasileiro (PT-BR), que por sua vez é desmembrado pelo *recognizer* em unidades de processamento, a sentença desmembrada é isolada e normalizada. Com a normalização é possível reconhecer interjeições e desvios padrões no vocabulário de entrada, a sentença normalizada é subdividida em itens lexicais, que recolhe as informações relativas a cada *token* e as repassa para que seja feita uma analise sintática, que posteriormente passa por uma analise semântica. Após passar por todo o processo é construída uma saída com o auxilio da gramatica e do dicionário, denominada *LIST*, na qual serve de entrada para o sintetizador de sinais.

2.7.1.5. FALIBRAS

O FALIBRAS (2001) teve inicio no Instituto de Computação da Universidade Federal de Alagoas, e tem como objetivo principal facilitar a comunicação entre surdos e ouvintes através de uma ferramenta de tradução automática do português para LIBRAS. Atualmente o FALIBRAS (2001) trabalha com uma arvore sintática baseada em uma gramática livre de contextos, devolvendo uma estrutura frasal em LIBRAS. Além de fazer uso da linguagem de programação Java, o FALIBRAS (2001), passou a utilizar também a linguagem de programação PROLOG como meio de expressar as regras de tradução que sustentam o analisador sintático e o módulo tradutor. A ferramenta se apresenta como um plug-in para o browser Firefox, facilitando a sua integração com a web. Sua saída se dá por meio de uma animação 3D que executa sinais em LIBRAS.

Figura 20 – Demonstração do software FALIBRAS em execução.

Fonte: FALIBRAS, 2001.

2.7.1.6. RYBENÁ

Desenvolvido pelo Grupo ICTS de Brasília - Distrito Federal (DF), no qual é uma união entre o instituto de pesquisa tecnológica CTS e a empresa privada CTS Ltda, o RYBENÁ (2009) é um software web que possibilita a tradução automática de palavras isoladas e frases em português, para LIBRAS e voz. A sua solução web é composta por dois softwares: o Player Rybená, que garante a tradução de qualquer página html (Hyper Text Markup Language) ou textos escritos em português para LIBRAS, e o Rybená Voz, que consegue converter textos de páginas html em voz humana sintetizada. O Player Rybená

possui uma comunicação amigável com o usuário, também consegue tratar de forma inteligente as ambiguidades linguísticas.

Figura 21 – Demonstração visual do Player Rybená e Rybená Voz.

Fonte: RYBENÁ, 2009.

2.7.2. Tradução LIBRAS - português

No contexto que se dispõe a tradução de LIBRAS para português, destacam-se os trabalhos de Melo (2013); Monteiro e Antunes, (2013); Pires, (2012); Tavares e Leithardt (2007); Perdomo e Siqueira, (2016).

2.7.2.1. Melo (2013)

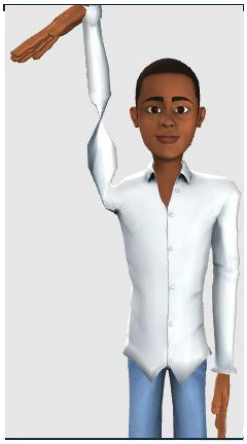
No trabalho elaborado por Melo (2013), foi desenvolvida uma solução capaz de mapear gestos efetuados pelas mãos através do dispositivo Leap Motion. Utilizando a linguagem de programação C#, selecionou algumas informações emitidas pelo dispositivo, tais como: valor da posição central da palma da mão, valor da direção da mão, valor da posição da mão e valor da posição do cotovelo. A combinação destas informações foi caracterizada como um gesto. A fim de analisar o comportamento das informações obtidas, a aplicação desenvolvida por Melo (2013) gera uma notação escrita no padrão *Extensible Markup Language* (*XML*). Tal arquivo será compilado pelo software ProDeaf, para que o mesmo possa executar a sequência de animações obtidas pela leitura do arquivo *XML* fornecido. Os primeiros cenários de testes foram executados com o objetivo de definir se o

avatar do ProDeaf iria conseguir interpretar o gesto composto pela seguinte configuração: mão no centro e direcionada para baixo. Para essa configuração, os movimentos que deveriam ser executados seriam: mão à direita, mão à esquerda, e por fim, a mão voltaria para o centro do corpo. Tal teste chegou a 92% de margem de erro, e apenas 8% de acerto. Melo (2013) informa que a falta de robustez dos dados adquiridos pelo dispositivo foi uma das maiores limitações encontradas durante o desenvolvimento. A Figura 22 e a Figura 23 demonstram o software ProDeaf executando a animação referente ao arquivo *XML* gerado, com os dados capturados pelo Leap Motion.

Figura 22 – Animação executando a letra ‘B’ corretamente mapeada pelo Leap Motion.

Fonte: MELO, 2013.

Figura 23 – Comportamento do avatar do ProDeaf ao carregar as coordenadas geradas pelo Leap Motion.

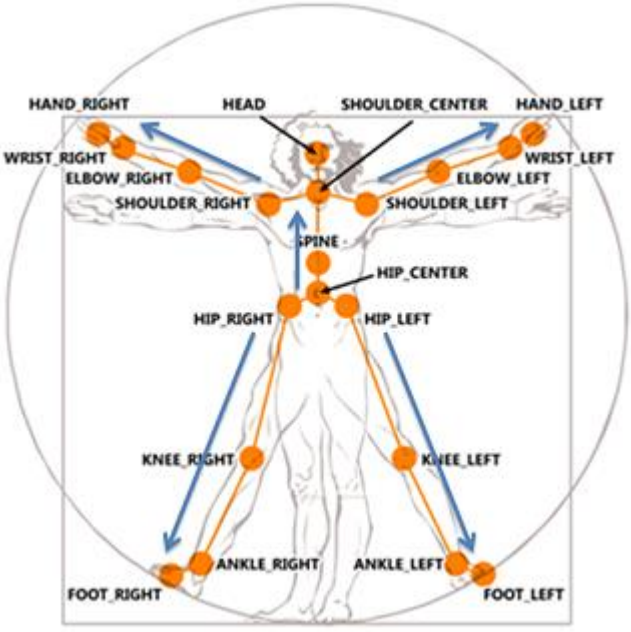


Fonte: MELO, 2013.

2.7.2.2. Monteiro e Antunes (2013)

O trabalho de Monteiro e Antunes (2013) teve como objetivo analisar a capacidade do sensor de profundidade kinect do console Xbox 360, para o reconhecimento de sinais de LIBRAS. Através de estudos sobre a arquitetura e testes práticos do dispositivo, Monteiro e Antunes (2013) chegaram à conclusão que o equipamento pode ser útil como complemento de outros dispositivos para o desenvolvimento de uma aplicação que realize a tradução de LIBRAS para português. O dispositivo kinect permite de uma forma eficiente a captação de movimentos, pois o mesmo reconhece o esqueleto humano e suas articulações. Esse fator é importante, pois alguns gestos em LIBRAS são elaborados pelo movimento de diversas partes do corpo além de braços e mãos, como pernas e cabeça. Vale ressaltar que uma deficiência do aparelho é não conseguir realizar a captura dos dedos, que são muito importantes ao se tratar de LIBRAS. Para mitigar esta dificuldade é necessário a utilização de alguns frameworks, como OpenNI, OpenCV e Nite. A Figura 24 permite a visualização de como o esqueleto humano é reconhecido pelo sensor kinect.

Figura 24 – Partes do esqueleto humano reconhecido pelo kinect.



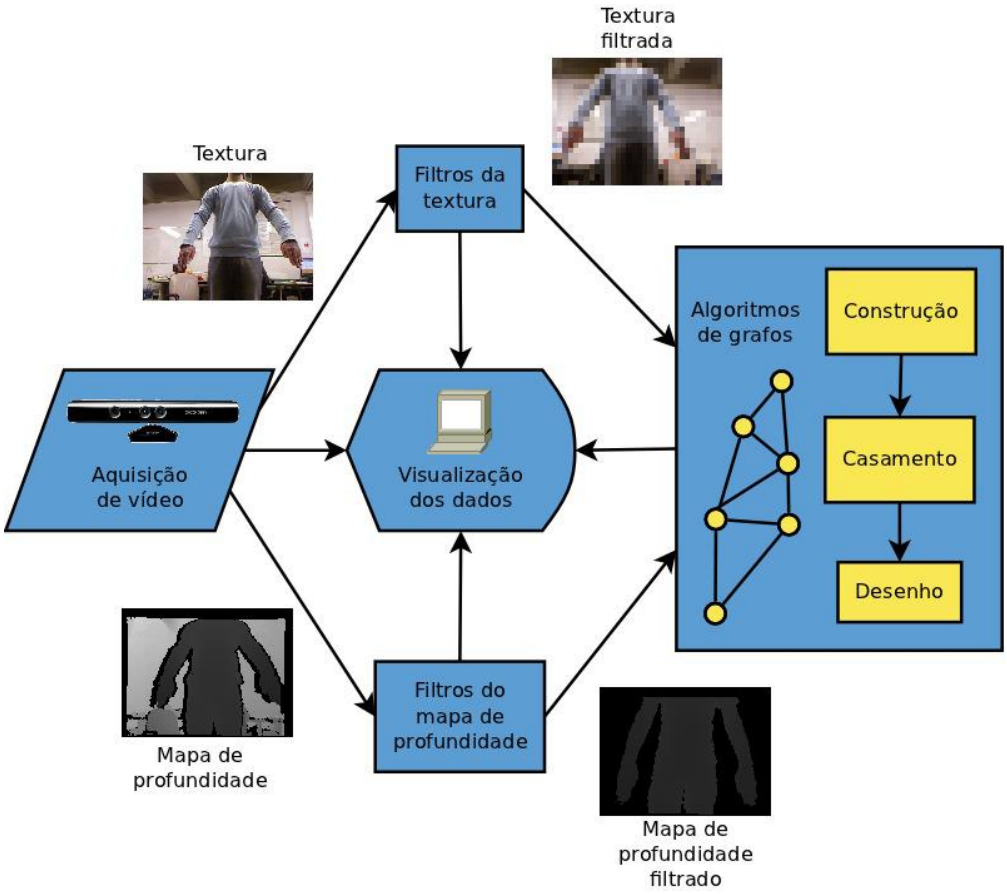
Fonte: MICROSOFT, 2013.

2.7.2.3. Pires (2012)

No trabalho de Pires (2012) foi desenvolvido um método baseado em casamento

entre grafos, explorando o lado positivo de se usar a informação complementar obtida pela imagem de profundidade registrada, com a imagem de textura. O procedimento desenvolvido visa demonstrar por meio de rótulos com cores de diferentes tonalidades, o local em que cada bloco de pixel está se deslocando. Os dados de profundidade quando utilizados em conjunto com valores RGB e coordenadas de textura, tendem a facilitar o torneamento de objetos demarcados como importantes, e deixam os resultados significativamente melhores quando estes são considerados característica descritiva de cada pixel. Como entrada o algoritmo recebe uma sequência de pares de imagens registradas contendo a textura colorida e o mapa de profundidade. A textura é representada por três canais contendo as cores aditivas, e o mapa de profundidade é uma imagem em diferentes tonalidades de cinza, sendo também conhecido como canal de profundidade. Pode-se dizer que o algoritmo constitui-se de uma entrada de quatro canais. O dispositivo de captura utilizado foi o kinect, Pires (2013) diz que o dispositivo demonstrou ser um sensor de grande importância para a área de visão computacional. A Figura 25 demonstra o fluxograma do método implementado por Pires (2013).

Figura 25 – Fluxo de dados do método implementado.



Fonte: PIRES, 2013.

Entretanto, constatou-se que o mapa de profundidade gerado pelo aparelho tem certa dependência com a emissão da luz infravermelha emitida pelo mesmo, de modo que existe bastante interferência em cima das imagens em locais que possuem a presença de raios infravermelhos. Também se constatou que em 1 frame a cada 60, ocorre falhas em capturar a imagem de textura. Essas falhas são caracterizadas por mudança nos quadros, em conjunto com a exibição de uma composição da parte textura como uma matriz de pontos. Na Figura 26, é demonstrada a falha de captura na imagem de textura, onde a imagem a esquerda representa um quadro anterior na sequência de gravação do quadro a direita, quadro em que ocorre a falha.

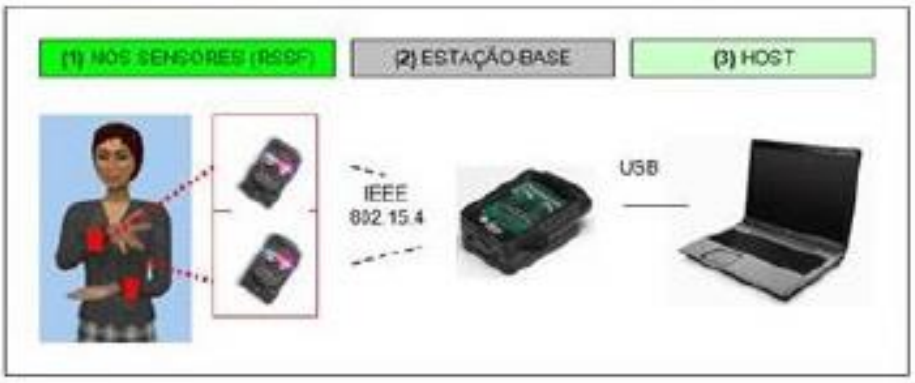
Figura 26 – Falha de captura na imagem de textura.

Fonte: PIRES, 2013.

2.7.2.4. Tavares e Leithardt (2007)

O tralho de Tavares e Leithardt (2007) denominado SensorLIBRAS, baseia-se nos conceitos de computação ubíqua pelo fato de ser um sistema embarcado, instalado no dispositivo portátil Sun *Small Programmable Object Technology* (*SPOT*). O dispositivo traz consigo diversos sensores, uma unidade central de processamento, além de possuir consumo eficiente de energia e uma comunicação sem fio. O Sun *SPOT* é capaz de ler os dados capturados pelos sensores de aceleração concatenados aos cinco pinos de entrada. O software foi programado em Java *Micro Edition* (*ME*) e trata cada dispositivo da Sun *SPOT* como um nó, constituindo uma Rede de Sensores Sem Fio (RSSF). Isso possibilita a leitura dos movimentos da mão para emitir o sinal de um gesto efetuado em LIBRAS. O SensorLIBRAS constitui-se de uma luva na qual o surdo utilizará para emitir o pressionamento dos dedos de ambas as mãos, os enviando via conexão *wireless* para estação base conectada via porta USB ao *host* como demonstra a Figura 27.

Figura 27 – Conexão da “Luva” com a estação base e o host.



Fonte: TAVARES e LEITHARDT, 2007.

O software que possui a responsabilidade de ler, interpretar e traduzir os sinais de LIBRAS para o português está dividido em quatro módulos: O primeiro denominado *Reader*, possui a funcionalidade principal de ler os dados obtidos pelos movimentos das mãos e o pressionamento dos dedos, transmitindo-os para a estação base, o segundo denominado *analyzer*, tem a responsabilidade de analisar e interpretar os dados recebidos pela estação base, retornando o comportamento dos três eixos (x , y e z) concatenados, formando um sinal de forma matemática, o terceiro módulo denominado *parser*, recebe do *analyzer* a função com o sinal concatenado para fazer uma busca na base de dados, retornando o caractere correspondente, o último e não menos importante é o *feedback*, responsável por sinalizar de forma visual através dos *LEDs* integrados ao dispositivo Sun *SPOT*. Ao receber o argumento positivo do módulo *parser*, o software faz piscar alternadamente os oito *LEDs* na cor verde, no entanto, caso receba um parâmetro contrário, os *LEDs* são sinalizados na cor vermelha. A Figura 28 permite visualizar a "luva" protótipo para capturar os dados pelo SensorLIBRAS.

Figura 28 – "Luva" protótipo para capturar dados.

Fonte: TAVARES e LEITHARDT, 2007

2.7.2.5. Perdomo e Siqueira (2016)

O trabalho de Perdomo e Siqueira (2016) apresenta um protótipo para reconhecimento de 20 letras do alfabeto manual de LIBRAS, nas quais foram escolhidas em razão da ausência de movimento para a realização de seus respectivos sinais. A captura de gestos se deve ao sensor kinect, combinado com a segmentação e extração de descritores da imagem. O resultado é alocado para classificação por *Support Vector Machine* (*SVM*). Para mitigar as restrições impostas pelo kinect *for* Xbox 360, assim como o trabalho de Monteiro e Antunes (2013), foram utilizadas as bibliotecas OpenCV, NITE, OpenNI e o driver PrimeSense que possui a biblioteca SURF integrada a implementação *SVM 1-vs-1*. Com isso, o mapeamento de disparidade emitido pelo sensor de profundidade, capturado em *near-mode*, pode ser utilizado sem o uso da sua biblioteca padrão. Entretanto, o mapeamento do esqueleto foi obtido pela biblioteca SkeletonSensor. Uma área quadrada de 180x180 pixels foi centralizada no nodo da mão direita, ponto no qual foi escolhido de forma aleatória como região de interesse (ROI) baseado na visão do espaço ocupado com uma distância de 0.8*m* para com o kinect.

O fundo da imagem capturada é removido com base no calculo de profundidade do nodo da mão. Remove-se também o antebraço, apagando as regiões mais distantes abaixo do ponto escolhido. Após a segmentação das características das mãos, utiliza-se a biblioteca SURF para extração dos descritores de cada quadro capturado, nos quais irão compor histogramas encaminhados a três *SVMs* diferentes: *kernels* lineares, polinomiais e radiais. Todos esses processos podem ser observados na Figura 29, que demonstra o protótipo desenvolvido em funcionamento.

Figura 29 – Protótipo desenvolvido demonstrando o esqueleto detectado, com a mão segmentada e os descritores SURF encontrados.

Fonte: PERDOMO e SIQUEIRA, 2016

2.7.3. Comparativo Entre Trabalhos Relacionados

A Tabela 5 apresenta um estudo comparativo entre o trabalho proposto e os demais trabalhos analisados:

Tabela 5 – Comparação entre os trabalhos analisados.

Trabalho	Tradução português - LIBRAS	Tradução LIBRAS - português	Interpretação em tempo real	Entrada
Hand Talk (2012)	Sim	Não	Sim	Texto / Voz
ProDeaf (2014)	Sim	Não	Sim	Texto / Voz
VLIBRAS (2016)	Sim	Não	Sim	Texto / Voz
PULØ	Sim	Não	Sim	Texto
FALIBRAS (2001)	Sim	Não	Não	Texto
Rybená (2009)	Sim	Não	Sim	Texto / Voz
Melo (2013)	Não	Sim	Sim	Leap Motion
Monteiro e Antunes (2013)	Não	Sim	Sim	Kinect
Pires (2012)	Não	Sim	Sim	Kinect
SensorLIBRAS	Não	Sim	Sim	Sun *SPOT*

Perdomo e Siqueira (2016)	Não	Sim	Sim	Kinect
SINTRALIB	**Sim**	**Sim**	**Sim**	**Texto/Voz/LeapMotion**

Destaca-se no SINTRALIB, o diferencial de traduzir tanto de português para LIBRAS, quanto de LIBRAS para português em tempo real, auxiliando a comunicação em ambas as direções entre deficientes auditivos e ouvintes.

3. TRABALHO DESENVOLVIDO

Neste capítulo, serão apresentadas as principais informações sobre o software SINTRALIB, bem como uma visão detalhada do desenvolvimento e usabilidade dos dois módulos que compõem a aplicação: tradutor português - LIBRAS e tradutor LIBRAS - português. Na seção 3.1, será abordada uma visão geral da aplicação SINTRALIB, e uma visão detalhada de ambos os módulos.

3.1. SINTRALIB

O SINTRALIB consiste em um software que inicialmente traduz letras do alfabeto português para LIBRAS, e também, a tradução de letras estáticas do alfabeto em LIBRAS para o português. Os recursos utilizados para desenvolvimento da aplicação foram:

- Java 8: segundo a Oracle (2009), Java é uma plataforma computacional e línguagem de programação anunciada pela Sun Microsystems em 1995. A sua versão 8 tem suporte para expressões Lambda;
- NetBeans IDE (Ambiente de Desenvolvimento Integrado): é um ambiente de desenvolvimento totalmente escrito na linguagem de programação Java, esta ferramenta permite aos programadores escrever, compilar, depurar e instalar programas (NETBEANS, 2017);
- Leap Motion: dispositivo escolhido para ser utilizado como o sensor capturador de movimentos na versão atual do SINTRALIB. Sua escolha se deve ao fato de o dispositivo conseguir capturar os movimentos das mãos com uma maior

sensibilidade e ter um custo financeiro mais baixo que os outros dispositivos analisados no capitulo 2, sessão 2.4;

- IBM ViaVoice: software para o reconhecimento e sintetização de voz;
- *Blender*: plataforma de código aberto para modelagem, criação, texturização e outras funcionalidades para animações 3D;
- *Bitbucket*: serviço de hospedagem de projetos, permite ter um controle de versão distribuido baseado em Git;
- *Wrike*: de acordo com Wrike (2017), é uma ferramenta de gestão de projetos, no qual se pode delegar tarefas, ordenar atividades de forma cronológica, receber e enviar notificações a cada nova alteração no projeto, estipular prazos, entre outras funcionalidades;
- Computador: foi utilizado o notebook Sony Vaio VPCEB4B4E, com 4 Gigabyte (GB) de memória ram, 500GB de disco, processador Intel core i5 - 1º geração, com Windows 7 64-bits, e também, o notebook Dell Inspiron Series 5100, com 8GB de memória ram, 1 Terabyte (TB) de disco, processador Intel core i5 - 5º geração, com Windows 10 64-bits.

A aplicação possui dois módulos independentes, o primeiro cujo nome é *Listener*, tem a responsabilidade de traduzir de português para LIBRAS uma entrada conhecida em sua gramática, o segundo denominado *Vision*, permite a tradução LIBRAS - português. A seguir iremos descrever detalhadamente cada um dos módulos.

3.1.1. *Listener*

O módulo *listener* é subdividido em: *input*, *core* e *out*. Todos são altamente dependentes e terão suas características exploradas neste tópico. O *listener* permite que o usuário ouvinte possa expressar-se em português por meio de voz ou texto, e retorna ao usuário surdo, através de um interprete virtual, a devida entrada em LIBRAS. Vale ressaltar que a versão atual do módulo é limitada em letras do alfabeto da língua portuguesa.

Para construção deste módulo, foi necessário o uso de algumas bibliotecas e de softwares que auxiliam no reconhecimento de voz, desenvolvimento de animações e reprodução de vídeos.

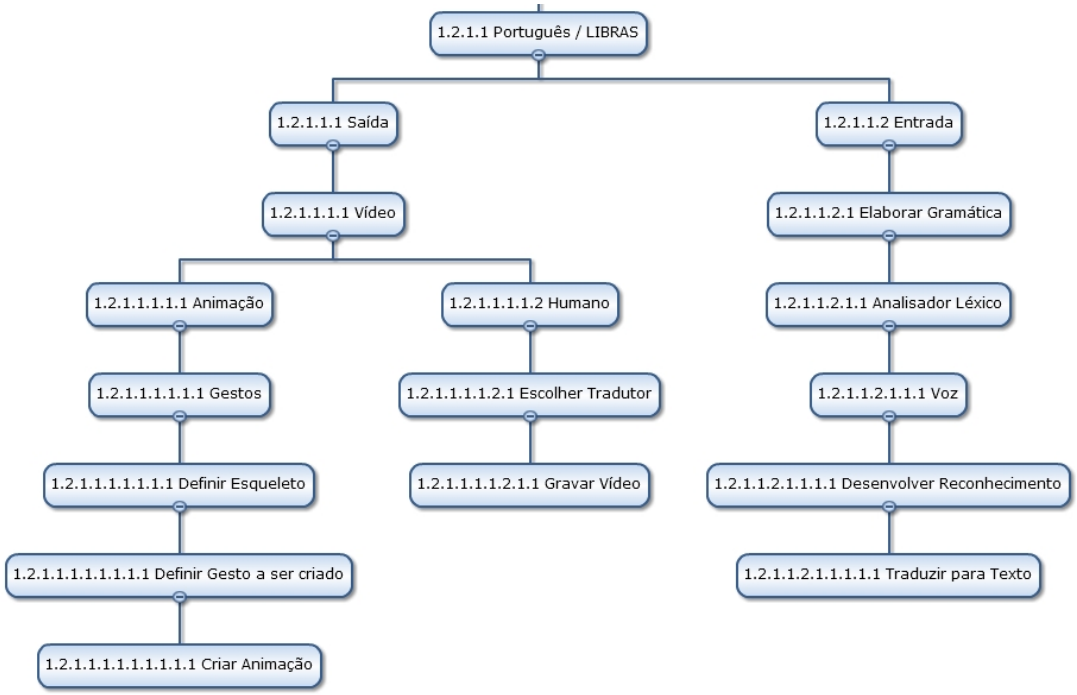
Na camada *input* fica a responsabilidade de capturar a entrada fornecida pelo usuário ouvinte, entrada essa que pode ser feita por meio de voz ou texto. Como a aplicação

foi totalmente desenvolvida na linguagem de programação Java, foi tida a necessidade de encontrar um sintetizador e reconhecedor de voz compatível com a linguagem. No meio de poucos encontrados, acabou-se por optar por *Java Speech API* (*JSAPI*), combinado com o software de reconhecimento e sintetização de voz IBM ViaVoice. O IBM ViaVoice é um software da IBM, que permite ao usuário falar ao invés de digitar. É necessário passar por um treinamento de voz e ambiente para que o software possa filtrar os ruídos encontrados durante o reconhecimento de voz. A camada *Input* cria uma instância através de um processamento paralelo com a *Application Programming Interface* (*API*) *JSAPI*, e permite que a mesma receba do software IBM ViaVoice, um *InputReader* com os dados sonoros obtidos. Com essa informação, faz-se uma varredura na gramática elaborada a fim de validar a entrada reconhecida por meio de um analisador sintático, caso a estrutura seja valida, é convertida para texto a entrada expressa de forma oral.

A camada *core*, é responsável por controlar o que será exibido para o usuário surdo, ela recebe o texto gerado na camada *input* e quebra o mesmo em vários *tokens*. Para cada *token* é feita uma verificação de relação, caso exista uma relação do *token* com algum sinal na base de dados, o *token* é acrescentado a uma fila de animações a serem exibidas pela camada *Out*, caso não exista, cada letra do *token* será acrescida na fila de animações, com isso o *token* será “soletrado” pelo interprete virtual na camada *out*.

A camada *out* é responsável por controlar o interprete virtual que emitirá a informação passada pelo ouvinte ao deficiente auditivo conhecedor de LIBRAS. Ela analisa constantemente se existe alguma animação na fila para exibição, caso sim, a mesma remove o objeto da fila recebendo as informações necessárias passada pela camada *core*, tais como: tempo inicial, tempo final, referência e descrição da animação (Falaremos a função de cada uma destas informações). Com estes dados, a camada *out* pode reproduzir a animação do interprete virtual realizando o sinal referente à entrada passada pelo ouvinte. A Figura 30 permite a visualização da Estrutura Analítica de Projetos (EAP) referente ao módulo *listener*.

Figura 30 – EAP do módulo *Listener*.



Fonte: Elaborada pelo autor.

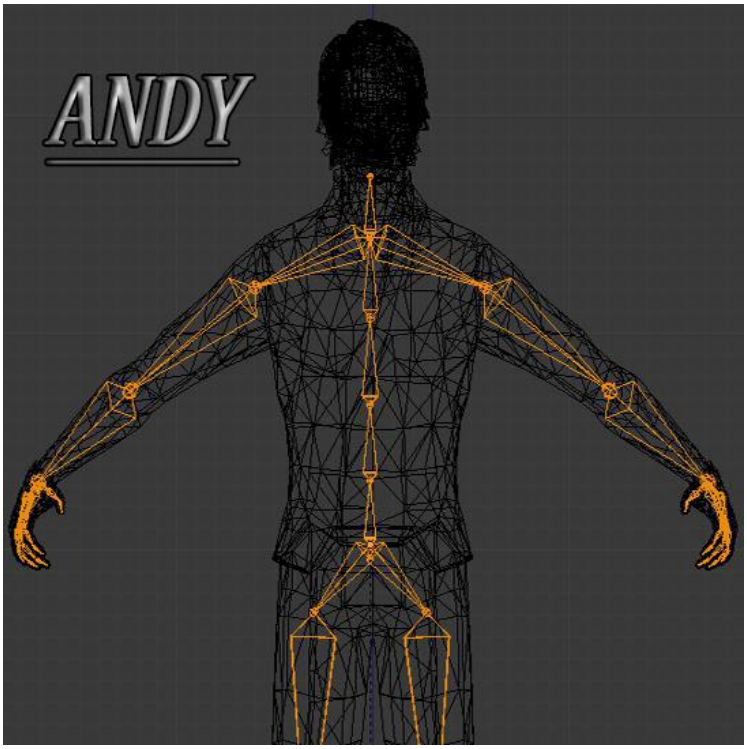
O interprete virtual é responsável por transmitir a informação passada pelo ouvinte para o deficiente auditivo conhecedor de LIBRAS, este tópico será abrangido detalhadamente sobre o interprete virtual do software SINTRALIB. Também será falado sobre as bibliotecas, ferramentas e algoritmos utilizados para a sua criação e desempenho do seu papel na aplicação, além de abordar as dificuldades apresentadas no decorrer da sua implementação.

Para desenvolver o personagem, foi desenvolvido um modelo 3D na plataforma de modelagem 3D, *Blender*, o modelo foi utilizado para fazer as animações em LIBRAS. O personagem, batizado como Andy, possui uma aparência simples, com roupas pouco chamativas, conforme visto na Figura 31, com isso consegue-se que a atenção do deficiente auditivo seja voltada aos movimentos do personagem e não a sua aparência. Com o modelo 3D feito, iniciou-se a preparação para que ele pudesse estabelecer a movimentação necessária para gesticular os sinais em LIBRAS. Para isso, foi adicionado um esqueleto ao modelo, conforme podemos observar na Figura 32.

Figura 31 – Modelo 3D criado.

Fonte: CUNHA; ALVES e FARO (2014).

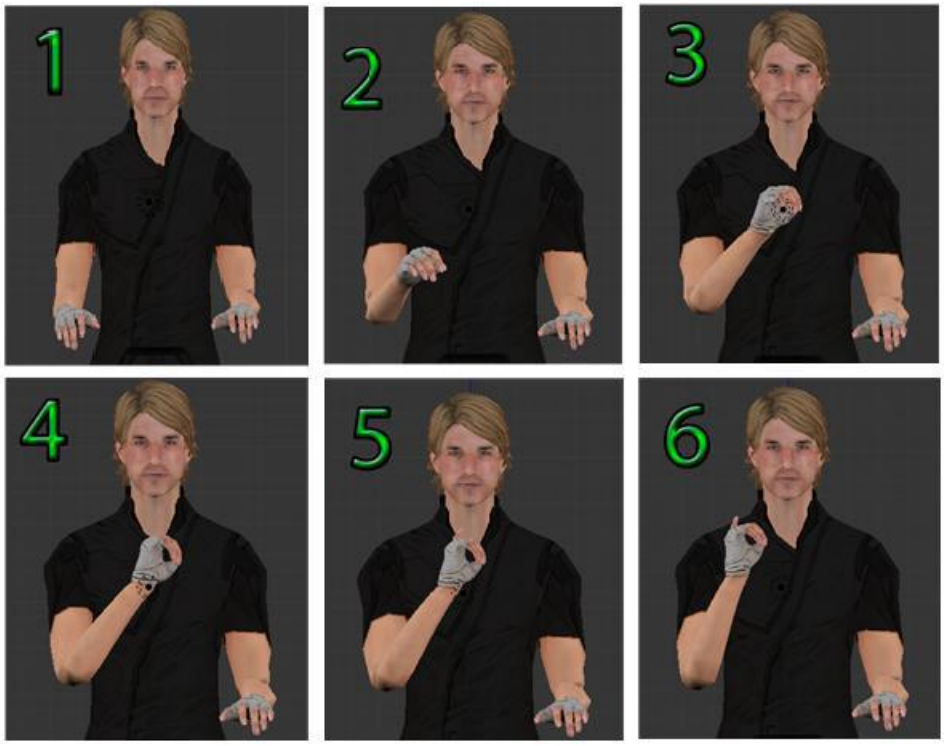
Figura 32 – Esqueleto adicionado ao modelo 3D.



Fonte: CUNHA, ALVES e FARO (2014).

A movimentação do modelo com base na posição de "ossos" consiste em associar um objeto denominado "osso" a um conjunto específico de vértices que se movimentarão para acompanhá-lo quando sua posição for alterada. Com o esqueleto anexado ao corpo do modelo 3D, pode-se dar inicio ao movimento com os ossos do esqueleto, agregando movimentação ao corpo. Cada osso é movimentado individualmente, no entanto, se existir um osso ligado a outro, pode-se estabelecer a movimentação conjunta. Foi utilizada a divisão de frames para fazer a movimentação do personagem, dividindo o movimento de cada sinal em várias partes, e alocando cada parte em uma imagem fixa, resultando em imagens contínuas com movimentos pré-estabelecidos. Na Figura 33 podem-se observar algumas das posições criadas para a construção da palavra “oi” em LIBRAS. Uma vez que a movimentação relativa a um sinal de LIBRAS esteja pronta, a animação pode ser exportada como um objeto XML, assim poderá ser utilizada em outras plataformas.

Figura 33 – Sequência de frames com as poses básicas para formar a palavra “oi”.



Fonte: CUNHA, ALVES e FARO (2014).

O Java pode utilizar o xml (exportado pelo *Blender*) contendo a animação criada como se fosse um objeto qualquer. Entretanto, faz-se necessário que a exportação seja com a extensão “.blender2java” ou salvar como um arquivo *Blender* Python, com extensão “.py”.

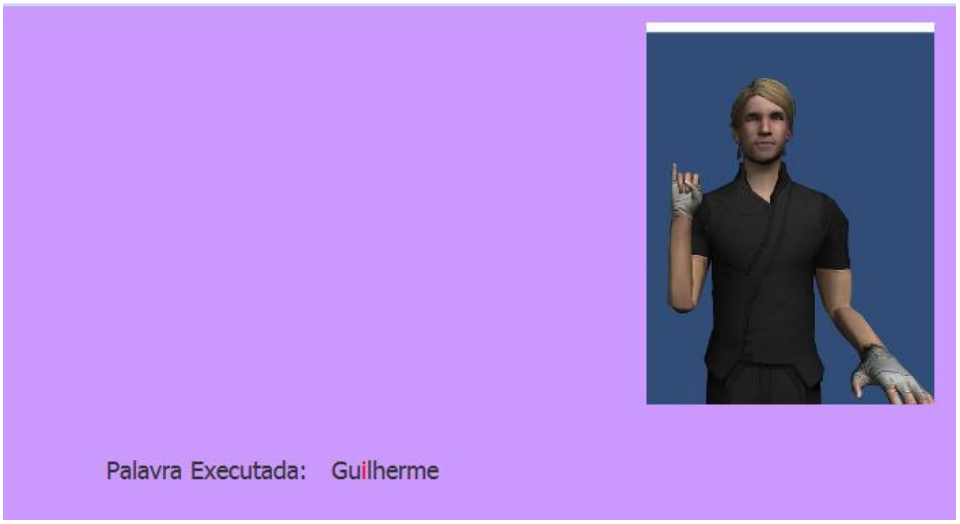
Com a pouca variedade de opções que o Java oferece para emular uma animação feita no *Blender*, recorremos em primeiro momento a plataforma *JmonkeyEngine*, que é uma *Game Engine* bastante utilizada por desenvolvedores Java. Com ela pode-se criar jogos 3D utilizando tecnologia moderna. Apesar de a plataforma oferecer bastantes recursos para desenvolvimento e modelagem 3D, não conseguimos aplicar suas funcionalidades de forma eficaz ao projeto SINTRALIB. Foi notado um lapso de tempo grande em relação à transição do movimento de uma animação para outra. Considerando que a aplicação necessita passar a informação de forma quase que instantânea, o resultado obtido ficou distante do considerado agradável.

Para resolver o problema detectado, decidiu-se por descartar a possibilidade de utilizar o Java em fazer a leitura do arquivo *xml* contendo a animação com os sinais, utilizou-se então o *Unity 3D*, que também é uma *Game Engine* assim como o *JmonkeyEngine*. O *Unity 3D* é totalmente compatível com os arquivos exportados pelo *Blender*, facilitando o reconhecimento e a programação de eventos para com a animação, vale ressaltar que até então não foi necessário utilizar programação em Java, já que o Unity 3D possui em sua plataforma um ambiente de desenvolvimento integrado, o que se permitiu criar uma aplicação em formato de *Game*, onde podemos executar os movimentos criados no *Blender*. Para não depender do Java ao exibir diretamente a animação, e visando um melhor desempenho em relação a tempo de execução da animação, foi decidido elaborar um simples reprodutor de vídeo em Java, que possa reproduzir um vídeo contendo todas as animações. Para que isso seja possível, foi feito um cadastro de animações na base de dados com os seguintes atributos: tempo inicio, tempo fim, descrição e referência, abaixo explicamos melhor o papel de cada um dos atributos.

- Tempo Inicio: *timing* em milisegundos onde se inicia a animação no vídeo ;
- Tempo fim: *timing* em milisegundos onde é finalizada a animação no vídeo;
- Descrição: descrição da animação a ser reproduzida;
- Referência: descrição do *token* na qual a animação faz referência.

Com isso, o desempenho da aplicação em relação ao vídeo, teve um ganho de desempenho em relação ao tempo de execução da animação de 73%. A Figura 34 demonstra o interprete virtual “Andy” reproduzindo o sinal da letra “i”.

Figura 34 – Andy reproduzindo o sinal da letra “i”.



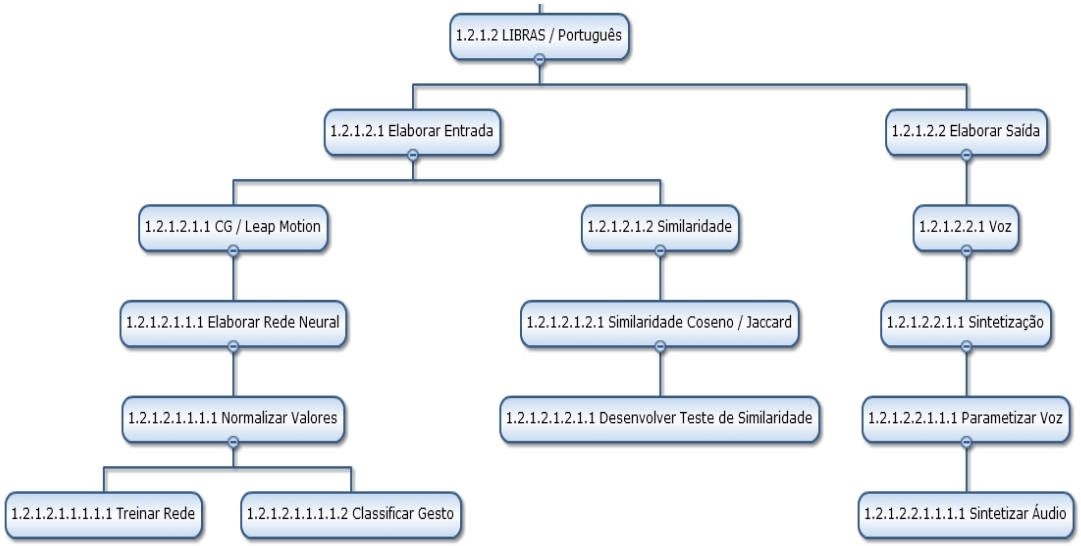
Fonte: Elaborada pelo autor.

A camada *Out* tem total controle sobre o reprodutor de vídeo incorporado a aplicação, e quando se executa a animação contida no inicio da fila, ordena-se para o reprodutor de vídeo mover o seu *timing* para o tempo inicio da animação a ser executada, com *delay* de 1 segundo, a animação é executada até que chegue ao seu tempo final, caso exista outras animações na fila, repete-se o processo até que a fila esteja vazia. Enquanto não existir nenhum elemento na fila de animações, o *timing* do reprodutor de vídeo move-se para um estado neutro, onde seu inicio e fim, ficam a repetir-se até que as condições sejam desfavoráveis.

3.1.2. *Vision*

De forma geral, o módulo *vision* tem a reponsabilidade de traduzir sinais efetuados em LIBRAS para o português. Vale destacar que nesta versão do SINTRALIB serão traduzidas somente letras estáticas do alfabeto em LIBRAS. A Figura 35 permite a visualização da EAP referente ao módulo *Vision*.

Figura 35 – EAP do módulo *Vision*.



Fonte: Elaborada pelo autor.

Como mencionado no item 3.1, ficou resolvido utilizar o Leap Motion como o dispositivo que realizará a captura dos movimentos de ambas as mãos do surdo. O Leap Motion possui uma imensa biblioteca para controlar o dispositivo. Para melhor entendimento deste trabalho, explicaremos o uso de algumas classes e métodos encontrados na biblioteca do Leap Motion para Java.

Para começar a utilizar o dispositivo, é necessário herdar da classe *Listener*, essa classe vem acompanhada de um conjunto de métodos *call-backs* que permitem o acesso ao dispositivo:

- *onConnect*: este método é chamado quando o dispositivo Leap Motion é conectado ao computador cliente;
- *onDeviceChange*: método invocado toda vez que o dispositivo for conectado, desconectado ou mudar de estado;
- *onDeviceFailure*: método acionado toda vez que o dispositivo é conectado ao computador cliente com funcionamento incorreto;
- *onDisconnect*: ao contrário do método *onConnect*, este método é requerido toda vez que o dispositivo Leap Motion for desconectado do computador cliente;
- *onExit*: de certa forma parecido com o método *onDisconnect*, com a diferença que além do dispositivo desconectado, o método será acionado caso a instância seja destruída;

- *onFocusGained*: toda vez que a aplicação é considera aplicativo de primeiro plano, este método será invocado;
- *onFocusLost*: o inverso do método onFocusGained, chamado quando o aplicativo não está mais em primeiro plano;
- *onFrame*: o método mais relevante para manipulação do controlador, é chamado quando um novo quadro de mãos e dedos está disponível para leitura. Todo o desenvolvimento referente ao rastreamento e mapeamento das mãos passam por esté método;
- *onImages*: chamado toda vez que novas imagens estiverem com acesso disponível pelo controlador;
- *onInit*: bem parecido com o método *onConnect*, esse método é chamado toda vez que o objeto *Listener* é adicionado ao objeto *Controller*;
- *onLogMessage*: invocado quando o serviço relata um erro, aviso, ou mudança de status, através de uma mensagem de log;
- *onServiceChange*: chamado toda vez que o serviço do Leap Motion é pausado ou reiniciado;
- *onServiceConnect*: chamado quando o serviço do Leap Motion é ligado e se conecta ao controlador do dispositivo;
- *onServiceDisconnect*: chamado quando o serviço do Leap Motion é desligado;

Todos os métodos acima requerem um objeto *Controller* como parâmetro. A classe *Controller* é a principal interface que faz referência ao dispositivo Leap Motion. Para acessar a sequência de quadros fornecidos pelo dispositivo, é necessário fazer uma instância desta classe, e posteriormente um acesso ao método *frame* que está contido no objeto *Controller*.

O método *frame* requer um parâmetro do tipo inteiro e retorna um objeto da classe *Frame*, esse parâmetro se faz valer pelo fato do dispositivo armazenar um histórico de quadros que podem ser manipulados. Passando como argumento do método *frame* o valor 0, o quadro retornado será o mais recente, no caso de se passar o valor 1, o objeto retornado será o quadro anterior, e assim por diante. O número máximo de histórico de quadros disponíveis são 59 quadros, por padrão, não fornecendo o argumento será retornado o quadro atual. Porém, caso forneça um valor maior que 59, ou referente a algum quadro indisponível, será retornado um quadro não valido.

A classe *Frame* representa um conjunto de dados referentes ao mapeamento de mãos e dedos em um único quadro. Esses objetos podem ser acessados através dos métodos *hand* e *fingerList* , o primeiro retorna um objeto do tipo *Hand*. É necessário um parâmetro do tipo inteiro que fará referência a mão mapeada pelo controlador, o índice especificado é distribuído por ordem de visualização e construção do objeto. Já o segundo retorna uma lista de dedos, ou seja, uma lista contendo objetos do tipo *Finger*. Em alguns casos não é recomendado o acesso dos objetos relacionados aos dedos através do *Frame*, pois os mesmos não indicam a qual mão pertence o objeto. Nestes casos obtém-se a lista de dedos através do objeto *Hand*.

A classe *Hand* relata as características físicas de uma mão mapeada pelo controlador. Com o objeto *Hand* pode-se acessar a posição, velocidade, ângulo e direção da palma da mão, além da lista de dedos que compõem a mão mapeada. Um método bastante importante dessa classe é denominado como *isLeft*. Ele verifica se o objeto *Hand* mapeado faz referência a mão esquerda, com isso, pode-se saber facilmente qual mão está sendo utilizada.

A classe *Finger* faz referência aos dedos mapeados pelo controlador, estes são objetos apontáveis que se permitem caracteriza-los de acordo com a noção do controlador. O tipo do dedo analisado pode ser retornado através do método *type*, obtendo um tipo anatômico do dedo de forma enumerada:

- *TYPE_THUMB*: estado do dedo mapeado, caso o mesmo seja o dedo polegar;
- *TYPE_INDEX:* estado do dedo mapeado, caso o mesmo seja o dedo indicador;
- *TYPE_MIDDLE*: estado do dedo mapeado, caso o mesmo seja o dedo médio;
- *TYPE_RING*: estado do dedo mapeado, caso o mesmo seja o dedo anelar;
- *TYPE_PINKY*: estado do dedo mapeado, caso o mesmo seja o dedo mindinho.

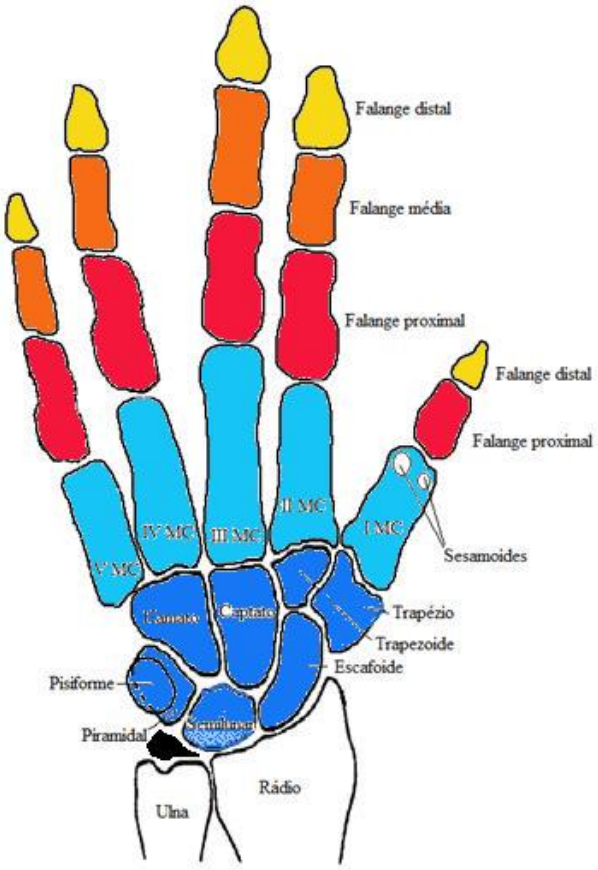
Para obter uma maior precisão em relação aos movimentos dos dedos, o dispositivo consegue mapear os ossos dos dedos. Sabe-se que todos os dedos com exceção do polegar possuem quatro ossos que compõem a sua anatomia:

- Falange distal: indica as extremidades distais dos dedos;
- Falange média: se encontram no meio dos dedos entre a falange distal e a proximal;
- Falange proximal: falanges que possuem articulações com os ossos metacarpais;
- Metacarpo: osso localizado entre as falanges e o carpo.

Para resolver a falta do osso intermediário nas falanges do dedo polegar, o controlador considera que o osso metacarpiano é o faltante, e atribui um osso válido de

comprimento zero em seu lugar. A localização do inicio e termino do osso em determinado dedo está acessível através dos métodos *prevJoint* e *nextJoint* no objeto *Bone*. O primeiro retorna as coordenadas de onde se inicia o osso consultado, e o segundo as coordenadas de onde o mesmo termina. . A Figura 36 permite a visualização anatômica da divisão óssea da mão humana.

Figura 36 – Divisão óssea da mão humana.

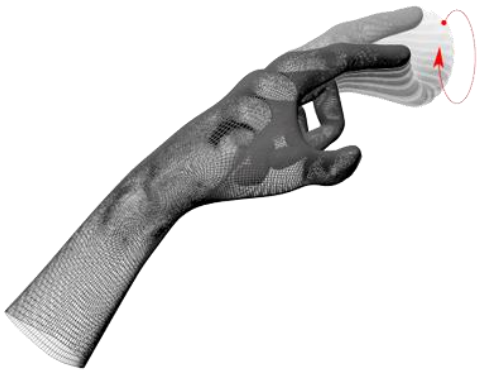


Fonte: COELHO; DANTAS e CUNHA (2012) – Modificada pelo autor.

Em primeiro momento, a fim de elaborar os sinais propostos em LIBRAS com o dispositivo Leap Motion, foi utilizada a classe *Gestures*, pertencente à biblioteca do dispositivo. Alguns padrões de movimento são reconhecidos pelo software do dispositivo, como movimentos que podem expressar uma intensão ou comando do usuário. Os padrões de movimento reconhecidos pelo dispositivo Leap Motion são:

- *Circle*: um único dedo trançando um circulo no espaço como demonstrado na Figura 37;

Figura 37 – Movimento *Circle* detectado pelo dispositivo como um gesto.



Fonte: LEAP MOTION, 2015.

- *Swipe*: um movimento longo e de forma linear, semelhante ao de deslizar na tela de um tablet ou smartphone. Esse gesto pode ser efetuado por qualquer dedo e em qualquer direção, como demonstrado na Figura 38;

Figura 38 – Gesto furtivo de forma horizontal.

Fonte: LEAP MOTION, 2015.

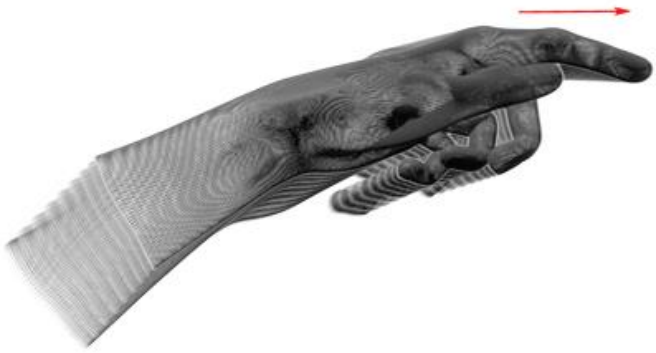
- *Key Tap*: movimento de toque com um dedo, similar a um toque na tecla de um piano. A Figura 39 permite a visualização do gesto *Key Tap*;

Figura 39 – Gesto de toque para baixo com o dedo indicador.

Fonte: LEAP MOTION, 2015.

- *Screen Tap*: movimento de batida com o dedo, esse gesto é caracterizado pelo movimento de empurrar para frente no espaço, como se estivesse tocando a tela de um computador na vertical. A Figura 40 permite a visualização do gesto *Screen Tap*.

Figura 40 – Gesto de toque de tela com o indicador.



Fonte: LEAP MOTION, 2015.

A ideia inicial era obter alguns sinais de LIBRAS com base nos gestos apresentados acima, ou elaborados através dos ângulos de abertura dos dedos em relação ao dedo vizinho e o centro da palma da mão. Para obter um controle mais abrangente destes ângulos, foi pretendido utilizar uma lógica nebulosa. Assim teríamos ângulos com valores fuzzificados: muito pequeno, pequeno, regular, grande e muito grande.

Este método foi descartado após analisar o alto grau de empenho para poder criar um gesto. Levou-se em consideração o desejo de obter uma maneira rápida em que os sinais em LIBRAS fossem cadastrados.

Para mitigar este problema, foi estudada a possibilidade de utilizar uma RNA do tipo MLP para efetuar a classificação dos sinais em LIBRAS. Para obter uma classificação eficiente, os valores que serão imputados na RNA devem passar por uma seleção, pois alguns dados podem piorar, ou não acrescentar de forma relativa no resultado. Analisando o movimento da mão em realizar alguns sinais de LIBRAS, percebeu-se que os movimentos das pontas dos dedos (Falanges distais) tinham fortes indícios em servirem como entrada da RNA.

O dispositivo Leap Motion trabalha com informações tridimensionais, referentes aos pontos das mãos. É necessário ter bastante cautela para informar esses dados como entrada na rede neural, pois se a mão estiver um pouco mais a esquerda, direita, acima ou a frente do dispositivo, mesmo com movimentos semelhantes, uma coordenada totalmente diferente é gerada.

A fim de centralizar a mão, sem a importância da sua posição para com o dispositivo, a palma da mão foi tida como o ponto zero das coordenadas. Calculou-se a distância da coordenada referente à falange distal de cada dedo para com a coordenada referente à palma da mão. Neste contexto, pode ser determinada pela Fórmula 09. Com isso, a RNA ficou com um total de cinco neurónios de entrada, cada um fazendo referência a um respectivo dedo.

/

$$Distância = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2 + (z_2 - z_1)^2}$$

Fórmula 09: Fórmula da Distância Entre Dois Pontos.

Pensando em unificar o tamanho das mãos para que esse atributo não afete na classificação do sinal, foi necessário normalizar os valores resultantes da função que calcula a distância entre os pontos, em valores com grandeza entre 0 e 1. Foi então utilizada a função min-max que consiste em converter todos os valores de um atributo para o intervalo [0,1] ou [-1,1] (TESCHL, 2010). Neste contexto, pode ser determinada pela Fórmula 10, onde x é o valor a ser normalizado, min_x é o menor valor de x e max_x é o maior de x.

$$x' = \frac{x - min_x}{max_x - min_x}$$

Fórmula 10: Fórmula da normalização min-max.

Para atestar que as entradas selecionadas seriam adequadas ao trabalho, foi criado um vetor bidimensional de cinco posições. Cada posição do vetor foi preenchida com a normalização da distância calculada pelos pontos referente aos dedos e a palma da mão. Em seguida utilizou-se da distância cosseno para calcular a similaridade entre dois gestos que estavam alocados em vetores diferentes. Com este teste, observou-se que gestos diferentes tinham a sua similaridade bem próxima de 0, enquanto que gestos iguais, se aproximavam de 1. A Tabela 6 permite a visualização dos valores que compõe os dedos quando sinalizam a letra A e B. A Tabela 7 demonstra os valores dos dedos referentes ao sinal da letra A, efetuado por pessoas diferentes.

Tabela 6 – Valores dos dedos referente às Letras A e B.

Sinal	Polegar	Indicador	Médio	Anelar	Mindinho
Letra A	1.0	0.3501542	0.3117259	0.1045289	0.0
Letra B	0.0	0.8367762	1.0	0.8580527	0.5389103

Tabela 7 – Valores dos dedos referentes à Letra A em mãos diferentes.

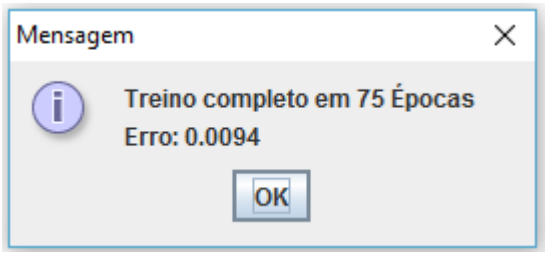
Sinal	Polegar	Indicador	Médio	Anelar	Mindinho
Letra A	1.0	0.3501542	0.3117259	0.1045289	0.0
Letra A	1.0	0.320421	0.29542	0.1043200	0.0

Aplicando a distância cosseno em cima dos valores informados na Tabela 6, obtêm-se como resultado o valor 0.37906268, o que implica que a comparação entre os dois gestos tem um valor de similaridade baixo. Entretanto, aplicando a distância cosseno para com os valores da Tabela 7, obtêm-se como resultado o valor 0.99960256, implicando em uma alta similaridade entre os dois conjuntos.

Para a criação e treinamento da RNA, utilizou-se a estrutura de redes neurais *Neuroph*, que pode ser usada para criar e treinar RNAs em programas Java (NEUROPH, 2014). A RNA ficou da seguinte forma: cinco neurônios de entrada, dez neurônios

intermediários e vinte neurônios de saída, seu treinamento levou 75 Épocas e uma taxa de erro de 0.0094. A Figura 41 permite a visualização do alerta exibido após o treinamento da RNA.

Figura 41 – Alerta de conclusão de treinamento da RNA.



Fonte: Elaborada pelo autor.

Com a RNA treinada, sempre que um novo quadro de mãos e dedos estiver disponível para leitura, obtêm-se os dados necessários para o calculo dos valores que serão imputados pela RNA, antes disso, esses valores são normalizados. Tem-se como resultado da RNA, um vetor de 20 posições referente às 20 letras estáticas do alfabeto em LIBRAS. Os valores contidos no vetor variam de 0 a 1, onde quanto mais próximo de 0, menor é a precisão de classificação para com aquela saída, o contrário é valido para quando o valor estiver próximo de 1.

Como a taxa de captura do Leap Motion pode alcançar até 200 fps, implicando em até 200 solicitações de classificação para a RNA, foi proposto à diminuição da quantidade de *frames* capturados. O Leap Motion não possui nenhum parâmetro para controle da quantidade de quadros a serem disponibilizados. Por está razão, foi estabelecido que o algoritmo retivesse somente três quadros por segundo, ignorando todos os outros quadros disponíveis pelo dispositivo.

3.1.3. Funcionalidades

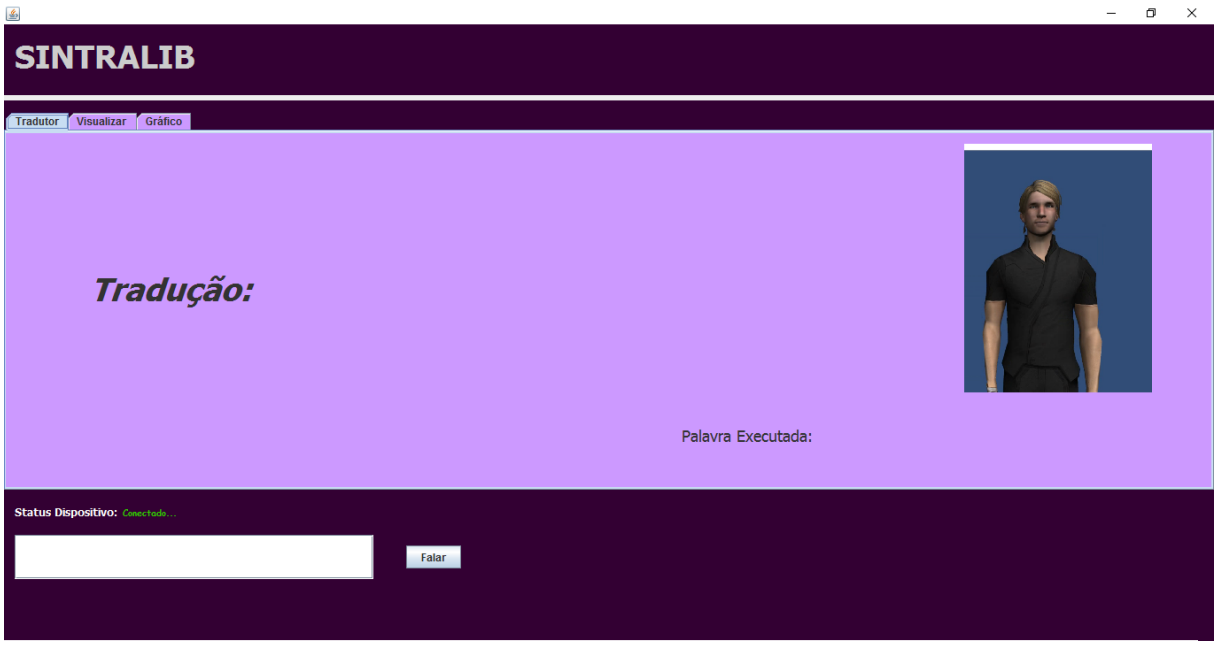
O SINTRALIB foi planejado para em versões futuras possibilitar uma conversação entre ouvintes e deficientes auditivos. Sabe-se também que os deficientes auditivos têm facilidade em distraírem com aspectos visuais chamativos, com isso, a interface do sistema deve ser simples e visualmente agradável, com foco nos elementos característicos a passarem informação. A Figura 42 permite a visualização da tela inicial do SINTRALIB.

Figura 42 – Tela inicial do SINTRALIB.

Fonte: Elaborada pelo autor.

Assim que o Leap Motion é conectado ao computador, o sistema irá reconhecê-lo e mudar o status do dispositivo para “Conectado” em uma tonalidade verde. A Figura 43 permite a visualização da mudança de status do dispositivo.

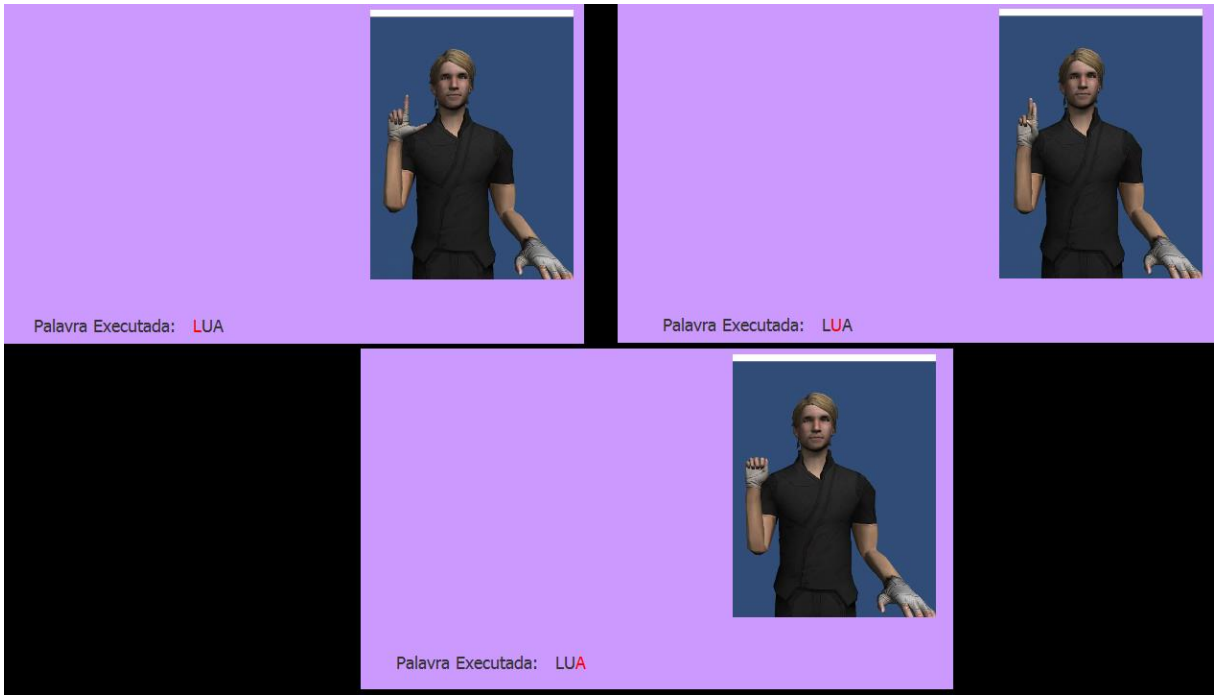
Figura 43 – Mudança de status do dispositivo ao conectar o Leap Motion.



Fonte: Elaborada pelo autor.

Quando o usuário digita, ou fala alguma expressão da língua portuguesa e que esteja na gramática do SINTRALIB, o avatar Andy trata de traduzir as letras que formam a expressão para LIBRAS. Dar-se ênfase que não é necessária a conexão do dispositivo Leap Motion para o funcionamento do tradutor português - LIBRAS. A Figura 44 permite a visualização da animação “soletrando” em LIBRAS a palavra “lua”, onde se destaca em vermelho a letra que está sendo executada pelo avatar no momento atual.

Figura 44 – Avatar ANDY “soletrando” a palavra “lua” em LIBRAS.



Fonte: Elaborada pelo autor.

Na aba “Visualizar”, o usuário pode obter a informação no formato de barras de progresso sobre a classificação do sinal efetuado, ou seja, existe uma barra de progresso para cada letra disponível, a mesma demonstrará em um percentual de 0 a 100%, qual a classificação da RNA em relação ao sinal capturado. De 0 a 30%, a barra de progresso possui uma cor avermelhada, de 31 a 70%, a barra de progresso possui uma cor de tonalidade amarela, e de 71 a 100%, sua cor ficará verde. A Figura 45 demonstra a aba “Visualizar” com o usuário sinalizando a letra D. A letra com maior percentual de classificação será exibida na aba “Tradutor”, se, somente se, este percentual for maior que 70%. A Figura 46 permite a visualização da aba “Tradutor”, no momento em que o usuário sinaliza a letra D.

Figura 45 – Aba “Visualizar” com o usuário sinalizando a letra D.

Fonte: Elaborada pelo autor.

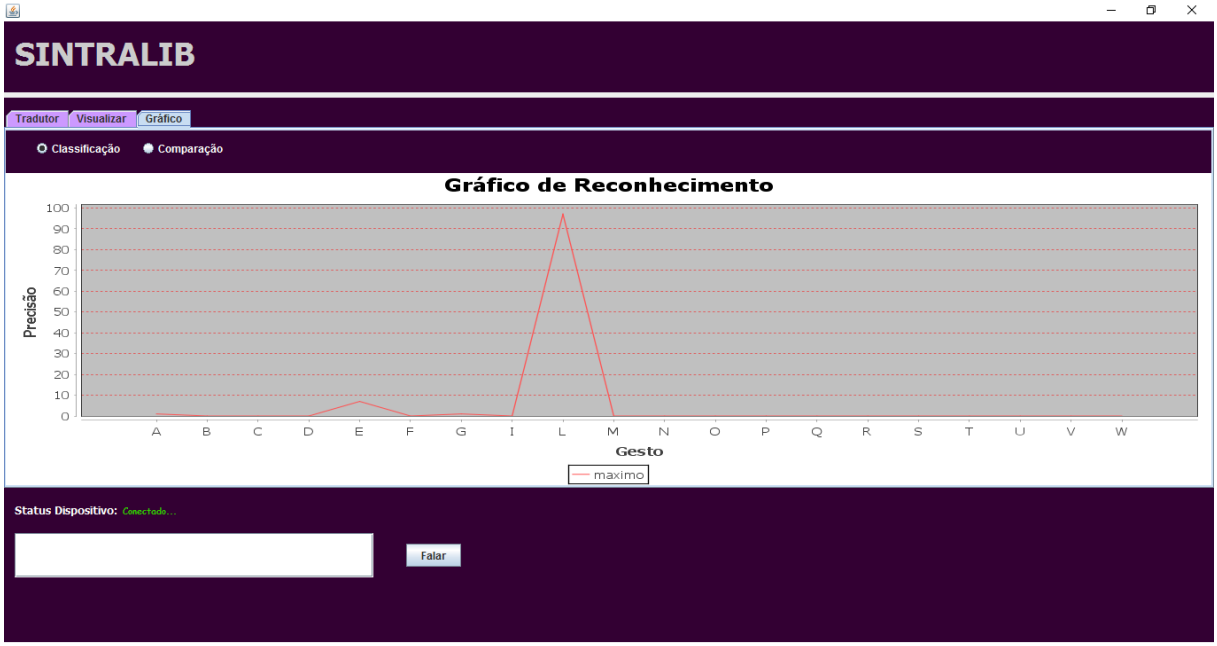
Figura 46 – Aba “Tradutor” com o usuário sinalizando a letra D.

Fonte: Elaborada pelo autor.

O SINTRALIB permite a visualização da taxa de classificação referente a cada letra disponível no sistema em forma de gráfico. Essa funcionalidade é bastante importante para compreender o nível de classificação da RNA, ou seja, pode-se visualizar a taxa de

classificação de todas as letras após efetuar um sinal, além de poder averiguar se a RNA está disparando sinais falsos positivos em relação ao gesto efetuado pelo usuário. A Figura 47 permite a visualização da taxa de classificação em forma gráfica, que se encontra na aba “Gráfico”, com o usuário sinalizando a letra L.

Figura 47 – SINTRALIB exibindo o gráfico de classificação com o usuário sinalizando a letra L.

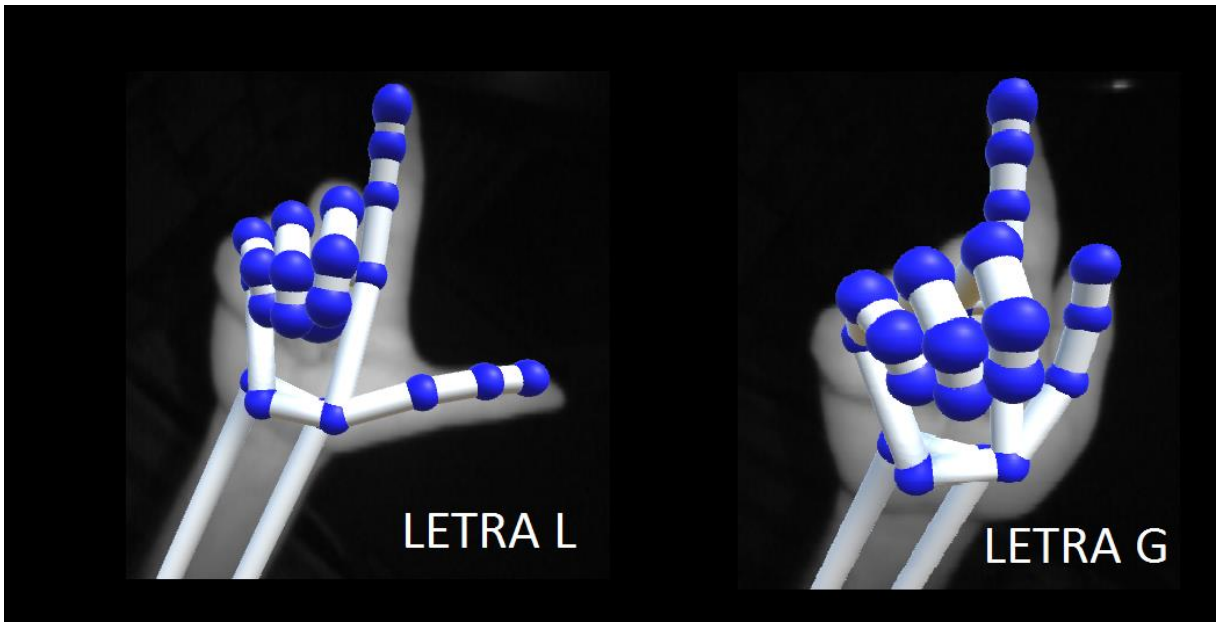


Fonte: Elaborada pelo autor.

O SINTRALIB permite que o usuário compare o gesto sinalizado com outros cadastrados na base de dados. A posição dos dedos capturados do usuário é comparada com a posição dos dedos referente aos sinais selecionados para comparação, isso facilita a análise dos dados que podem influenciar a RNA na classificação de falsos-positivos. Essa informação é exibida no formato de gráfico, onde o SINTRALIB permite a comparação do gesto atual com uma ou mais letras. A Figura 49 demonstra a visualização do gráfico de comparação exibido pelo SINTRALIB, onde o usuário faz o sinal da letra G e compara com as letras G e L que se encontram na base de dados. Já na Figura 50, o usuário executa o sinal da letra L e compara com os sinais da letra G e L cadastradas na base de dados. As letras G e L têm posições de dedos similares, exceto pelo dedo polegar, onde quando efetuado o sinal da letra G, seu ângulo de abertura para com o dedo indicador é relativamente menor quando

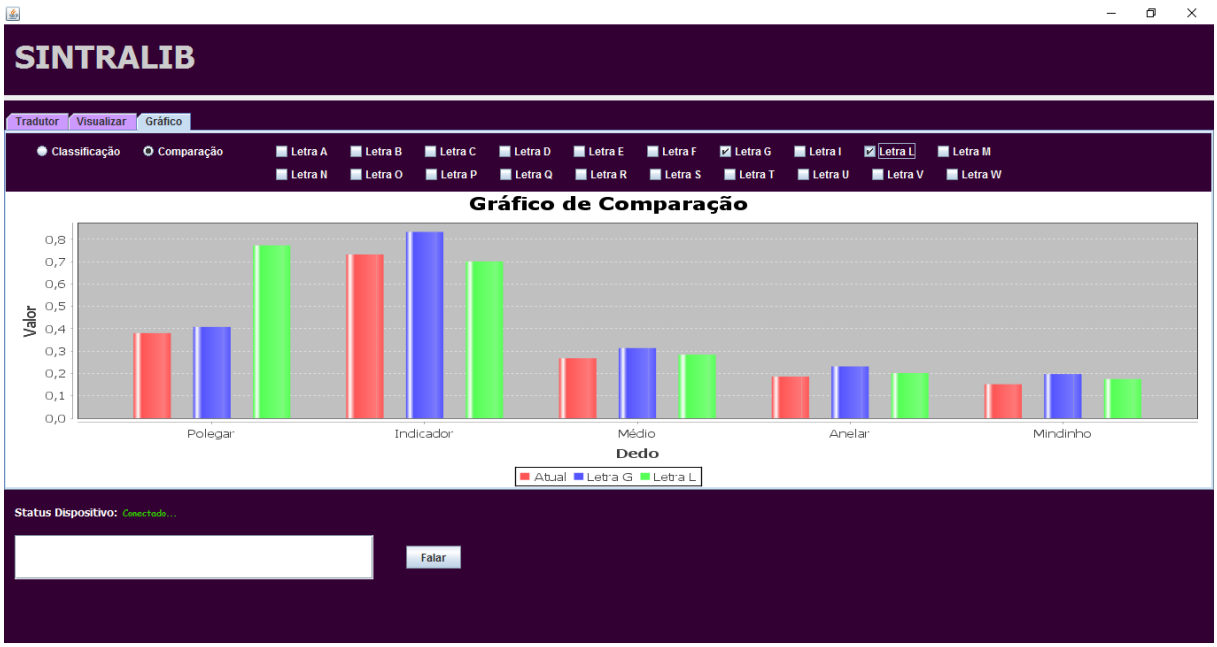
comparado com o ângulo de abertura da letra L. A Figura 48 demonstra a diferença entre o sinal da letra G e o sinal da letra L, onde a letra G encontra-se a direita e a letra L à esquerda.

Figura 48 – Demonstração da letra G e L no visualizador 3D do Leap Motion.



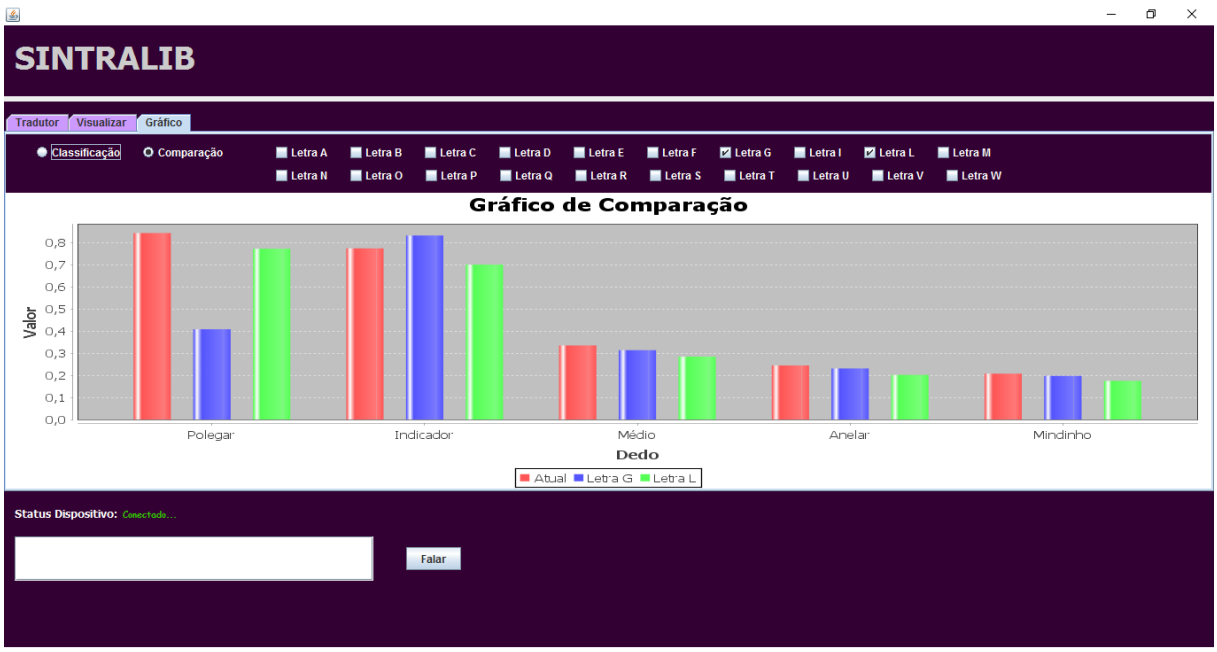
Fonte: Elaborada pelo autor.

Figura 49 – SINTRALIB exibindo o gráfico de comparação entre o sinal executado pelo usuário (Letra G) e as letras G e L da base de dados.



Fonte: Elaborada pelo autor.

Figura 50 – SINTRALIB exibindo o gráfico de comparação entre o sinal executado pelo usuário (Letra L) e as letras G e L da base de dados.



Fonte: Elaborada pelo autor.

Analisando os gráficos de comparação das Figuras 48 e 49, foram visualizados cinco grupos denominados “Polegar”, “Indicador”, “Médio”, “Anelar” e “Mindinho”, que fazem referência ao valor da posição dos dedos com estes respectivos nomes. Estes grupos estão formados por barras de três cores: vermelha, azul e verde. A barra vermelha representa a posição atual dos dedos do usuário, a barra azul representa a posição dos dedos referente à letra G, e a barra verde representa a posição dos dedos referente à letra L. No gráfico da Figura 48, quando o usuário sinaliza a letra G, fica notória a aproximação da barra vermelha para com os dedos dos dois gestos, exceto o dedo polegar, que se aproxima mais do valor referente ao dedo polegar da letra G, classificando o gesto como letra G. Já na Figura 49, quando o usuário sinaliza a letra L, o dedo polegar se aproxima mais do valor referente ao dedo polegar da letra L, implicando na classificação da letra L.

4. TESTES E RESULTADOS

Neste capitulo serão apresentados os testes realizados sobre o sistema SINTRALIB, primeiro será descrita a base de dados utilizada, os modelos de teste aplicados, em seguida serão apresentados os resultados obtidos, e finalmente será feita uma avaliação sobre os mesmos.

4.1. Base de dados

A base de dados utilizada para treinamento da RNA aplicado ao SINTRALIB está disponível no site https://www.4shared.com/office/63Y3_jtyei/entradasF.html. Nesta base estão contidos os dados dos sinais que representam as letras do alfabeto em LIBRAS, com exceção das letras H, J, K, X, Y e Z. Este trabalho tem como proposta utilizar somente as letras do alfabeto cujos sinais são estáticos, as letras enquadradas como exceção possuem movimento e não fazem parte do escopo estabelecido.

A base de dados é formada por 240 amostras, que por sua vez é dividida em 12 amostras para cada um dos 20 sinais. A fim de evitar um possível vício na RNA, os sinais foram capturados de quatro pessoas distintas (conhecedores de LIBRAS), nas quais contribuíram com três poses para cada sinal. Os dados das amostras são separados por uma vírgula, e organizados da seguinte forma: as cinco primeiras posições são referentes aos dedos (polegar, indicador, médio, anelar e mindinho), as vinte posições restantes fazem referência as 20 letras estáticas do alfabeto em LIBRAS, 19 destas 20 posições recebem o valor 0 e apenas a posição referente a letra cadastrada recebe o valor 1.

4.2. Testes aplicados

Para validar o SINTRALIB, dividiu-se o sistema em duas versões: uma com 10 letras estáticas de A até M, e a outra com 20 letras estáticas de A até W. Foram efetuados dois testes para cada versão: no primeiro, o sistema classificou as entradas utilizadas para o treinamento da RNA, a fim de verificar se os dados utilizados para treinar a rede implicaram ou não em um aprendizado consistente, já o segundo teste foi utilizado para mensurar a acurácia do sistema quando utilizado por pessoas conhecedoras de LIBRAS.

Foram selecionados 10 participantes que frequentaram a turma de LIBRAS ou que possuem conhecimento em nível de expressar sinais no idioma. Todos os participantes do

teste tiveram que preencher um termo de consentimento livre e esclarecido que pode ser visualizado no Anexo I, encontrado na sessão de anexos deste trabalho. Para uma melhor avaliação dos resultados obtidos no teste com os participantes, foram utilizados alguns formulários que se encontram no Anexo II da sessão de anexos. No Formulário II e Formulário III, o participante deve executar o sinal de 10 letras em ordem definida pelo mesmo. Para cada sinal executado na ordem definida, deve ser preenchida, em mesma ordem, a letra classificada pelo sistema no quadro de resultados referente ao formulário em execução. Na concretização dos sinais referentes às letras escolhidas de cada formulário, é registrado o tempo de inicio e término da execução total do teste naquele formulário, com isso pode-se ter uma média de tempo em que os participantes concluíram este teste.

Os resultados obtidos serão apresentados no formato de uma matriz de confusão. A diagonal principal da matriz é constituída por resultados verdadeiro-positivos, e a acurácia é dada pela média aritmética dos resultados que formam a diagonal da matriz.

4.2.1. Resultados com 10 letras

Foram efetuados dois testes para a versão SINTRALIB com 10 letras. No primeiro foi classificada a base de dados que serviu para treinar a RNA, e no segundo o sistema classificou os sinais elaborados por usuários conhecedores de LIBRAS.

- Teste 1: A acurácia média obtida neste teste é de 100%. Através da matriz de confusão que se encontra na Tabela 8, pode-se visualizar os dados que levaram a este resultado.

Tabela 8 – Matriz de confusão da classificação dos 10 sinais em cima dos dados utilizados para treinar a RNA.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**I**	**L**	**M**
A	100	0	0	0	0	0	0	0	0	0
B	0	100	0	0	0	0	0	0	0	0
C	0	0	100	0	0	0	0	0	0	0
D	0	0	0	100	0	0	0	0	0	0
E	0	0	0	0	100	0	0	0	0	0

F	0	0	0	0	0	100	0	0	0	0
G	0	0	0	0	0	0	100	0	0	0
I	0	0	0	0	0	0	0	100	0	0
L	0	0	0	0	0	0	0	0	100	0
M	0	0	0	0	0	0	0	0	0	100

- Teste 2: A acurácia média obtida neste teste é de 87%. Através da matriz de confusão que se encontra na Tabela 9, pode-se visualizar os dados que levaram a este resultado.

Tabela 9 – Matriz de confusão contendo os resultados obtidos com os usuários através do Formulário II, contido no Anexo II.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**I**	**L**	**M**
A	100	0	0	0	0	0	0	0	0	0
B	0	90	10	0	0	0	0	0	0	0
C	0	0	100	0	0	0	0	0	0	0
D	0	0	0	80	10	10	0	0	0	0
E	0	0	0	10	80	0	0	0	0	10
F	0	0	0	0	0	100	0	0	0	0
G	0	0	0	0	0	0	90	0	10	0
I	10	0	0	0	0	0	0	90	0	0
L	0	0	0	0	0	0	0	0	100	0
M	0	15	30	0	0	0	15	0	0	40

4.2.2. Resultados com 20 letras

Foram efetuados dois testes para a versão SINTRALIB com 20 letras. No primeiro foi classificada a base de dados que serviu para treinar a RNA, e no segundo o sistema classificou os sinais elaborados por usuários conhecedores de LIBRAS.

- Teste 1: A acurácia média obtida neste teste é de 95,41%. Através da matriz de confusão que se encontra na Tabela 10, pode-se visualizar os dados que levaram a este resultado.

Tabela 10 – Matriz de confusão da classificação dos 20 sinais em cima dos dados utilizados para treinar a RNA.

	A	B	C	D	E	F	G	I	L	M	N	O	P	Q	R	S	T	U	V	W
A	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
B	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
E	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
F	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0	0
L	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0
M	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0.4	91.6	0	0	0	0	0	0	0	0	0
O	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0
P	0	0.4	0	0	0	0	0	0	0	0	0	0	91.6	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0
T	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0
U	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	25	75	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100

- Teste 2: A acurácia média obtida neste teste é de 67,05%. Através da matriz de confusão que se encontra na Tabela 11, pode-se visualizar os dados que levaram a este resultado.

Tabela 11 – Matriz de confusão contendo os resultados obtidos com os usuários através do Formulário III, contido no Anexo II.

	A	B	C	D	E	F	G	I	L	M	N	O	P	Q	R	S	T	U	V	W
A	87	0	0	0	0	0	0	0	0	0	0	0	0	12	0	0	0	0	0	0
B	0	100	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
C	0	0	87	0	0	12	0	0	0	0	0	0	0	0	0	0	0	0	0	0
D	0	0	0	50	16	0	0	0	16	0	0	0	0	0	0	0	16	0	0	0
E	0	0	0	0	66	0	0	0	16	0	0	0	0	16	0	0	0	0	0	0
F	0	0	0	0	33	66	0	0	0	0	0	0	0	0	0	0	0	0	0	0
G	0	0	0	0	0	0	75	0	0	0	0	0	25	0	0	0	0	0	0	0
I	0	0	0	0	0	0	0	87	0	0	0	0	0	0	0	12	0	0	0	0
L	0	0	0	0	0	0	0	0	100	0	0	0	0	0	0	0	0	0	0	0
M	0	0	33	0	0	0	0	0	0	66	0	0	0	0	0	0	0	0	0	0
N	0	0	0	0	0	0	0	0	0	0	50	0	0	0	0	0	0	0	0	50
O	0	0	0	0	0	25	0	0	0	0	0	50	0	25	0	0	0	0	0	0
P	0	25	0	0	0	0	25	0	0	0	0	25	25	0	0	0	0	0	0	0
Q	0	0	0	0	0	0	25	0	25	0	0	0	0	25	0	25	0	0	0	0
R	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0	0	0	0	0
S	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	75	0	0	0	0
T	0	0	0	33	0	0	0	0	0	0	0	0	0	0	0	0	66	0	0	0
U	0	0	0	0	0	0	33	0	0	0	0	0	0	0	0	16	0	0	50	0
V	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	100	0
W	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	66

5. CONCLUSÃO

O reconhecimento de linguagens de sinais através de sistemas computacionais é algo bastante promissor e vem ganhando força internacional.

Diferentes maneiras são utilizadas para o reconhecimento de sinais, desde luvas computacionais, câmeras comuns, até dispositivos mais sofisticados como o kinect, Leap

Motion e *Creative Senz* 3D. Métodos mais tradicionais utilizam imagens RGB com um fundo uniforme para obter informações mais precisas sobre o mapeamento da mão.

A tradução textual para língua de sinais é tida como mais simples em relação ao contrário. Diversos trabalhos encontrados abordaram de forma parecida a tradução de textos para LIBRAS. Vários desses trabalhos encontram-se disponíveis e contribuíram bastante na inclusão social de deficientes auditivos, com destaque para os aplicativos Hand Talk e ProDeaf.

A principal contribuição deste trabalho foi conseguir fazer a tradução em ambas as direções, de LIBRAS para o português e de português para LIBRAS, ainda que o atual escopo tenha sido apenas de letras do alfabeto com gestos estáticos. O sistema desenvolvido possibilitou a troca de informações entre surdos e ouvintes. Com isso, avançou-se tecnologicamente colaborando para o desenvolvimento de futuros trabalhos. É intensão dos autores, posteriormente, continuar o desenvolvimento do sistema de modo que o mesmo venha a traduzir palavras e frases bidireccionalmente.

Neste trabalho, utilizou-se do dispositivo Leap Motion em conjunto com uma RNA para efetuar a tradução de letras estáticas do alfabeto em LIBRAS para o português. Também foi utilizado o IBM ViaVoice, em conjunto com o *JSAPI*, para o reconhecimento de voz.

Diversas dificuldades, com medidas diferentes, foram encontradas em ambas as traduções. Na tradução de português para LIBRAS, o reconhecimento de voz na plataforma Java é bastante limitado. São poucas e descontinuadas as bibliotecas que trabalham com essa funcionalidade. Muitos softwares, que trabalham em conjunto com essas bibliotecas, são tidos como arcaicos e necessitam de ferramentas extras para poderem ser utilizados em computadores modernos. Utilizar uma animação 3D exportada pelo *Blender* foi bastante trabalhoso. A maneira utilizada para chamar a animação referente à letra reconhecida obteve um *delay* muito grande e ficou inviável para este trabalho, tendo como solução a criação de um vídeo contendo todas as animações e carregando o mesmo em um *player* de vídeo incorporado a aplicação.

O resultado dos testes efetuados na tradução LIBRAS-português demonstrou uma maior taxa de acurácia quando se trabalhou com 10 letras. Na versão do SINTRALIB, com 20 letras, a taxa de acurácia reduziu de forma significativa. Algumas letras tiveram seus dados reconhecidos como bastante similares a outras letras, o que implicou na classificação de

falsos-positivos, ou um percentual baixo de classificação da RNA. Este problema poderia ser mitigado com uma quantidade maior de variáveis de entrada para a RNA, podendo ser acrescentado como variável a distância relativa entre outros ossos dos dedos, ou até mesmo a distância relativa de um dedo para com outro, fazendo com que os dados entre letras distintas divergissem de forma que a RNA classificasse, corretamente, uma quantidade maior de letras.

O Leap Motion demonstrou ser bom no que propõe: mapeamento de braços e mãos. Nas primeiras versões do seu software, a precisão demonstrou ser relativamente menor. Na medida em que novas atualizações foram disponibilizadas, o seu poder de captura de movimentos ficou cada vez maior. Entretanto, existem ainda muitos pontos a serem ajustados. A luminosidade tem certa influência na captura dos movimentos. Por mais que sua câmera de profundidade seja infravermelha, e seu software compense com ajustes de luminosidade, há incoerência nos dados obtidos, mesmo num ambiente muito luminoso.

A LIBRAS não se faz somente dos movimentos das mãos para elaborar vários dos seus sinais. São utilizados, também, expressões faciais e movimentos do corpo. Com isso, não existe a possibilidade de traduzir palavras somente com o Leap Motion, é necessária a inclusão de outro hardware para a captura dos movimentos e expressões faciais.

5.1. Trabalhos Futuros

Em consequência deste trabalho, é possível enxergar os seguintes trabalhos futuros:

- Tradução de letras que possuem movimento do alfabeto em LIBRAS;
- Aumentar a acurácia na classificação das letras com o crescimento de variáveis de entrada para a RNA;
- Vincular um novo hardware que capture movimentos do corpo e expressões faciais para auxiliar o Leap Motion em traduzir palavras e expressões mais complexas de LIBRAS para o português;
- Traduzir frases de português para LIBRAS e de LIBRAS para português.

REFERÊNCIAS

ANDRADE, Isabela Freixo Côrtes de; LEWIS, Doris R. **A Negligência Mundial Sobre a Deficiência Auditiva Infantil em Países em Desenvolvimento**. 2007. <Disponível em

http://revistas.pucsp.br/index.php/dic/article/viewFile/6825/4944. Acesso em 14 de fev. 2017>.

ATLASSIAN. **Codifique, Gerencie, Colabore**. 2017. <Disponível em https://br.atlassian.com/software/bitbucket. Acesso em 21 de abr. 2017>.

BARRETO, Jorge M. **Introdução às Redes Neurais Artificiais**. Laboratório de Conexionismo e Ciências Cognitivas – Departamento de Informática, Universidade Federal de Santa Catarina, Florianópolis, 2002.

BENICASA, Alcides Xavier. **Sistemas Computacionais para Atenção Visual Top-Down e Bottom-UP usando Redes Neurais Artificiais**. 2013. <Disponível em http://www.teses.usp.br/teses/disponiveis/55/55134/tde-29042014-162209/pt-br.php. Acesso em 23 de nov. 2016>.

BRAGA, Daniela. **Máquinas Falantes: Novos Paradigmas da Língua e da Linguística – Microsoft Language Development Center**. 2007. <Disponível em http://download.microsoft.com/download/A/0/B/FA0B1A66A-5EBF-4CF3-9453-4B13BB027F1F/ColoquioPoliticaLinguistica_2007.pdf. Acesso em 18 de abr. 2017>.

CAMPOS, JOSÉ R et al. **Implementação de Redes Neurais Artificiais Utilizando a Linguagem de Programação JAVA**. 9º Conference on Dynamics, Control and their Aplications – Universidade Estadual de São Paulo, Ilha Solteira, 2010.

CARDON, André; MULLER, Daniel N. **Introdução Às Redes Neurais Artificiais**. Dissertação (Pós Graduação em Ciência da Computação) – Instituto de Informática, Universidade Federal do Rio Grande do Sul, Porto Alegre, 1994.

CARVALHO, Paulo V. **Breve História dos Surdos no Mundo e em Portugal**. Lisboa: Surd'Universo, 2007.

CARVALHO, André Ponce de Leon F. de. **Redes Neurais Artificiais**. Departamento de Ciência da Computação - USP. São Carlos, 2009. <Disponível em http://conteudo.icmc.usp.br/pessoas/andre/research/neural/. Acesso em 19 de abr. 2017>.

COELHO, Renato Ramos; DANTAS, Estélio Henrique Martin; CUNHA, Gerson Gomes. **USO DO CONJUNTO BENDSENSOR/ARDUINO PARA A CAPTURA DE MOVIMENTO DO PUNHO-MÃO**. 2012. <Disponível em https://www.researchgate.net/publication/233953248_USO_DO_CONJUNTO_BENDSENSORARDUINO_PARA_A_CAPTURA_DE_MOVIMENTO_DO_PUNHO-MAO. Acesso em 14 de abr. 2017>.

CUNHA, Anderson de Andrade; ALVES, Guilherme Bruno Viturino; FARO, Mai-Ly Vanessa Almeida Saucedo. **SINTRALIBD - Sistema Integrado Tradutor de LIBRAS - Comunicação para todos**. 2014. <Disponível em http://www.itatechjr.com.br/arquivos/SEMINFO_UFSITA_2014_2.pdf. Acesso em 14 de fev. 2017>.

DAMASCENO, Rafael H.F. et al. **Cartilha LIBRAS - Sinais de Inclusão**. Universidade José do Rosário Vellano - UNIFENAS. Alfenas, 2010. <Disponível em http://www.unifenas.br/extensao/cartilha/CartilhaLIBRAS.pdf. Acesso em 14 de ago. 2016>.

DOSS, Shwetha et al. **Developing applications using Intel® Perceptual Computing Technology**. <Disponível em https://software.intel.com/en-us/articles/developing-applications-using-intel-perceptual-computing-technology. Acesso em 30 de out. 2016>.

FALIBRAS. **FaLIBRAS® App**. 2001. <Disponível em http://www.faLIBRAS.org/nossa-hist-ria. Acesso em 04 de abr. 2017>.

FERREIRA, Fernando Henrique. **Desenvolvendo com o Kinect para Window**s. 2013. <Disponível em https://ferhenriquef.com/2013/04/23/desenvolvendo-com-o-kinect-para-o-windows/. Acesso em 18 de abr. 2017>.

FERNANDES, Gustavo de Oliveira. **Cálculo da Similaridade Entre Planilhas Eletrônicas**. (Dissertação de Mestrado, como parte dos requisitos necessários para obtenção do título de Mestre em Engenharia de Sistemas e Computação) – Programa de Pós-graduação em Engenharia de Sistemas e Computação, Universidade Federal do Rio de Janeiro, Rio de Janeiro. 2014.

FONSECA, L. M. G. **Processamento Digital de Imagens** – Instituo Nacional de Pesquisas Especiais (INPE), 2000. 105p.

HANDTALK. **Hand Talk® App**. 2016. <Disponível em https://handtalk.me/sobre/. Acesso em 20 de fev. 2017>.

IBGE, Instituto Brasileiro de Geografia e Estatística. **Censo Demográfico 2010 Características gerais da população, religião e pessoas com deficiência**. 2010. <Disponível em http://biblioteca.ibge.gov.br/visualizacao/periodicos/94/cd_2010_religiao_deficiencia.pdf. Acesso em 10 de out. 2016>.

INTEL CORPORATION. **Intel® RealSense™ SDK Details**. <Disponível em https://software.intel.com/en-us/intel-realsense-sdk/details. Acesso em 30 de out. 2016>.

LEAP MOTION. **LEAP MOTION® v2 Java SDK Documentation**. 2015. <Disponível em https://developer.leapmotion.com/documentation/v2/java/index.html. Acesso em 22 de out. 2016>

LACERDA, Cristina Broglia Feitosa de. **A Inclusão Escolar De Alunos Surdos: O Que Dizem Alunos, Professores e Interpretes Sobre Esta Experiência**. 2006. <Disponível em http://150.164.100.248/dialogosdeinclusao/data1/arquivos/Inclusao_Lacerda2006.pdf. Acesso em 19 de abr. 2017>.

MARQUES, Paulo César Florentino; ADVÍNCULA, Alessandro Bezerra; ALMEIDA, Isledna Rodrigues. **Reconhecimento e Sintetização de Voz Como Alternativa Para Software Inclusivo**. – XIII Jornada de Ensino, Pesquisa e Extensão – JEPEX, UFRPE. 2013.

<Disponível em http://www.eventosufrpe.com.br/2013/cd/resumos/R1187-3.pdf. Acesso em 18 de abr. 2017>

MARAGONI, Josemar Barone; PRECIPITO, Waldemar Barilli. **Reconhecimento e Sintetização de Voz Usando Java Speech**. Revista Cientifica Eletrônica de Sistemas de Informação – ISSN 1807-1872, Ano II, FAEG, 2006. <Disponível em http://faef.revista.inf.br/imagens_arquivos/arquivos_destaque/bjMnA2Zwc9685z8_2013-5-27-15-40-25.pdf. Acesso em 22 de fev. 2017>.

MARTINS, Ronaldo; PELIZZONI, Jorge; HASEGAWA, Ricardo. **PULØ - Para um sistema de tradução semi-automática português-LIBRAS.** XXV Congresso da Sociedade Brasileira de Computação – Universidade do Vale do Rio Sinos, São Leopoldo, 2005. <Disponível em http://nilc.icmc.sc.usp.br/til/til2005_English/arq0058.pdf. Acesso em 03 de abr. 2017>

MATOS, Pablo Freire et al. **Um Ambiente Para Análise de Dados da Doença Anemia Falciforme**. – USP, UFSCar, Unimep, São Carlos. 2009. <Disponível em http://conteudo.icmc.usp.br/pessoas/taspardo/TechReportUFSCar2009a-MatosEtAl.pdf. Acesso em 12 de mar. 2017>.

MELO, Alain Rosembergue Lívio de. **UM ESTUDO SOBRE O MAPEAMENTO DE GESTOS DO LEAP MOTION PARA A LÍNGUIA BRASILEIRA DE SINAIS (LIBRAS)**. (Dissertação de Mestrado para o grau de Mestre em Ciência da Computação) – Centro de Informática, Universidade Federal de Pernambuco, Recife, 2013.

MICROSOFT. **KINECT® FOR WINDOWS**, 2013. <Disponível em https://developer.microsoft.com/pt-br/windows/kinect/. Acesso em 29 de out. 2016>.

MONTEIRO, Otaviano Martins; ANTUNES, Lucas Leite. **Estudo do Uso do Kinect Para Interpretação de Gestos Visando LIBRAS**. 13º Congresso Nacional de Iniciação Científica – Faculdade Anhanguera de Campinas, Campinas, 2013. <Disponível em http://conic-semesp.org.br/anais/files/2013/trabalho-1000015887.pdf. Acesso em 04 de nov. 2016>.

NETBEANS. **O que é o NetBeans?**. 2017. <Disponível em https://netbeans.org/index_pt_PT.html. Acesso em 21 de abr. 2017>.

NEUROPH. **Neuroph – Java Neural Network**. 2014. <Disponível em http://neuroph.sourceforge.net/index.html. Acesso em 15 de nov. 2016>.

NIED, Ademir. **Treinamento de Redes Neurais Artificiais Baseado em Sistemas de Estrutura Variável Com Taxa De Aprendizado Adaptativa**. (Tese de Doutorado submetida à Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Minas Gerais) – UFMG, 2007. <Disponível em http://www.ppgee.ufmg.br/defesas/852D.PDF. Acesso em 04 de nov. 2016>

OLIVEIRA, Felipe Francisco Ramos de; FERREIRA, Marlon Marques; FURST, Alexandre. **Estudo Da Usabilidade Nas Interfaces Homem-Máquina**. Revista Exacta – UniBH, Belo

Horizonte, 2013. <Disponível em http://revistas.unibh.br/index.php/dcet/article/download/1079/623. Acesso em 18 de abr. 2017>

ORACLE. **O que é a Tecnologia Java e porque preciso dela?**. 2009. <Disponível em https://www.java.com/pt_BR/download/faq/whatis_java.xml. Acesso em 21 de abr. 2017>.

PERDOMO, Leonardo; SIQUEIRA, Mozart Lemos de. **Reconhecimento de sinais estáticos de LIBRAS com *Support Vector Machines* usando Kinect**. (35º CTIC - Concurso de Trabalhos de Iniciação Científica) – CSBC, 2016.

PERROCA, Márcia Galan; GAIDZINSKI, Raquel Rapone. **Avaliando a confiabilidade interavaliadores de um instrumento para classificação de pacientes - Coeficiente Kappa***, 2003. <Disponível em http://www.scielo.br/pdf/reeusp/v37n1/09.pdf. Acesso em 19 de abr. 2017>.

PIRES, David da Silva. **Estimação de Movimento a partir de imagens RGBD usando homomorfismo entre grafos**. (Tese Apresentada ao Instituto de Matemática e Estatística da Universidade de São Paulo para obtenção do Titulo de Doutor em Ciências) – USP, 2012.

PRESIDÊNCIA DA REPUBLICA. **LEI Nº 10.436, DE 24 DE ABRIL DE 2002**, 2002. <Disponível em http://www.planalto.gov.br/ccivil_03/Leis/2002/L10436.htm. Acesso em 14 de out. 2016>.

PRINA, Bruno Zacuni; TRENTIN, Romario. **GMC: Geração de Matriz de Confusão a partir de uma classificação digital de imagem do ArcGIS ®** – XVII Simpósio Brasileiro de Sensoriamento Remoto – SBSR, João Pessoa, 2015. <Disponível em http://www.dsr.inpe.br/sbsr2015/files/p0031.pdf. Acesso em 20 de abr. 2017>.

PRODEAF. **ProDeaf® App,** 2016 <Disponível em http://www.prodeaf.net/pt-br/OQueE. Acesso em 20 de fev. 2017>.

RYBENÁ. **Rybená® App**, 2009 <Disponível em http://portal.rybena.com.br/site-rybena/conheca-o-rybena. Acesso em 04 de abr. 2017>.

ROSA, Andréa da Silva, **Entre a visibilidade da linguagem de sinais e a invisibilidade da tarefa do intérprete**. 206 f. (Dissertação de Mestrado em Educação) - Faculdade de Educação, Universidade Estadual de Campinas, Campinas, 2005.

SANTOS, Breno Santana et al. **Análise Comparativa de Algoritmos de Mineração de Texto Aplicados a Históricos de Contas Públicas** – XI *Brasilian Symposium on Information System* – SBSI, Goiânia, 2015. <Disponível em http://www.lbd.dcc.ufmg.br/colecoes/sbsi/2015/089.pdf. Acesso em 20 de mar. 2017>.

STROBEL, Karin Lilian; FERNANDES, Sueli. **Aspectos Linguísticos da LIBRAS – Língua Brasileira de Sinais** - Secretaria de Estado e da Educação. Superintendência de Educação. Departamento de Educação Especial, Curitiba, 1998. <Disponível em

http://www.LIBRASgerais.com.br/materiais-inclusivos/downloads/Aspectos-linguisticos-da-LIBRAS.pdf. Acesso em 19 de abr. 2017>.

TAVARES, João E. da R.; LEITHARDT Valderi. **SensorLIBRAS: Tradução Automática LIBRAS-Português Através da Computação Ubíqua**, 2007. <Disponível em https://www.researchgate.net/profile/Valderi_Leithardt/publication/267242691_SensorLIBRAS_Traducao_Automatica_LIBRASPortugues_Atraves_da_Computacao_Ubiqua/links/55c949ce08aeb97567477c06.pdf. Acesso em 03 de abr. 2017>.

TESCHL, Gerald. ***Mathematical methods in quantum mechanics : with applications to Schr¨odinger operators***. *Second Edition*, Rhode Island: *American Mathematical Society*, 2010. 370p.

TISSOT, Hegler C.; CAMARGO Luiz C.; Pozo, Aurora T. R.. **Treinamento de Redes Neurais Feedforwards: comparativo dos algoritmos Backpropagation e Differential Evolution**, 2002. <Disponível em http://www.ppgia.pucpr.br/~enia/anais/enia/artigos/105243_2.pdf. Acesso em 05 de mar. 2017>.

UZAI, Luiz Gustavo de Carvalho; DUARTE, Mauricio. **Sistema de Reconhecimento de Voz - Aplicabilidade** - Faculdade de Tecnologia de Garça, Garça, [ca. 2010]. <Disponível em http://www.fatecgarca.edu.br/revista/Volume1/2.pdf. Acesso em 18 de abr. 2017>.

VALENTIN, J.L. **AGRUPAMENTO E ORDENAÇÃO**. Instituto de Biologia – Universidade Federal do Rio de Janeiro, Rio de Janeiro, 1995.

VLIBRAS. **VLIBRAS® App**, 2016. <Disponível em http://www.vLIBRAS.gov.br/. Acesso em 20 de fev. 2017>.

VOLPI, Alexandre. **FERRAMENTAS DE PYTHON PARA APRENDIZADO DE MÁQUINA**, 2016. <Disponível em https://alexandrevolpi.wordpress.com/tag/dino/ acessado em 28 de out. 2016>.

WRIKE. **Real-time Work Management Software**. 2017. <Disponível em https://br.atlassian.com/software/bitbucket. Acesso em 21 de abr. 2017>.

ANEXOS

Anexo I – Termo de Consentimento Livre e Esclarecido

Anexo I

TERMO DE CONSENTIMENTO LIVRE E ESCLARECIDO

Título da Pesquisa: **SINTRALIB - Sistema Integrado Tradutor de LIBRAS: Comunicação para todos**
Orientadora: **Mai-Ly Vanessa Almeida Saucedo Faro**
Pesquisador: **Guilherme Bruno Viturino Alves**

1. **Natureza da pesquisa**: o senhor (a) está sendo convidado (a) a participar desta pesquisa, a qual tem como finalidade avaliar o comportamento da ferramenta SINTRALIB na classificação de sinais de LIBRAS.
2. **Participantes da pesquisa**: pessoas que já frequentaram turmas de LIBRAS, ou possuem conhecimento em nível de expressar sinais nessa língua.
3. **Envolvimento na pesquisa**: ao participar deste estudo o senhor (a) permitirá que o pesquisador, **Guilherme Bruno Viturino Alves**, colete e divulgue de forma anônima a sua faixa etária, gênero (sexo) e resultado referente à Análise Qualitativa sobre o ambiente de testes da ferramenta SINTRALIB. O senhor (a) tem liberdade de se recusar a participar e ainda se recusar a continuar participando em qualquer fase da pesquisa, sem qualquer prejuízo para o senhor (a). Sempre que quiser, poderá pedir mais informações sobre a pesquisa através do telefone do pesquisador do projeto.
4. **Riscos e desconforto**: a participação nesta pesquisa não traz complicações legais. Os procedimentos adotados nesta pesquisa obedecem aos Critérios da Ética em Pesquisa com Seres Humanos, conforme Resolução no. 196/96 do Conselho Nacional de Saúde. Nenhum dos procedimentos usados oferece riscos à sua dignidade.
5. **Confidencialidade**: todas as informações coletadas neste estudo são estritamente confidenciais. Somente o pesquisador e a orientadora terão conhecimento dos dados.
6. **Benefícios**: ao participar desta pesquisa a senhor (a) não terá nenhum benefício direto. Entretanto, esperamos que este estudo forneça informações importantes sobre a tradução

de sinais de LIBRAS para o português, de forma que o conhecimento que será construído a partir desta pesquisa possa ser explorado e continuado por outros pesquisadores, onde o pesquisador se compromete a divulgar os resultados obtidos.

7. **Pagamento**: a senhor (a) não terá nenhum tipo de despesa para participar desta pesquisa, bem como nada será pago por sua participação.

Após estes esclarecimentos, solicitamos o seu consentimento de forma livre para participar desta pesquisa. Portanto preencha, por favor, os itens abaixo.

Obs: Não assine este termo, se ainda tiver dúvida a respeito.

Consentimento Livre e Esclarecido

Tendo em vista os itens acima apresentados, eu, de forma livre e esclarecida, manifesto meu consentimento em participar da pesquisa. Declaro que recebi cópia deste termo de consentimento e autorizo a realização da pesquisa e divulgação dos dados obtidos neste estudo.

_____

Assinatura do Participante da Pesquisa

_____

Assinatura do Pesquisador

_____

Assinatura do Orientador

Pesquisador: GUILHERME BRUNO VITURINO ALVES - (79) 9 9136-2152

Anexo II – Formulário para teste de reconhecimento de sinais **em LIBRAS**

Anexo II

Formulário para Teste de Reconhecimento de Sinais em LIBRAS

Título da Pesquisa: **SINTRALIB - Sistema Integrado Tradutor de LIBRAS: Comunicação para todos**
Orientadora: **Mai-Ly Vanessa Almeida Saucedo Faro**
Pesquisador: **Guilherme Bruno Viturino Alves**

1 - Esta análise tem como objetivo coletar informações referentes ao ambiente de teste da ferramenta SINTRALIB.

Formulário I - Dados Pessoais Sexo: Masculino ( ) Feminino ( ) Faixa etária: ( ) 18 - 25 anos ( ) 26 - 30 anos ( ) Maior de 30 anos

Formulário II - Teste com a rede treinada para reconhecer 10 sinais Selecione 10 letras e numere-as de 1 a 10 para ordem de execução: ( ) A ( ) B ( ) C ( ) D ( ) E ( ) F ( ) G ( ) I ( ) L ( ) M

Formulário III - Teste com a rede treinada para reconhecer 20 sinais

Selecione 10 letras e numere-as de 1 a 10 para ordem de execução:

( ) A ( ) B ( ) C ( ) D ( ) E ( ) F ( ) G ( ) I ( ) L ( ) M ( ) N ( ) O ( ) P
( ) Q ( ) R ( ) S ( ) T ( ) U ( ) V ( ) W

Resultados obtidos referentes ao Formulário II:

Letras classificadas:

1 - ( ) 2 - ( ) 3 - ( ) 4 - ( ) 5 - ( ) 6 - ( )
7 - ( ) 8 - ( ) 9 - ( ) 10 - ( )

Início do teste às ___ : ___ Término do teste às ___ : ___

Resultados obtidos referentes ao Formulário III:

Letras classificadas:

1 - ( ) 2 - ( ) 3 - ( ) 4 - ( ) 5 - ( ) 6 - ( )
7 - ( ) 8 - ( ) 9 - ( ) 10 - ( )

Início do teste às ___ : ___ Término do teste às ___ : ___