DEPARTAMENTO DE INFORMÁTICA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

André Henrique Brandt

LiRANN – Sistema de Reconhecimento de LIBRAS baseado em Redes Neurais Artificiais com Kinect

Santa Cruz do Sul
2015

André Henrique Brandt

LiRANN – Sistema de Reconhecimento de LIBRAS baseado em Redes Neurais Artificiais com Kinect

Trabalho de Conclusão II apresentado ao Curso de Ciência da Computação da Universidade de Santa Cruz do Sul para obtenção do título de Bacharel em Ciência da Computação.

Orientadora: Prof.ª. Dra. Rejane Frozza

Santa Cruz do Sul
2015

AGRADECIMENTOS

Em primeiro lugar, gostaria de agradecer aos meus pais, Manfredo e Carmen Brandt, meus irmãos, amigos, colegas, professores e todos aqueles que de alguma forma contribuíram para o desenvolvimento deste trabalho.

A minha noiva Cristiane, por estar sempre ao meu lado, me apoiando e motivando na busca de meus objetivos, e por ser uma pessoa tão carinhosa e companheira, o que me dá forças para seguir em frente.

Agradeço também a minha orientadora Prof.ª. Dra. Rejane Frozza por todo apoio no desenvolvimento deste trabalho, estando sempre disposta a solucionar minhas dúvidas e me motivando a seguir em frente.

RESUMO

A Língua Brasileira de Sinais, LIBRAS, é um idioma de modalidade gesto-visual, utilizada pela comunidade surda brasileira, que combina movimentos gestuais e expressões faciais com a finalidade de transmitir uma mensagem. A aplicação da linguagem de sinais em conjunto com a tecnologia tenta aproximar as pessoas, promovendo inclusão social, facilitando a comunicação. Esse trabalho possui como objetivo principal desenvolver uma aplicação que interprete e traduza o alfabeto da língua brasileira de sinais em tempo real, utilizando o Kinect como forma de entrada, e utiliza as Redes Neurais Artificiais como técnica para reconhecimento e processamento das informações. O sistema desenvolvido permite escrever palavras a partir da identificação das posições do alfabeto de Libras, possibilitando seu uso para o ensino da língua. Como resultado, tem-se a aplicação desenvolvida que permite o reconhecimento das posições estáticas do alfabeto de LIBRAS, com a escrita de palavras em tempo real.

Palavras chave: Kinect, LIBRAS, Redes Neurais Artificiais, Sistema de Reconhecimento do alfabeto de LIBRAS

ABSTRACT

Brazilian Sign Language, LIBRAS, is a language of gesture-visual modality used by the Brazilian deaf community which combines gestural movements and facial expressions in order to express a message. The use of sign language combined with technology attempts to bring people together, promoting social inclusion, making easier the communication. This work has as main objective to develop an application that interprets and translate the alphabet of Brazilian Sign Language in real time using the Kinect as input, and Artificial Neural Networks as a technique for recognition and processing of information. The developed system allows writing words from the identification of Libras alphabet positions, allowing its use for language teaching. As a result, there has developed the application to the recognition of static positions of LIBRAS alphabet, with the writing of words in real time.

Keywords: Kinect, LIBRAS, Artificial Neural Network, Recognition System for LIBRAS alphabet

LISTA DE FIGURAS

Figura 1 - Configurações estáticas que representam letras do alfabeto em Libras 13
Figura 2 - Exemplo utilizado para demonstração da execução do sinal “maçã” 13
Figura 3 - Espaço de sinalização utilizado por um intérprete de língua de sinais 14
Figura 4 - Tecnologias utilizadas pelo Microsoft Kinect 15
Figura 5 - Representação do sensor de profundidade do Kinect 16
Figura 6 - Imagem gerada pela captura do sensor de profundidade 17
Figura 7 - Campo de visão horizontal do Kinect para a configuração padrão 18
Figura 8 - Campo de visão vertical do Kinect para a configuração padrão 19
Figura 9 - Joints do esqueleto humano reconhecido pelo Kinect SDK 20
Figura 10 - Joints do esqueleto humano reconhecido no modo sentado 21
Figura 11 - Pixels resultantes para os gestos de mão aberta e mão fechada 25
Figura 12 - Aluno utilizando o aplicativo do quadro virtual 26
Figura 13 - Interface do ProDeaf 27
Figura 14 - Rastreamento da mão utilizando o Framework Candescent NUI 31
Figura 15 - Rastreamento errôneo de um objeto diferente de uma mão 31
Figura 16 - Área de funcionamento do Candescent NUI 32
Figura 17 - Polígono gerado pelo algoritmo Convex Hull e detecção dos dedos 33
Figura 18 - Módulos da aplicação desenvolvida 35
Figura 19 - Image Recognition with Neural Networks 36
Figura 20 - Interface da aplicação desenvolvida 38
Figura 21 - Diagrama de classes LiRANN 39
Figura 22 - Representação de uma rede neural artificial backpropagation 41

LISTA DE QUADROS

Quadro 1 - Comparativo dos trabalhos relacionados estudados 28
Quadro 2 - Base utilizada para validação e taxa de convergência 42
Quadro 3 - Quadro referente a aplicação desenvolvida 43

LISTA DE ABREVIATURAS

2D Duas dimensões

3D Três dimensões

API *Application programming interface*

EP Elementos de processamento

BSB *Berkeley Software Distribution*

IBGE Instituto Brasileiro de Geografia e Estatística

FPS *Frames* por segundo

SDK *Software Development Kit*

RGB *Red Green and Blue*

RNA Rede Neural Artificial

VGA *Video Graphics Array*

FULL HD Alta definição de Vídeo

DRW *Dynamic Time Warping*

IR *Infrared*

SUMÁRIO

1. INTRODUÇÃO 10
2. FUNDAMENTAÇÃO TEÓRICA 12
2.1 Língua de Sinais 12
2.2 Kinect 14
2.3 Software Development Kits (SDKs) 19
2.4 Redes Neurais Artificiais (RNA) 22
2.5 Considerações 23
3. TRABALHOS RELACIONADOS 24
3.1 Recognizing hand gestures with Microsoft Kinect 24
3.2 Construção de ambiente para desenvolvimento de jogos educacionais baseados em interface de gestos 25
3.3 PRODEAF Aplicativo de Libras 26
3.4 Kinect SDK Dynamic Time Warping (DTW) Gesture Recognition 27
3.5 O uso do Kinect na acessibilidade de pessoas com deficiência visual 28
3.6 Quadro comparativo dos Trabalhos Relacionados Estudados 28
4. LiRANN – SISTEMA DE RECONHECIMENTO DE LIBRAS BASEADO EM REDES NEURAIS ARTIFICIAIS COM KINECT.....30
4.1 Candescent NUI 30
4.2 Aplicação desenvolvida 34
4.3 Algoritmo Backpropagation 39
5. CONCLUSÃO 42
6. REFERÊNCIAS 45

1. INTRODUÇÃO

Pessoas consideradas diferentes, de outras culturas, línguas e necessidades possuem um histórico de exclusão social. Isto justifica o surgimento de políticas de inclusão e legislações que contemplem as necessidades destes grupos. Compreender e admitir que a exclusão existe em nossa sociedade é tão desafiador quanto incluir. Permitir que estas minorias possam compartilhar de todos os aspectos sociais significa mudar, adaptar, flexibilizar e tornar possível o acesso para elas nas mais variadas ações e situações. Ignorar as necessidades e a compreensão frente a esses grupos leva à segregação (DOMINGOS, 2010).

A comunicação é um fator fundamental e indispensável para o desenvolvimento do ser humano. Por meio da linguagem é possível expor ideias, organizar pensamentos, expressar sentimentos e conviver em sociedade.

O principal sentido que possibilita a comunicação é a audição. Porém, algumas pessoas nascem com deficiências auditivas, sejam elas parciais ou até mesmo totais. Devido a isso, a fala também é prejudicada, não sendo raros os casos em que ela não é desenvolvida. Ainda há casos em que o indivíduo perde a audição devido à exposição constante ao barulho excessivo ou por complicações de saúde. Essas pessoas geralmente se comunicam através de gestos, utilizando uma linguagem própria feita através de sinais (BARROS, 2010).

As Línguas de Sinais são línguas complexas com estruturas gramaticais próprias, compostas pelos níveis linguísticos: fonológico, morfológico, sintático e semântico. O que diferencia as línguas de sinais das demais línguas é a sua modalidade visual-espacial. Porém como qualquer outra língua, podem ser usadas para expressar qualquer pensamento ou ideia, por mais abstratos que sejam (RODRIGUES, 2007).

Mesmo que não exista um sinal para expressar uma determinada palavra, é possível transmiti-la através da datilologia, ou alfabeto manual, que consiste num sistema de representação das letras do alfabeto das línguas orais escritas, por meio das mãos. É comumente utilizado para soletrar palavras e para sinalizar palavras que não possuem sinal próprio em Libras, como por exemplo, palavras estrangeiras e nomes pessoais.

O reconhecimento de gestos é um processo no qual os gestos-manuais são automaticamente reconhecidos em tempo real por um software de computador, através de uma câmera. O estudo nesta área ganhou bastante atenção nos últimos anos devido à popularidade e as possibilidades criadas pelo lançamento do Microsoft Kinect, sensor de movimentos desenvolvido para o Xbox 360 e Xbox One, o qual possui um conjunto de

tecnologias que permite aos seres humanos interagirem naturalmente com ambientes virtuais sem a necessidade de ter em mãos um controle/*joystick* (MICROSOFT, 2015).

Neste trabalho, propõe-se utilizar o Kinect para fazer a captura do alfabeto manual de Libras, transcrevendo-o em tempo real por meio de reconhecimento com a técnica de Redes Neurais Artificiais.

O trabalho é embasado em duas principais justificativas, a primeira num contexto científico, onde a pesquisa traz esclarecimentos de onde se encontra o nível de estudo na área, bem como identifica o foco dos trabalhos relacionados e tendências futuras. Já a segunda, na questão social, onde o trabalho tem como principal objetivo promover a inclusão de usuários de Libras no contexto social e educacional, aplicando a tecnologia ao aprendizado da língua.

Esse trabalho possui como objetivo principal desenvolver uma aplicação que interprete e traduza o alfabeto da língua brasileira de sinais em tempo real, utilizando o Kinect como forma de entrada, e a técnica de Redes Neurais Artificiais para fazer o reconhecimento e processamento das informações.

Como objetivos específicos, pode-se citar:

- Estudar as bibliotecas disponíveis pelo Kinect.
- Estudar as Redes Neurais Artificiais para reconhecimento de padrões.
- Pesquisar trabalhos relacionados ao tema da pesquisa.
- Desenvolver um sistema capaz de captar as informações e traduzi-las em tempo real.
- Analisar os resultados obtidos, dificuldades encontradas e possibilidades futuras.

O trabalho está organizado na seguinte estrutura: O capítulo 2 apresenta o referencial teórico relativo à língua de sinais, descrição do Kinect e redes neurais artificiais. No capítulo 3 são apresentados os trabalhos relacionados; o capítulo 4 descreve as funcionalidade do sistema desenvolvido; no capítulo 5 é apresentada a conclusão.

2. FUNDAMENTAÇÃO TEÓRICA

Este capítulo apresenta os conceitos relacionados ao desenvolvimento deste trabalho, como língua de sinais, dispositivo Kinect e redes neurais artificiais.

2.1 Língua de Sinais

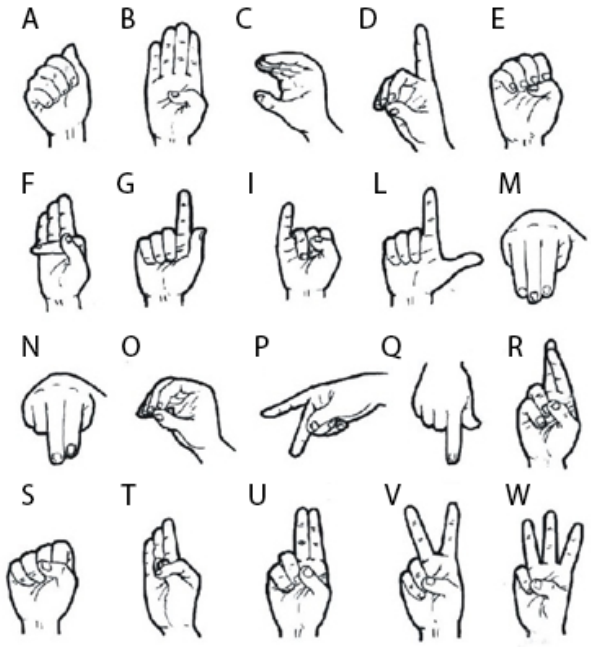
Línguas de sinais não são meras transcrições de línguas faladas, elas são compostas de regras léxicas, sintáticas e semânticas próprias e utilizam de recursos não disponíveis em línguas faladas, como por exemplo, espacialidade. Assim como as línguas faladas, apresentam constante evolução, adequando-se às necessidades de seus utilizadores.

Diferente do que muitos acreditam, as línguas de sinais não são universais, até mesmo dentro de um país é comum a utilização de diferentes dialetos em uma mesma língua. Sendo assim, um brasileiro que se comunica utilizando Libras não é capaz de se comunicar com uma pessoa que utiliza ASL (*American Sign Language*) (PISTORI; NETO, 2003).

A Língua Brasileira de Sinais (LIBRAS) é a forma de comunicação oficial utilizada pela comunidade brasileira portadora de deficiências auditivas. Segundo dados do IBGE (www.ibge.com.br), existem no Brasil cerca de seis milhões de pessoas portadoras de deficiências auditivas, sendo que destas, cerca de 170 mil são completamente surdas. Para a maior parte destas pessoas, a Libras é principal língua natural utilizada para sua comunicação.

Assim como em todas as línguas de sinais, a Libras é formada de um conjunto de formas básicas para as mãos, chamadas de configurações. É estimado que exista em Libras um total de 46 configurações. Para a representação do alfabeto, por exemplo, são utilizadas 20 configurações estáticas e 6 configurações dinâmicas, ou seja, exigem movimentação das mãos para representação (correspondentes às letras h, j, k, x, y, e z). A Figura 1 mostra as representações das letras estáticas feitas com as mãos.

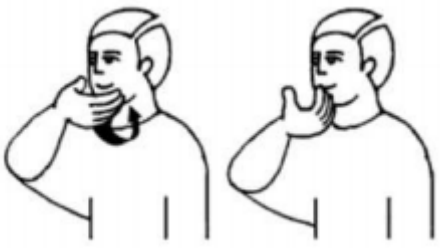
Figura 1 - Configurações estáticas que representam letras do alfabeto em Libras



Fonte: (CBSURDOS.ORG.BR, 2008).

A comunicação em línguas de sinais não envolve somente os movimentos das mãos, ela é feita utilizando o corpo como um todo, onde além de configurações das mãos são utilizados posicionamentos e movimentos dos braços, do tronco, da cabeça e expressões faciais. A Figura 2 demonstra um exemplo do um sinal da palavra “maçã” representada em Libras.

Figura 2 - Exemplo utilizado para demonstração da execução do sinal “maçã”

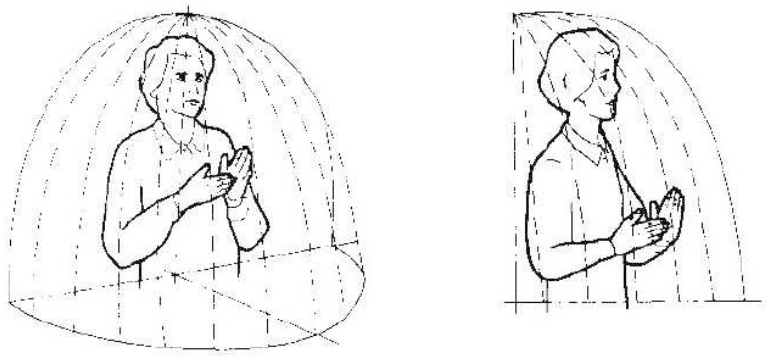


Fonte: (CAPOVILLA; RAPHAEL, 2001).

A forma com que o corpo se movimenta pode ainda determinar a intensidade do que está sendo transmitido. Desta forma, torna-se extremamente complexo o reconhecimento computacional deste tipo de língua, frente à grande variedade de sinais existentes e a forma com que estes podem ser realizados (FIALHO, 2004).

A Figura 3 apresenta uma representação gráfica do espaço de sinalização por pessoas que se utilizam de línguas de sinais. É possível verificar que este espaço de sinalização, requer que as ferramentas computacionais sejam capazes de determinar não somente a localização em duas dimensões das mãos, mas sim a localização espacial em três dimensões das mãos e de todo o corpo do intérprete (MORRISSEY, 2008).

Figura 3 - Espaço de sinalização utilizado por um intérprete de língua de sinais



Fonte: (MORRISSEY, 2008).

Na sequência é apresentado o dispositivo Kinect, seus recursos de hardware, e os recursos do software que possibilitam fazer o rastreamento do esqueleto humano.

2.2 Kinect

Lançado em junho de 2009, durante a conferência *Electronic Entertainment Expo* e previamente chamado de “Projeto Natal”, a Microsoft anunciou o acessório para o console de vídeo game Xbox 360, oferecendo recursos de interação inovadores, sem a necessidade de utilizar um *joystick*, graças a câmeras e sensores de movimento contidos no dispositivo. Na conferência do ano seguinte, foi revelado seu nome comercial, Kinect. O lançamento ocorreu em novembro de 2010, vendendo em suas primeiras semanas o equivalente a oito milhões de

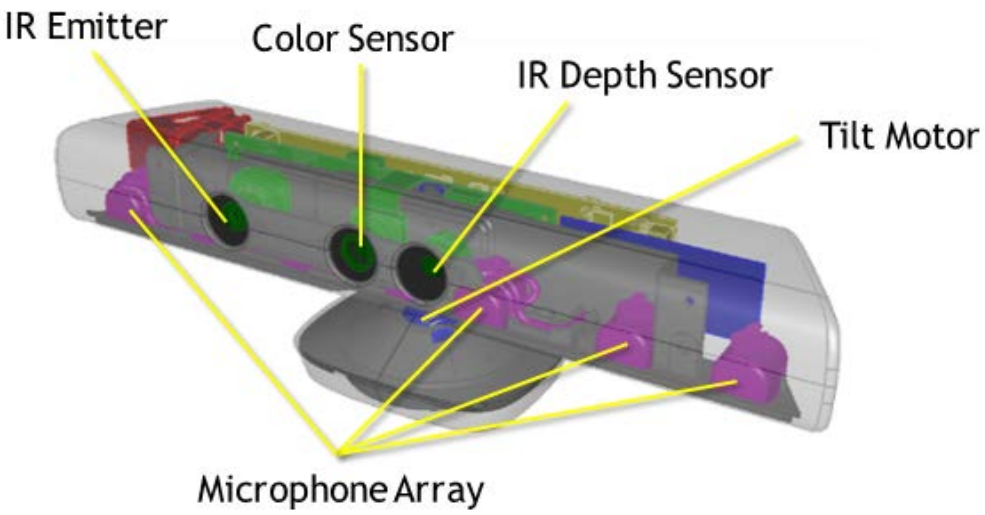
unidades, inclusive entrando para o *Guinness World Records*, como o dispositivo eletrônico de venda mais rápida (MICROSOFT, 2011).

Com o lançamento do kit de desenvolvimento para o Kinect é possível criar aplicações que vão muito além de jogos, como novos programas onde é possível desenvolver interfaces naturais de comunicação com o computador e dispositivos com sistemas embarcados.

2.2.1 Tecnologias do Kinect

O Kinect é um dispositivo composto de múltiplas tecnologias, como câmeras de vídeo, sensor de profundidade e microfones multi-matriz. A Figura 4 mostra uma imagem do dispositivo, identificando os recursos disponíveis.

Figura 4 - Tecnologias utilizadas pelo Microsoft Kinect



Fonte: (MSDN MICROSOFT, 2015).

(a) Câmera de Vídeo

A câmera de vídeo detecta as cores vermelho, verde e azul, utilizando o padrão RGB e possui resolução de 640 x 480 *pixels* a 30 FPS (frames por segundo), ou 1280 x 960 *pixels* a um máximo de 15 FPS. É responsável pela captura da imagem reproduzida, ajudando na detecção de movimentos e reconhecimento facial (CRAWFORD, 2010).

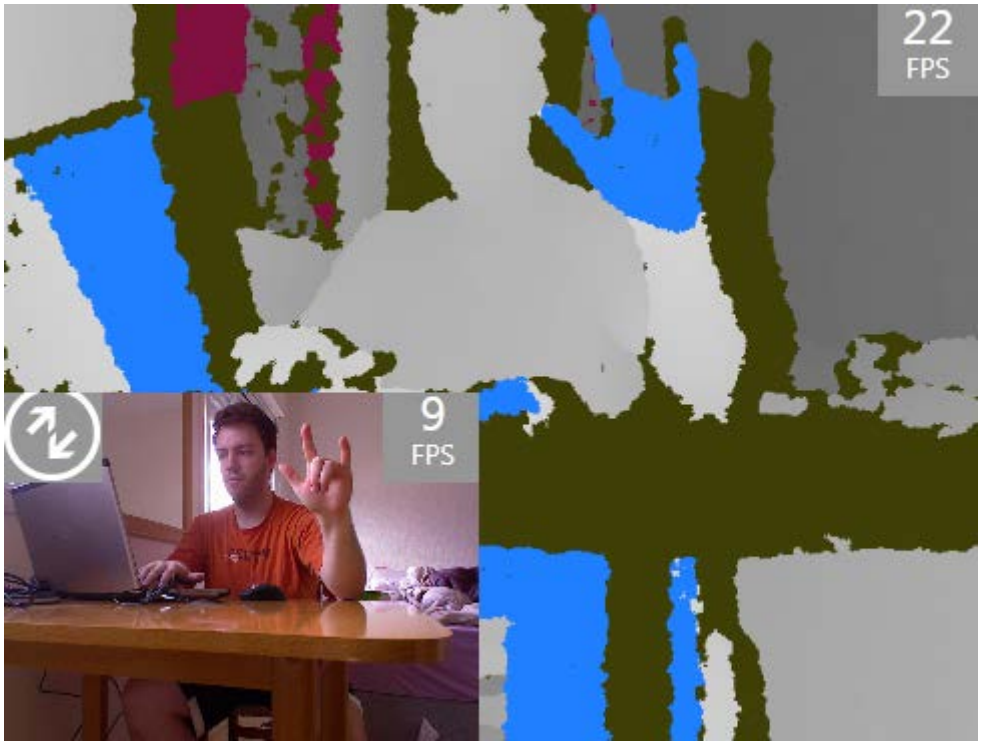
(b) Sensor de Profundidade

O sensor de profundidade é composto por um emissor e um receptor de luz infravermelha. O emissor dispara um feixe com diversos pontos de luz no ambiente a sua frente. Estes pontos são refletidos de volta para o sensor, onde o receptor analisa a distância entre os pontos e gera uma representação de profundidade. Na imagem gerada, a área mais distante é exibida com tons mais escuros e a área mais próxima com tons mais claros, sendo nesta última onde existe maior concentração de pontos infravermelhos (KHOSHELHAM, 2011). A resolução da imagem gerada pelo sensor de profundidade é de 640 x 480 *pixels*, utilizando o padrão VGA para cores, a uma taxa de 30 frames por segundo. A figura 5 representa os pontos de luz infravermelha emitidas pelo sensor de profundidade e a figura 6 representa a imagem gerada.

Figura 5 - Representação do sensor de profundidade do Kinect

Fonte: (FISHER, 2014).

Figura 6 - Imagem gerada pela captura do sensor de profundidade



Fonte: (AUTORES, 2015).

(c) Microfone Multi-matriz

O Kinect possui quatro microfones capazes de reconhecer a localização da origem do som e a direção das ondas sonoras. Também é capaz de distinguir a voz de diferentes pessoas, isolando os barulhos internos do ambiente e removendo o som emitido pela TV (CRAWFORD, 2010).

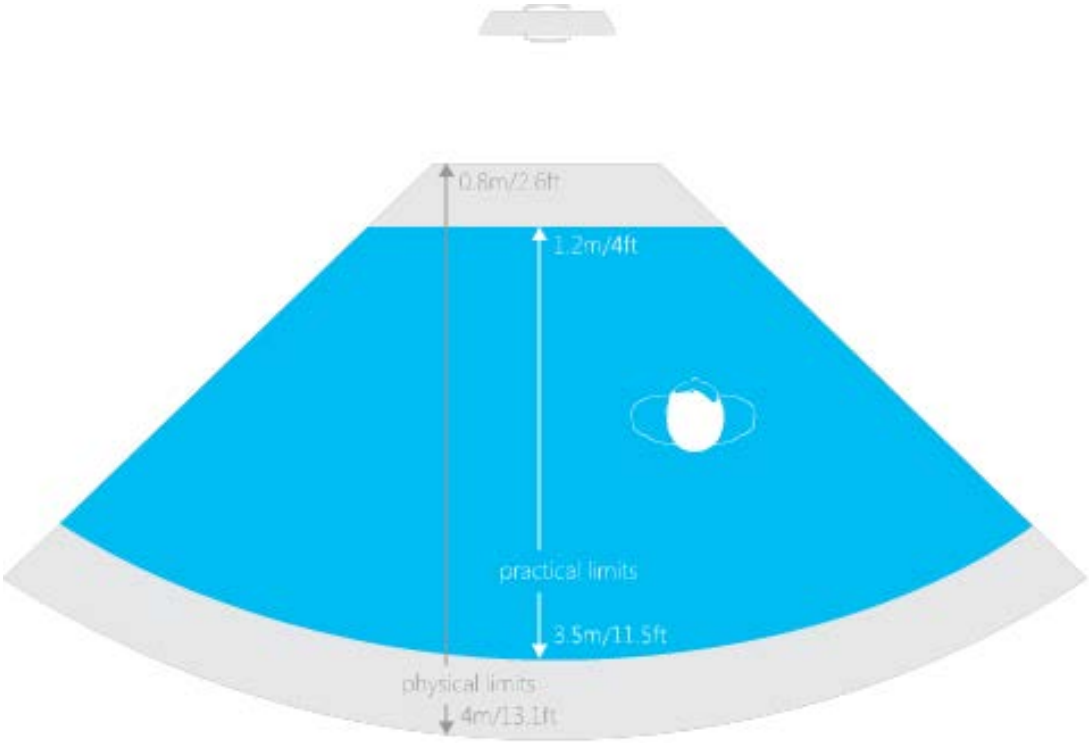
(d) Motor de Inclinação

O sensor Kinect possui um motor que possibilita ao dispositivo inclinar-se automaticamente conforme a necessidade. O sensor inclina-se para baixo para encontrar o chão e para cima para ver os jogadores no campo de visão. O ângulo pode ser ajustado num intervalo de +27 graus e -27 graus, aumentando assim a área de interação em frente ao sensor (MSDN MICROSOFT, 2015).

(e) Campo de visão

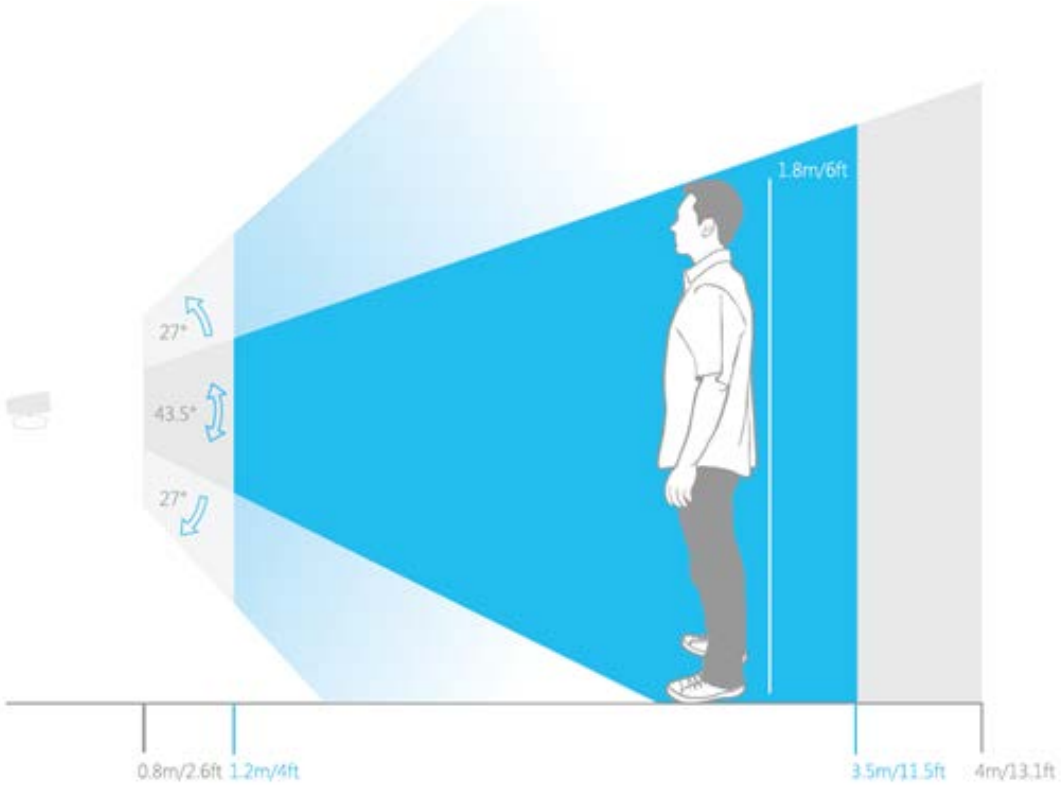
O campo de visão do Kinect em relação aos usuários é determinado pelas configurações da câmera infravermelha. No modo padrão, o Kinect pode identificar pessoas que estão entre 0,8 e 4,0 metros de distância do dispositivo, porém o valor prático sugerido é entre 1,2 a 3,5 metros. Utilizando o "*Near Rage Mode*" o Kinect consegue identificar a uma distância de 0,4 até 3 metros, sendo o valor prático ideal entre 0,8 e 2,5 metros. As figuras 7 e 8 demonstram a área de funcionamento do dispositivo.

Figura 7 - Campo de visão horizontal do Kinect para a configuração padrão



Fonte: (MSDN MICROSOFT, 2015).

Figura 8 - Campo de visão vertical do Kinect para a configuração padrão



Fonte: (MSDN MICROSOFT, 2015).

2.3 *Software Development Kits* (SDKs)

Atualmente, existem dois principais *frameworks* utilizados para o desenvolvimento de aplicações com o Kinect, o OpenNI e o Microsoft Kinect SDK. Nesta seção, é apresentada uma visão geral destas ferramentas que fazem a interação entre o *hardware* e os usuários.

2.3.1 OpenNI

A primeira SDK a ser lançada foi a OpenNI (*Open Natural Interaction*) da empresa israelense PrimeSense, responsável pela tecnologia utilizada no sensor 3D do Microsoft Kinect. Essa SDK foi lançada no final de 2010 e é utilizada largamente como forma de fazer o Microsoft Kinect funcionar em outra máquina (computador) diferente do Xbox 360. De modo geral, o *framework* fornece diversas funcionalidades, tais como rastreamento do esqueleto, reconhecimento de gestos, gravação e reprodução dos dados do sensor. Atualmente possui algumas vantagens em relação à SDK oficial, como um módulo de reconhecimento de

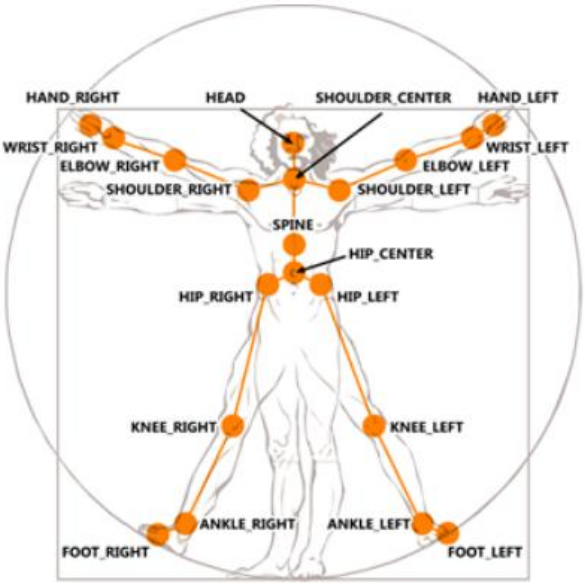
gestos que permitem, clicar, arrastar, levantar a mão, etc. Também possui o recurso de gravação e reprodução do sensores, facilitando no desenvolvimento de aplicações e o fato de ser multissistemas, funcionando em Linux, Windows (XP, Vista e 7) e Mac OS X. Possui, entretanto, algumas desvantagens, como fazer uso de uma série de drivers que precisam ser instalados individualmente e não possui suporte ao reconhecimento de áudio do Kinect. A SDK requer ainda uma pose de calibragem para o reconhecimento do usuário no início das aplicações, sendo que este processo leva em média três segundos e ainda é suscetível a falhas.

2.3.2 Kinect SDK

O *kit* de desenvolvimento de *software* oficial da Microsoft, Kinect SDK, foi lançado em junho de 2011, composto de vários exemplos de códigos, cada um demonstrando uma função específica da SDK, como reconhecimento de áudio, amostragem das diferentes câmeras, detecção do esqueleto e colisão do esqueleto com objetos. A SDK possui classes prontas para essas funções, tornando simples a utilização de cada uma das funções.

O esqueleto detectado pela SDK possui uma coleção de pontos do corpo humano (articulações) que são chamadas de *joints*. Cada ponto mapeado possui três coordenadas (x, y e z) expressas em centímetros, onde x e y representam a posição dos pontos e z representa a distância entre o ponto do corpo e o Kinect.

Figura 9 - *Joints* do esqueleto humano reconhecido pelo Kinect SDK



Fonte: (MSDN MICROSOFT, 2015).

A estrutura possui um total de 20 *joints*, 5 a mais que a OpenNI, sendo elas: cabeça, ombro central (pescoço), ombro direito e esquerdo, cotovelos direito e esquerdo, pulsos direito e esquerdo, mãos direita e esquerda, espinha, lados direito, esquerdo e centro do quadril, joelhos direito e esquerdo, calcanhares direito e esquerdo e pés direito e esquerdo. A figura 10 demonstra o reconhecimento do esqueleto para a opção "*Seated Mode*", modo sentado (JANA, 2012).

Figura 10 - *Joints* do esqueleto humano reconhecido no modo sentado

Fonte: (AUTORES, 2015).

Conforme pode ser observado na figura 10, nesta configuração o Kinect reconhece somente a parte superior do esqueleto, possibilitando o uso para distâncias mais curtas, ou em situações em que não sejam necessárias informações das outras *joints*. Na sequência é apresentada a tecnologia de inteligência artificial utilizada para o reconhecimento de padrões.

2.4 Redes Neurais Artificiais (RNA)

As redes neurais artificiais são inspiradas no modelo biológico da inteligência, simulando a maneira como o cérebro é organizado em sua arquitetura e como é capaz de executar tarefas computacionais (TAFNER, 1998). As redes neurais artificiais emulam o

comportamento do sistema neural humano, assim, o modelo de neurônio artificial assemelha-se ao neurônio biológico, apresentando funções e arquitetura cognitiva parecidas, apesar de serem constituídos de materiais diferentes, um virtual e outro orgânico (FREEMAN, 1992). Como objetivo, visam solucionar problemas computacionais baseados nas seguintes características:

- Capacidade de “aprender” através de exemplos e de reconhecer padrões similares que nunca haviam sido apresentadas como exemplo.
- Bom desempenho em tarefas onde o conhecimento sobre como encontrar uma solução não é explícito.
- Não requer conhecimento de modelos matemáticos dos domínios da aplicação.
- Elevada imunidade a ruído. Seu funcionamento não entra em colapso na presença de informações falsas ou ausentes.
- Possibilidade de simulação do raciocínio.

Uma rede neural artificial é composta por um grande número de elementos de processamento (neurônios), amplamente conectados entre si. Possui, no mínimo, uma camada de entrada, responsável por armazenar a informação de entrada para ser passada à camada seguinte, e uma camada de saída, onde o resultado final é concluído e apresentado. Entre estas duas camadas, pode possuir diversas camadas intermediárias, onde é feita a maior parte do processamento.

Os neurônios são responsáveis por todo o processamento realizado na RNA, sendo realizado de forma distribuída entre os elementos processadores (EP) da rede. Cada EP coleta a informação enviada a ele e produz um único valor de saída, sendo sua saída uma função dos pesos e das entradas. Este valor de saída é propagado através das conexões, que fluem em único sentido. Cada conexão possui um peso, o qual é responsável pela memorização do padrão.

Os padrões são os dados de entrada da rede correspondem à modelagem de um problema aplicado a uma RNA. A um determinado padrão de entrada corresponde um sinal de saída. Uma das principais aplicações de RNAs é o reconhecimento de padrões. Para tanto, é necessário a utilização de funções, as quais são modelos matemáticos, utilizados no treinamento e reconhecimento de padrões. As funções de aprendizado servem para realizar o ajuste dos pesos da rede, possibilitando o aprendizado de um determinado padrão. Em geral, dependem do modelo de rede neural artificial escolhido (FROZZA, 2014).

Em uma RNA, o conhecimento é implicitamente armazenado nas conexões entre os neurônio. O aprendizado, por outro lado, é a habilidade de modificar os pesos destas conexões, podendo ser classificados em dois tipos:

- Aprendizado supervisionado: A rede neural artificial é treinada com um conjunto de entradas e um conjunto de saídas desejadas para cada entrada. Sempre que for apresentada uma entrada à rede, o usuário deve verificar o resultado obtido. Caso o resultado seja diferente do esperado, deve-se fazer os ajustes nos pesos. Exemplo: Modelo Perceptron.
- Aprendizado não-supervisionado: São utilizados apenas os valores de entrada para o treinamento da rede. A rede classifica as entradas usando seus critérios, envolvendo processo de competição e o processo de cooperação entre os neurônios da rede. Exemplo: Modelo de Kohonen.

2.5 Considerações

De forma geral, ambas SDKs apresentadas fornecem recursos semelhantes. Apesar do *framework* OpenNI fornecer reconhecimento de gestos e a possibilidade de gravar e reproduzir os dados do sensor, a API oficial do Kinect possui melhor documentação e acesso a todos os recursos do Kinect, além de não ser necessária a calibragem inicial do esqueleto.

Já em relação às RNAs, o modelo de aprendizado supervisionado mostra-se mais indicado para a resolução de problemas nos quais se deseja fazer o reconhecimento de padrões relativos a imagens. Por este motivo, para o desenvolvimento deste trabalho, optou-se por utilizar um modelo supervisionado.

3. TRABALHOS RELACIONADOS

Neste capítulo, serão abordados trabalhos relacionados à língua de sinais, auxiliada por computador e também a aplicação do Kinect para auxiliar deficiências físicas.

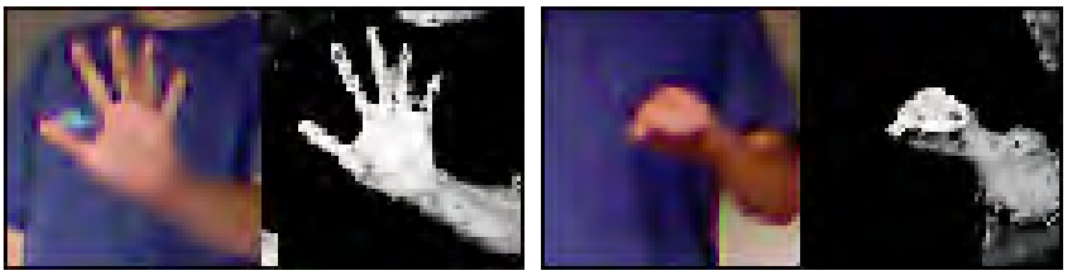
3.1 *Recognizing hand gestures with Microsoft Kinect*

Neste trabalho, o objetivo é estudar a viabilidade do reconhecimento de gestos em uma escala pequena. Ao invés de tentar reconhecer ações de corpo inteiro como acenar e pular, tenta-se identificar gestos simples que uma pessoa pode realizar com a sua mão, por exemplo, agarrar, apontar. São utilizados os recursos da câmera RGB e o sensor de profundidade do Kinect.

Foi utilizado um rastreador de corpo inteiro, que primeiramente identifica a localização da mão de uma pessoa, a qual deve estar posicionada na extremidade de um braço na estrutura esquelética. A segunda etapa consiste em reconhecer os *pixels* que constituem a área referente à mão. Nesta etapa também é feito um processamento da imagem, onde são ajustados o brilho e contraste da imagem para obter melhores resultados. Na última etapa são identificados os movimentos em uma sequência de imagens e poses pré-definidas em um conjunto de treinamento para obter-se o gesto atribuído. Como resultado foi obtida uma taxa de sucesso de 90% para reconhecimento de gestos de agarrar e soltar.

Das dificuldades apresentadas pelo trabalho, pode-se citar o problema em identificar os *pixels* referentes à área da mão, tanto por questões de tons diferentes de pele, quanto por questões de iluminação no ambiente. Outro problema citado no trabalho é a resolução do dispositivo. Para uma pessoa que está a cerca de 3 metros de distância do Kinect, a mão ocupa uma região de não mais de 64 x 64 *pixels*. Outro fator que pode influenciar no resultado é a questão do ângulo de visão. A figura 11 demonstra a imagem resultante para os gestos de mão aberta e fechada (TANG, 2011).

Figura 11 - *Pixels* resultantes para os gestos de mão aberta e mão fechada



Fonte: (TANG, 2011).

3.2 Construção de ambiente para desenvolvimento de jogos educacionais baseados em interface de gestos

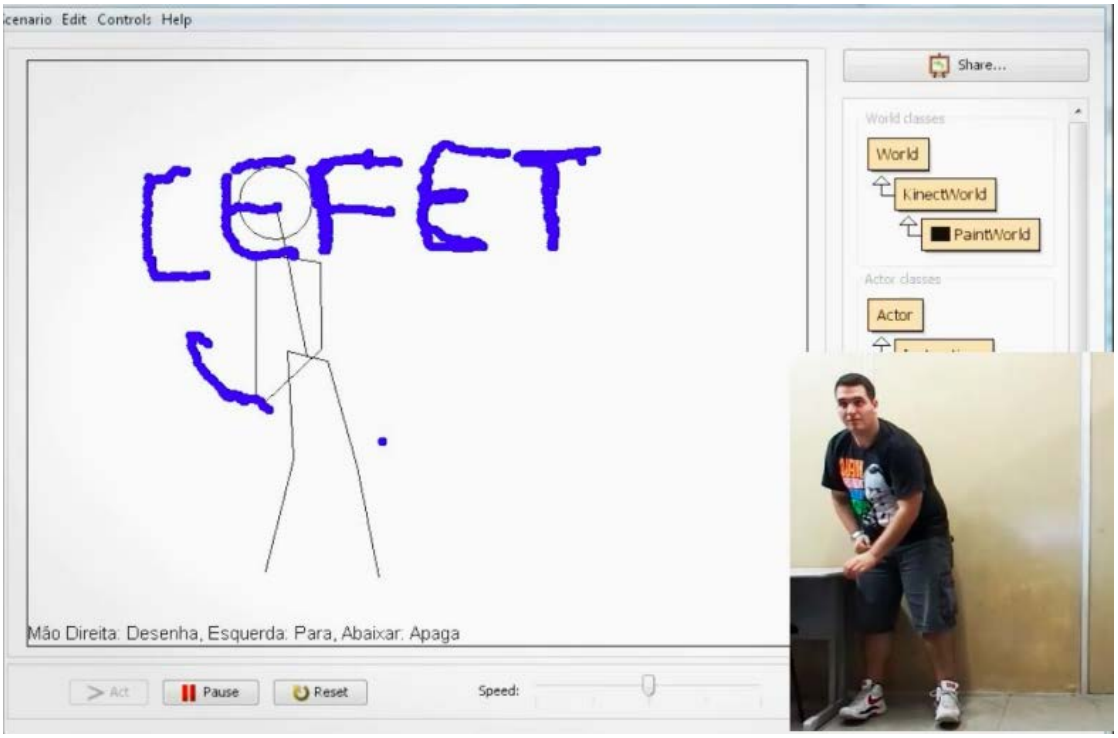
Este projeto foi desenvolvido no Centro Federal de Educação Tecnológica do Rio de Janeiro (CEFET-RJ) o qual visa o desenvolvimento de diversas aplicações de jogos educacionais baseados em interfaces de gestos.

Uma das aplicações desenvolvidas consiste em uma espécie de quadro virtual, o qual possibilita ao usuário escrever em uma tela imaginária com sua mão direita, parar de escrever levantando a mão esquerda e apagar todo o quadro, realizando o movimento de abaixar.

Nesse ambiente, a proposta era que um aluno, ao escrever, pudesse ter seus textos verificados e corrigidos por um professor. Foi identificado, também, que esse protótipo poderia servir de base para um aplicativo de ajuda a pessoas com deficiências motoras que desejassem comunicar-se com gestos e para o desenvolvimento de jogos educacionais para o público infantil.

Um dos desafios apontados pelo projeto era a questão de, ao mesmo tempo, aprender os conceitos de programação e produzir jogos interessantes em um ambiente que não fosse complexo demais para não desestimular o aprendizado. Nesse contexto de facilitar o aprendizado de programação, as aplicações foram desenvolvidas em uma plataforma didática para a construção de aplicativos, baseada na linguagem de programação Java, chamada de *Greenfoot*, a qual possui uma interface gráfica e suporte para a API do Kinect. A figura 12 demonstra a interface da aplicação desenvolvida no *Greenfoot* utilizando a API do Kinect (CEFET-RJ, 2013).

Figura 12 - Aluno utilizando o aplicativo do quadro virtual



Fonte: (CEFET-RJ, 2013).

3.3 PRODEAF Aplicativo de Libras

O aplicativo ProDeaf é um dicionário e tradutor de libras desenvolvido pela ProDeaf Tecnologias Assistivas em parceria com a Bradesco Seguros. Ele possibilita traduzir diversas palavras para gestos utilizando computadores e smartphones. Pode-se digitar a palavra, procurar no dicionário dentro do aplicativo ou utilizar o reconhecimento de voz. Para utilizar o recurso de reconhecimento de voz é necessário ter conexão com a internet.

Após escolher a palavra a ser traduzida o aplicativo volta para a tela inicial e o *Avatar* realiza os gestos, indicando a palavra logo acima. Se a palavra não estiver disponível no dicionário a palavra é soletrada utilizando o alfabeto de libras. A figura 13 demonstra a interface da aplicação (PRODEAF, 2013).

Figura 13 - Interface do ProDeaf

Fonte: (JANELA TECH, 2015).

3.4 *Kinect SDK Dynamic Time Warping (DTW) Gesture Recognition*

É um projeto que permite aos desenvolvedores incluírem o reconhecimento de gestos rápidos, confiáveis e altamente personalizáveis utilizando a SDK oficial da Microsoft para o Kinect. Utiliza rastreamento do esqueleto e atualmente suporta vetores 2D.

No projeto, o Kinect é utilizado para gravar os gestos, capturando doze pontos diferentes do esqueleto do usuário. Estes pontos representam a parte superior do corpo: cabeça, centro dos ombros, ombro direito, ombro esquerdo, coluna, centro do quadril, cotovelo direito, cotovelo esquerdo, pulso direito, pulso esquerdo, mão direita e mão esquerda.

Para gravar o gesto, são coletados 32 *frames* em um espaço de tempo de aproximadamente um segundo. A partir destes gestos, são armazenadas as coordenadas X e Y de cada posição em um arquivo de texto. Como o arquivo gravado possui apenas as coordenadas X e Y, não é possível reconhecer movimentos em 3D, como por exemplo, a mão se aproximando da câmera.

KinectDTW utiliza um algoritmo de vizinho mais próximo baseado em vetor para monitorar e classificar seus gestos. O elemento *Dynamic Time Warping* significa que pode reconhecer gestos executados em velocidades diferentes, sem importar a velocidade em que foram gravados (KinectDTW, 2011).

3.5 O uso do Kinect na acessibilidade de pessoas com deficiência visual

Neste trabalho, é apresentada uma proposta de um *software* que utiliza processamento de imagem e os recursos do Kinect para localizar objetos e calcular a distância do indivíduo em relação a esses. O *software* trabalha com uma técnica de processamento de imagem denominada “Crescimento de Regiões” a fim de identificar o objeto mais próximo em relação ao Kinect, contribuindo, assim, para a autonomia na locomoção de deficientes visuais, identificando obstáculos e informando sua distância para o usuário (GASSEN, 2013).

3.6 Quadro comparativo dos Trabalhos Relacionados Estudados

A seguir é apresentado um quadro comparativo (Quadro 1) dos trabalhos relacionados estudados, demonstrando seus objetivos, dispositivos, técnicas utilizadas e os resultados atingidos.

Quadro 1 - Comparativo dos trabalhos relacionados estudados

Trabalho	Objetivo do Trabalho	Dispositivos e Técnicas Utilizadas	Questões a Destacar
Recognizing hand gestures with Microsoft Kinect (TANG, 2011)	Estudar a viabilidade do reconhecimento de gestos em uma escala pequena como, por exemplo, agarrar e apontar.	São utilizados os recursos da câmera RGB e o sensor de profundidade do Kinect.	Como resultado foi obtida uma taxa de sucesso de 90% para reconhecimento de gestos de agarrar e soltar.
Construção de ambiente para desenvolvimento de jogos educacionais baseados em interface de gestos (CEFET-RJ, 2013)	Promover o estudo do Kinect e suas aplicações em contextos educacionais.	Kinect SDK e GreenFoot (*framework* didático para desenvolvimento java)	Um dos trabalhos desenvolvidos consiste em uma espécie de quadro virtual, o qual possibilita ao usuário escrever em uma tela imaginária com sua mão.
PRODEAF Aplicativo de Libras (PRODEAF, 2013)	Traduzir diversas palavras para gestos em Libras, utilizando computadores e smartphones.	-	O aplicativo já é utilizado por mais de 130.000 pessoas de acordo com informações do site.
Kinect SDK Dynamic Time Warping (DTW) Gesture Recognition (KINECTDTW, 2011)	Reconhecer gestos rápidos, confiáveis e altamente personalizáveis, utilizando rastreamento do esqueleto em 2D.	Kinect SDK, Algoritmo de vizinho mais próximo baseado em vetor para monitorar e classificar seus gestos.	O elemento *Dynamic Time Warping* significa que pode reconhecer gestos executados em velocidades diferentes, sem importar a velocidade em que foram gravados, porém, o fato de utilizar apenas 2D limita seu uso para gestos simples.

O uso do Kinect na acessibilidade de pessoas com deficiência visual (GASSEN, 2013)	Promover autonomia de deficientes visuais por meio de um software desenvolvido com o Kinect, o qual calcula e informa a distância entre o usuário e os objetos.	Utiliza Kinect e sua SDK, juntamente com um algoritmo de processamento de imagem denominado "Crescimento de Regiões"	O *software* é capaz de detectar a distância de objetos com pequenas variações, portanto acredita-se que seja possível aplicar a tecnologia num ambiente real.

Fonte: (AUTORES, 2015).

Através do estudo dos trabalhos relacionados, pode-se observar no quadro 1 a versatilidade do uso do Kinect em diversas aplicações que visam promover a acessibilidade de pessoas portadoras de deficiências. No comparativo, dos trabalhos que utilizam o Kinect, pode-se notar que a SDK oficial do Kinect é a mais utilizada e que ainda existe muito trabalho a ser desenvolvido para que esta tecnologia possa ser utilizada em um ambiente prático.

4. LiRANN – SISTEMA DE RECONHECIMENTO DE LIBRAS BASEADO EM REDES NEURAIS ARTIFICIAIS COM KINECT

Neste capítulo, será descrita a aplicação desenvolvida, suas funcionalidades e recursos necessários para sua utilização.

O primeiro desafio para esta pesquisa foi buscar uma biblioteca ou projeto que fornecesse informações sobre o rastreamento de mãos e dedos, visto que o objetivo do projeto é desenvolver uma aplicação que traduza o alfabeto de Libras e não implementar um algoritmo para detecção e rastreamento de mãos e dedos.

Existem diversos projetos em relação à detecção e rastreamento da mão, porém a grande maioria destes é focado em identificar apenas a mão, representando-a com uma única *joint*, isto significa que nestes não existe rastreamento para os dedos, tornando-se difícil para a identificação de sinais do alfabeto de Libras.

Foram analisados diversos projetos, tais como *Kinect Paint*[1], *Kinect Toolbox*[2], *Flexible Action and Articulated Skeleton Toolkit (FAAST)*[3], porém foram descartados, uma vez que apenas fazem rastreamento do esqueleto, assim como o *Kinect SDK Dynamic Time Wrapping (DTW) Gesture Recognition*[4], que possibilita gravar gestos em 2D e reconhecê-los, porém funciona somente como as *joints* do esqueleto.

Das bibliotecas e projetos estudados durante a fase de pesquisa, o Candescent NUI, demonstrou ser a melhor opção para o desenvolvimento do trabalho proposto, uma vez que fornece rastreamento da mão, reconhecimento da ponta dos dedos, palma da mão e posição dos dedos. Mas, após o estudo efetuado, não foi possível utilizá-lo devido à alta variação dos valores, principalmente pela baixa resolução do Kinect em relação ao mapa de profundidade.

4.1 Candescent NUI

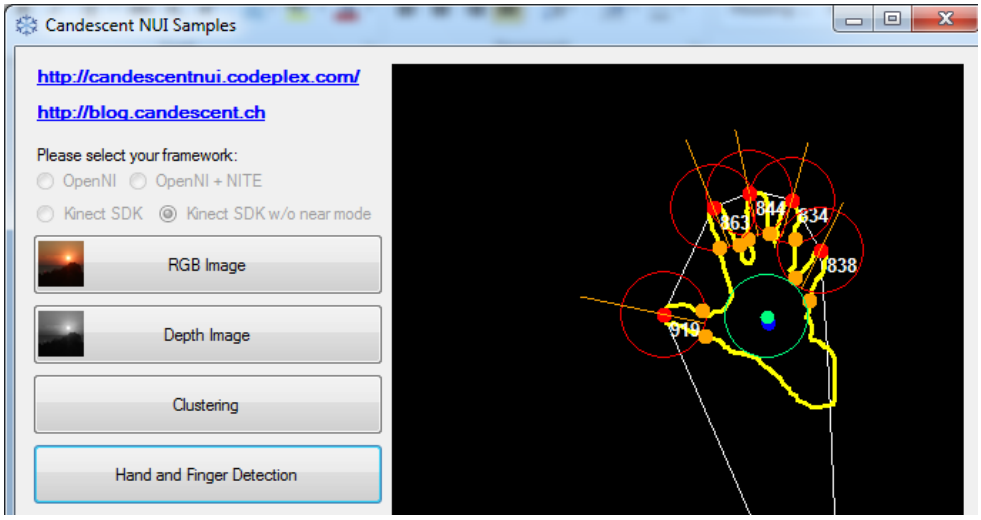
Candescent NUI é um projeto criado por Stefan Stegmueller, desenvolvido com o objetivo de efetuar rastreamento de mãos e dedos, utilizando a informação de profundidade do Kinect. Foi desenvolvido em C# com OpenNI e a Microsoft Kinect SDK. Possui código aberto e permite aos desenvolvedores utilizarem as bibliotecas, desde que os direitos permaneçam no projeto. O framework encontra-se disponível para download no site do

[1] (http://paint.codeplex.com), [2] (http://kinecttoolbox.codeplex.com), [3] (http://projects.ict.usc.edu/mxr/faast)

[4] (http://kinectdtw.codeplex.com)

Candescent NUI sob a modalidade de licenciamento New BSD License (BSD) (CANDESCENT NUI, 2015). A figura 14 demonstra a interface do *framework*.

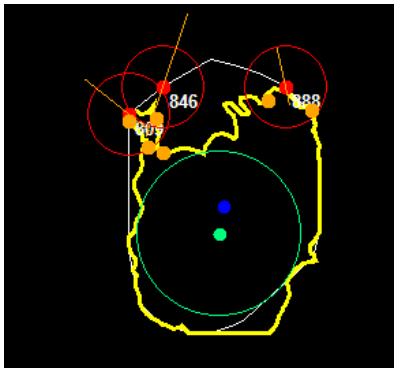
Figura 14 - Rastreamento da mão utilizando o *Framework* Candescent NUI



Fonte: (AUTORES, 2015).

O Candescent NUI inicia detectando objetos próximos ao Kinect, considerando no máximo dois objetos. Estes objetos são tratados para se obter as informações das mãos. Quando identificada uma mão, o algoritmo informa a posição (X, Y e Z) da ponta dos dedos e sua direção, posição da palma da mão, entre outras informações. Caso um objeto tratado não seja uma mão, seja uma cabeça, por exemplo, o algoritmo tenta buscar as informações referentes à mão, mostrando dados errôneos, conforme demonstrado na figura 15.

Figura 15 - Rastreamento errôneo de um objeto diferente de uma mão



Fonte: (AUTORES, 2015).

4.1.1 Campo de visão

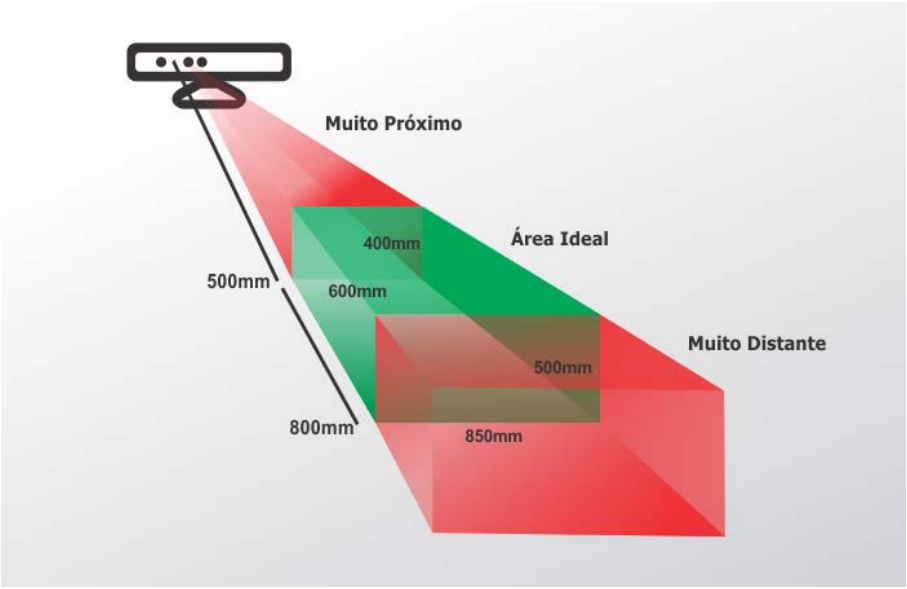
O Kinect para Windows detecta objetos a aproximadamente 400 mm no modo mais próximo. Com o Candescent NUI, as mãos devem estar pelo menos 500 mm de distância do dispositivo para serem detectadas corretamente. A distância máxima é de 800 mm. Além disto, a distância é considerada grande para detecção e rastreio as mãos.

A distância mínima e máxima podem ser customizadas na ferramenta, porém em testes realizados, na tentativa de aumentar a resolução para melhorar o rastreamento das mãos, notou-se que não há ganho diminuindo a distância mínima.

Devido à resolução do sensor de profundidade do Kinect, qualquer movimento mínimo é suficiente para perder o rastreamento correto dos dedos, e qualquer outro objeto que estiver presente no campo de visão do Kinect é suficiente para causar interferência no momento de identificar as mãos.

A distância ideal para utilização do *framework* é em torno de 650 milímetros e, para melhor identificação da mão, o ideal é manter o corpo afastado do campo de visão do Kinect, conforme ilustrado na figura 16.

Figura 16 - Área de funcionamento do Candescent NUI



Fonte: (AUTORES, 2015).

4.1.2 Identificação da mão

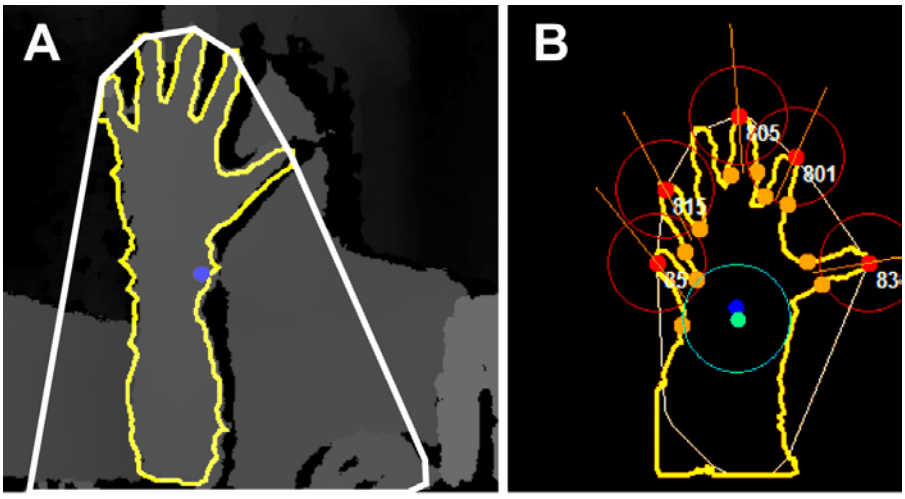
O *framework* consegue detectar até duas mãos simultaneamente. Os dados são armazenados em uma lista chamada de "*HandData*". Caso sejam detectadas duas mãos, a mão da direita será identificada como a primeira, ocupando a posição Hand[0] e a posição Hand[1] fica reservada para a mão esquerda. Cada mão é identificada com um id, e possui as informações de localização, volume, identificação da palma da mão, dedos, forma do contorno, e *Convex Hull* (Polígono formado com as extremidades do objeto reconhecido).

4.1.3 Detecção dos dedos

Na detecção dos dedos não existe identificação sobre os tipos de dedo, como polegar ou indicador, por exemplo. Os dedos são numerados de 0 a 4 em cada uma das mãos.

Os dedos na verdade são os pontos gerados pelo algoritmo C*onvex Hull*, que tem como finalidade gerar o menor polígono que englobe um determinado conjunto de pontos. Na figura 17-A, a linha branca representa o polígono gerado pelo algoritmo, onde as extremidades são consideradas como dedos. Já a figura 17-B demonstra a possibilidade de uma mão ter mais de 5 dedos, devido ao algoritmo utilizado.

Figura 17 - Polígono gerado pelo algoritmo *Convex Hull* e detecção dos dedos



Fonte: (AUTORES, 2015).

O primeiro dedo é sempre representado pelo que estiver mais alto. O restante dos dedos é ordenado no sentido horário. Isto significa que quando a mão gira, os dedos mudam de posição, não havendo uma maneira de travar, para que o rastreio mantenha identificados os dedos.

Os dados em relação aos dedos são armazenados numa lista chamada "*FingerPoints*". Também existe uma função chamado "*FingerCount*" que retorna a quantidade exata de dedos em um determinado momento.

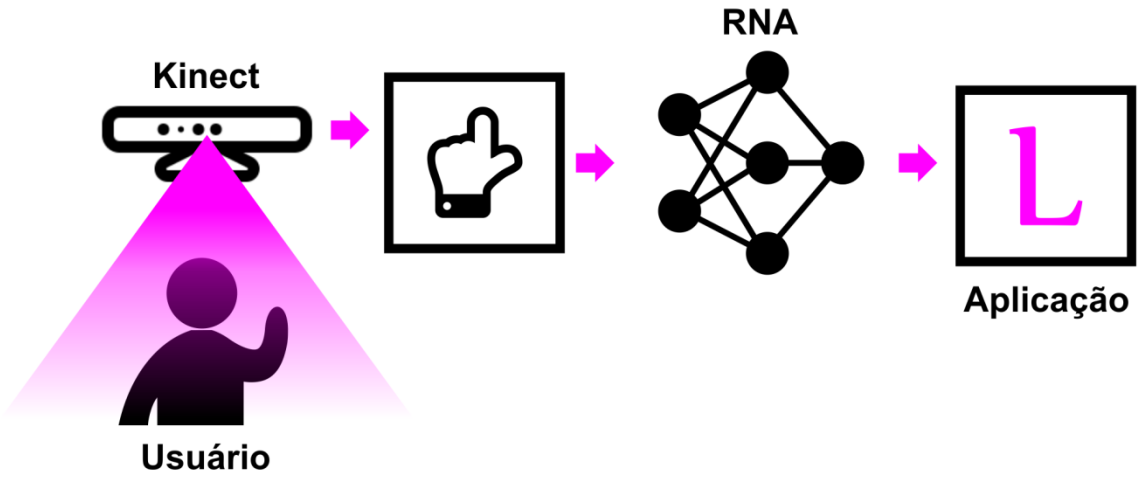
4.2 Aplicação Desenvolvida

Durante a fase de desenvolvimento, foram realizados testes com a ferramenta Candescent NUI para fazer o rastreamento dos dedos, porém a mesma mostrou-se ineficiente, devido à alta variação dos valores, principalmente pela baixa resolução do Kinect em relação ao mapa de profundidade.

Devido à falta de recursos para reconhecimento das *joints* das mãos, optou-se por utilizar a câmera RGB do Kinect para fornecer as entradas para reconhecimento.

De modo geral, a aplicação desenvolvida pode ser dividida em dois módulos. O primeiro, utiliza a câmera RGB do Kinect, da qual são feitas capturas dos padrões a serem reconhecidos. O segundo módulo consiste em uma rede neural artificial para fazer o reconhecimento de posições baseado nas informações capturadas, conforme ilustrado na figura 18. O desenvolvimento da aplicação foi feito por meio de programação *desktop*, utilizando Windows Forms com o *framework* .net e a linguagem C# do Microsoft Visual Studio 2012. Esta tecnologia foi escolhida devido à SDK oficial do Kinect utilizá-la como linguagem padrão.

Figura 18 - Módulos da aplicação desenvolvida



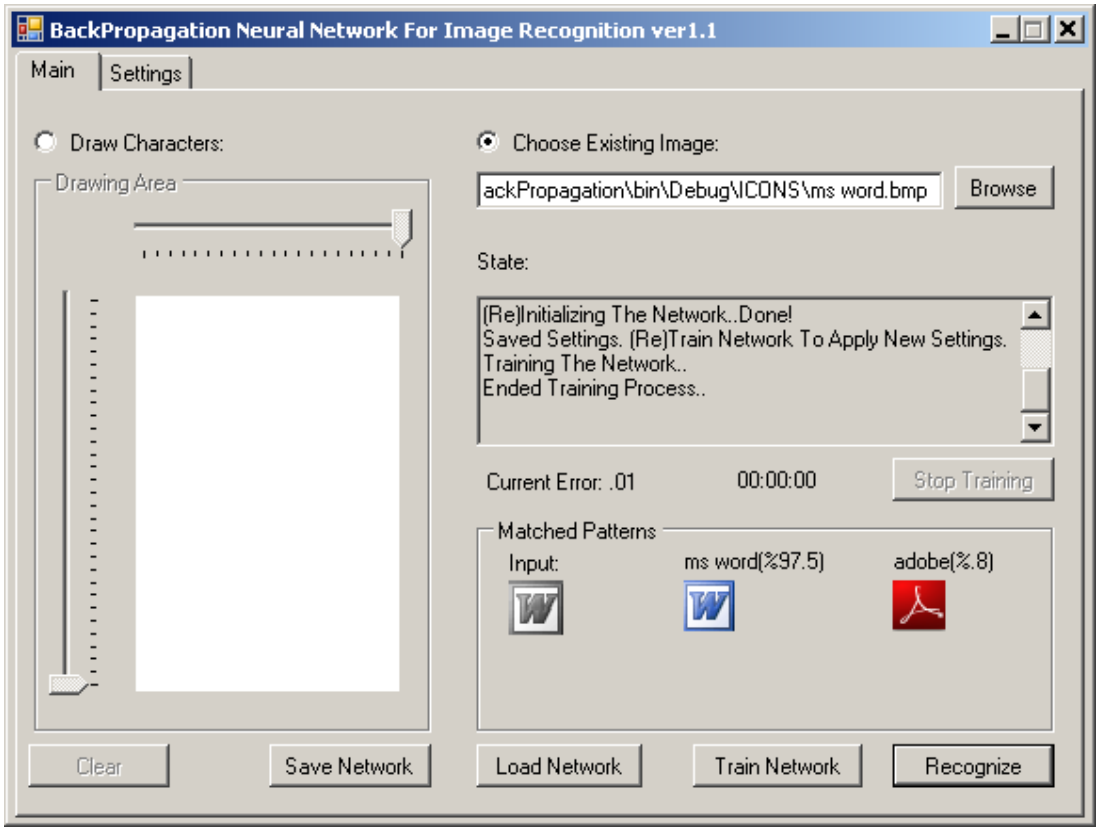
Fonte: (AUTORES, 2015).

A base para desenvolvimento da aplicação foi a utilização do projeto de código aberto, desenvolvido por Murat Firat (2007), chamado *Image Recognition with Neural Networks* (Reconhecimento de Imagem com Redes Neurais), disponível para download no site Code Project[2].

A aplicação foi desenvolvida em C# e possui uma interface simples, que possibilita fazer o reconhecimento de padrões, os quais podem ser desenhados em um painel ou ainda buscando por imagens. O algoritmo utilizado pela rede neural artificial é o *Backpropagation* e a aplicação permite fazer alguns ajustes como por exemplo definir o número de camadas e definir a taxa máxima de erro, a qual é responsável por otimizar o treinamento da rede. Na figura 19 é demonstrada a interface da aplicação. Os três ícones visíveis na tela representam respectivamente o padrão de entrada e dois resultados com a maior taxa de acerto. Ainda é apresentado um registro referente ao processo de treinamento da rede neural artificial na seção *State*.

[2] (http://www.codeproject.com/Articles/19323/Image-Recognition-with-Neural-Networks)

Figura 19 - Image Recognition with Neural Networks



Fonte: (CODE PROJECT, 2015).

Ao iniciar a aplicação, o sistema identifica a quantidade de padrões armazenados em um diretório específico da aplicação chamado *PATTERNS*.

Estes padrões são imagens *Bitmap* (.BMP) com um tamanho de 40 x 40 *pixels*, que são definidos como base de treinamento e como resultado do processamento da rede neural artificial. Para aumentar a base de treinamentos, deve-se adicionar imagens neste diretório e efetuar o treinamento da Rede Neural Artificial.

A quantidade de neurônios da camada de entrada da rede neural artificial é definida pela quantidade de *pixels* do padrão de treinamento. Uma imagem com tamanho 10 x 10 *pixels* por exemplo, possui 100 neurônios na sua camada de entrada. Cada um desses 100 *pixels* possui um valor RGB que representa sua cor. Este valor é convertido para escala de cinza e armazenado em uma matriz de *double* (números reais).

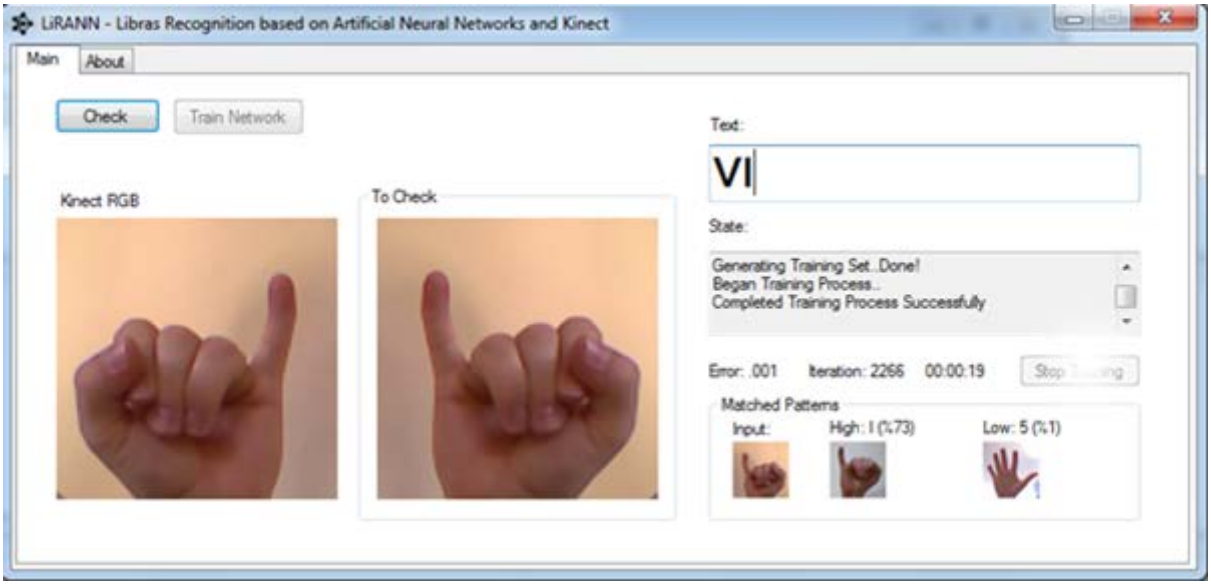
A conversão para cinza é feita através da média dos valores *RED*, *GREEN* e *BLUE*, que são valores de 0 a 255, porém existem relações diferentes, que podem apresentar

melhores resultados como o padrão *Luma Coding*, utilizado na aplicação desenvolvida. Este padrão é definido na proporção de 30% do valor vermelho, 59% do valor verde e 11% do valor azul. Esta proporção considera o fato de o olho humano não enxergar as cores de uma maneira uniforme, percebendo o verde mais forte que o vermelho, e o vermelho mais forte que o azul. (HELLAND, TANNER, 2011)

O tamanho de 40 x 40 *pixels* foi definido após uma série de testes feitos com a rede neural artificial. Com este tamanho verificou-se que é possível fazer o reconhecimento de diferentes padrões. Em testes realizados com imagens maiores, 100 x 100 *pixels*, ou 200 x 200 *pixels* por exemplo, não foi possível executar o treinamento da rede neural artificial devido ao aumento considerável do número de neurônios da camada de entrada. Uma imagem com o tamanho de 200 x 200 *pixels*, possui um total de 40 mil neurônios na camada de entrada, valor 25 vezes maior do que o utilizado com a imagem de 40 x 40 *pixels*, que possui 1600 neurônios.

Para fazer o reconhecimento do alfabeto de Libras, a ferramenta intitula-se LiRANN (*Libras Recognition based on Artificial Neural Networks with Kinect*), conforme demonstrado na figura 20.

Figura 20 – Interface da aplicação desenvolvida

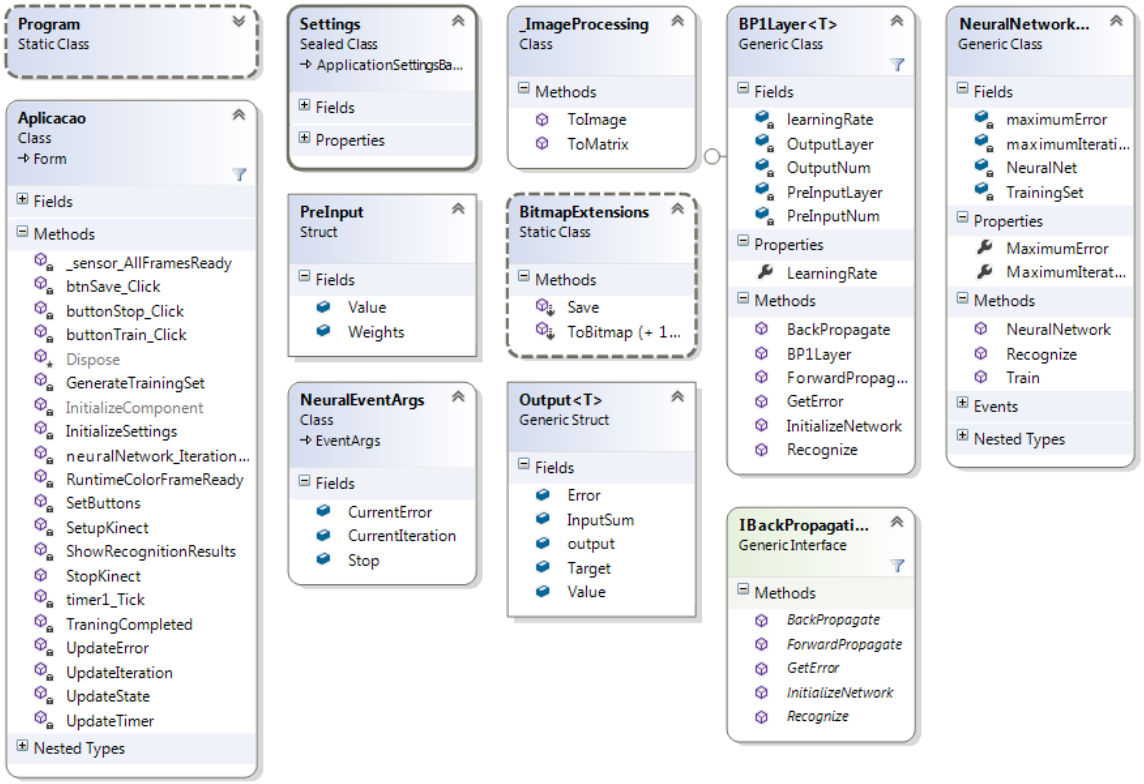


Fonte: (AUTORES, 2015).

Conforme pode-se observar na figura 20, existem dois *Image Frames*, o primeiro, chamado de *Kinect RGB*, é responsável por receber as imagens capturadas pelo Kinect e exibi-las em tempo real. Para realizar esta tarefa, é utilizada a SDK oficial do Kinect. Em relação à área de captura, esta foi limitada para capturar apenas as posições da mão. O segundo *Image Frame*, *To Check* recebe o padrão a ser processado pela rede neural artificial.

Ao executar a aplicação, inicialmente deve-se treinar a rede neural artificial com os padrões definidos no diretório *Patterns*. Após o treinamento, é habilitado o botão que permite fazer o processamento de um padrão. Deve-se então posicionar a mão no espaço de captura do Kinect e pressionar o botão "*Check*". Pressionando este botão, é feita a captura da imagem, a qual é inserida no segundo *Image Frame*. Esta imagem é então dimensionada de acordo com o tamanho médio das imagens na base de treinamento, ou seja 40 x 40 *pixels*, convertida para matriz de *double*, e processada pela rede neural artificial. A rede neural artificial traz como resultado os dois valores com maior taxa de acerto que compõem a base de treinamento e insere o nome do arquivo, que corresponde às letras do alfabeto, com maior taxa de acerto no campo de texto, conforme demonstrado na figura 20. Na figura 21 é apresentado o digrama de classes da aplicação desenvolvida, destacando os métodos utilizados na aplicação.

Figura 21 – Diagrama de classes LiRANN



Fonte: (AUTORES, 2015).

A seguir é explicado o funcionamento do algoritmo *Backpropagation*, utilizado pela rede Neural Artificial na aplicação.

4.3 Algoritmo *Backpropagation*

Redes Neurais Artificiais (RNAs) têm a capacidade de se adaptar, aprender, generalizar ou organizar os dados. Há muitos modelos de RNAs incluindo, *Perceptron*, *Adaline*, *Madaline*, *Kohonen*, *Backpropagation*, entre outros. A estrutura *Backpropagation* é a mais comum por ser simples de implementar e eficaz.

Redes Neurais Artificiais que utilizam a estrutura *Backpropagation* contêm duas ou mais camadas, cada uma das quais está ligada à camada seguinte. A primeira camada é chamada de camada de entrada, na qual é inserida a entrada inicial ou padrão de entrada (por exemplo, *pixels* de uma letra). A última camada é chamada de camada de saída, normalmente mantém o identificador da entrada - *input* (por exemplo, nome da letra de entrada). As camadas entre a camada de entrada e saída são chamadas de camadas ocultas.

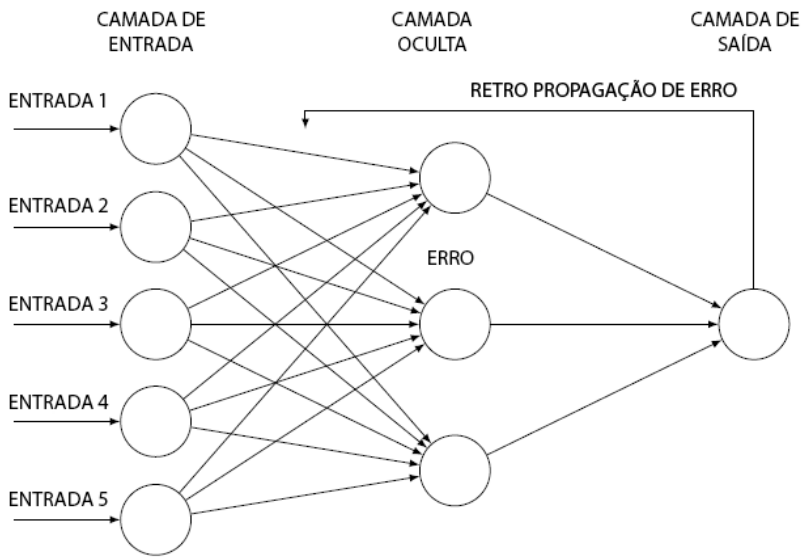
De forma geral, o algoritmo *Backpropagation* é um algoritmo baseado no paradigma de aprendizado supervisionado, e se baseia em um mecanismo de correção de erros em que os erros da camada de saída vão sendo propagados para as camadas ocultas a fim de se realizar a correção dos pesos destas camadas (COSTA E SILVA, 2003).

O processo de aprendizado da rede *Backpropagation* é dividido em duas etapas, sendo a primeira uma fase de alimentação da rede, conhecida como *forward*, onde as informações são propagadas para frente, resultando em uma saída. A segunda etapa consiste em uma fase de correção dos pesos no sentido contrário, ou retro propagação dos erros, conhecida como fase *backward*, onde os pesos da RNA são atualizados para que a rede neural artificial atinja um valor mais próximo ao desejado.

O treinamento da rede neural é feito em diversos ciclos das etapas *forward* e *backward*, até que atinja um valor de erro aceitável e apresente o resultado esperado pelo usuário. Na aplicação para identificação do alfabeto de Libras, o treinamento foi executado com 10 padrões. O treinamento foi efetuado em um total de 2163 ciclos, levando em torno de 22 segundos para ser realizado, considerando uma taxa de erro de 0.001. A máquina utilizada nos testes foi um Dell Vostro, processador Intel Core I5 2.53GHz com sistema operacional Windows 7 32 bits e 4 GB de memória RAM.

Na figura 22 é apresentado um exemplo de Rede Neural Artificial *Backpropagation*. Os nós localizados à esquerda, são as entradas iniciais, ao centro a camada oculta e ao lado direto localizam-se os nós de saída.

Figura 22 – Representação de uma rede neural artificial *backpropagation*



Fonte: (CODE PROJECT, 2015).

Para validar a aplicação desenvolvida e a rede neural artificial utilizada, foram realizados testes com o conjunto de treinamento, conforme demonstrado no quadro 2. Foi possível escrever as seguintes palavras utilizando a aplicação: LIBRA, VACA, LIVRO, VIVER, entre outras.

O quadro 2 apresenta resultados médios de convergência obtidos por meio do processamento do LiRANN com a base de treinamentos utilizada para validação da aplicação. Para determinar a taxa de acerto, foram feitas 20 tentativas com cada uma das posições demonstradas no quadro. Uma taxa de acerto de 40%, por exemplo, representa que das 20 tentativas, 8 vezes o resultado apresentado convergiu ao padrão esperado.

Em relação à rede neural artificial utilizada, pôde-se observar que em algumas situações ela não convergiu ao resultado esperado. Principalmente, em situações onde não há uma distinção acentuada da posição da mão, como no caso da letra ‘A’, e da letra ‘E’. No entanto, a RNA mostrou-se eficiente para reconhecer padrões quando a base de treinamento não é muito grande, como, por exemplo, uma quantidade de 10 padrões, sendo composta por imagens com posições bem distintas entre si.

Quadro 2 - Base utilizada para validação e taxa de convergência

LIBRAS	Transcrição	Taxa de Acerto
	5	80%
	A	40%
	B	90%
	C	40%
	E	10%
	I	70%
	L	90%
	O	10%
	R	80%
	V	90%

Fonte: (AUTORES, 2015).

5. CONCLUSÃO

Este trabalho apresentou a aplicação que identifica as posições do alfabeto da língua brasileira de sinais, utilizando o Kinect como forma de entrada e redes neurais artificiais para fazer o reconhecimento e processamento das informações. A partir dos conhecimentos adquiridos durante a fase de pesquisa, constatou-se que somente seria possível fazer o reconhecimento das posições estáticas, portanto, as posições do alfabeto que exigem movimento, correspondentes às letras h, j, k, x, y, e z, não foram contempladas na execução do trabalho.

Existiram diversos desafios para o desenvolvimento deste trabalho, desde o estudo da linguagem de programação C#, a busca por algoritmos de identificação e reconhecimento das mãos e dedos, e ainda o estudo de uma rede neural artificial que fosse aplicável para a solução do problema proposto. Também foi feita uma pesquisa e estudo por trabalhos relacionados, conforme apresentado no Quadro 1, seção 3.6 do capítulo 3, onde há uma comparação entre os trabalhos relacionados estudados. Retomando-se o quadro, inclui-se aqui as características da aplicação desenvolvida.

Quadro 3 - Quadro referente a aplicação desenvolvida

Trabalho	Objetivo do Trabalho	Dispositivos e Técnicas Utilizadas	Questões a Destacar
LiRANN – Sistema de Reconhecimento de LIBRAS baseado em Redes Neurais Artificiais com Kinect	Desenvolver uma aplicação que interprete e traduza o alfabeto da língua brasileira de sinais em tempo real, utilizando o Kinect e Redes Neurais Artificiais	São utilizados os recursos da câmera RGB do Kinect e Rede Neural Artificial *Backpropagation*	Reconhecimento das posições estáticas do alfabeto.

Fonte: (AUTORES, 2015).

Como contribuição social e científica deste trabalho, pode-se citar a possibilidade de o mesmo vir a ser utilizado por pessoas com deficiências auditivas no aprendizado da língua brasileira de sinais. Notou-se no estudo a escassez de ferramentas e trabalhos desenvolvidos nesse sentido, demonstrando possibilidades de avanço neste campo de pesquisa.

Deve-se citar que a plataforma Kinect, escolhida para implementação do projeto, mostra-se eficaz e viável para aplicação de diversos projetos devido ao grande número de tutoriais e códigos da própria Microsoft, os quais facilitam o estudo da plataforma, porém constatou-se que o mesmo ainda não é a ferramenta ideal para a solução do problema proposto, devido à falta de recursos referentes à identificação dos dedos das mãos e suas *joints*.

Em uma pesquisa realizada sobre a nova versão do Kinect, a qual foi lançada em julho de 2014, nota-se uma evolução do dispositivo. Diferente do Kinect 1, versão utilizada no trabalho, a qual possui câmera VGA, esta nova versão possui câmera *Full HD*. Além disto, há um grande avanço do dispositivo em relação ao reconhecimento das mãos durante a leitura do esqueleto, ao qual foram adicionadas novas *joints*, que identificam o polegar e o dedo indicador. Ainda, nesta nova versão, o sensor de profundidade foi aprimorado e permite que o usuário interaja mais perto do aparelho.

Baseado nesta evolução do dispositivo, acredita-se que futuramente o mesmo irá abranger todas as *joints* dos dedos da mão, o que permitirá fazer o reconhecimento das posições através de coordenadas (X, Y e Z) ao invés de utilizar imagens, conforme feito neste trabalho.

Nos testes realizados com a rede neural artificial, constatou-se que sua utilização para reconhecimento de imagens demanda alto poder de processamento devido ao grande número de neurônios na camada de entrada. Também foi possível notar o aumento na demanda de processamento, na medida em que a base de treinamentos era incrementada. Das dificuldades apresentadas pelo processamento da RNA, pode-se citar o problema em fazer o reconhecimento em ambientes onde não há iluminação adequada. Outros problemas encontrados foram em relação à distância e o ângulo da imagem de captura. Caso seja muito diferente do padrão treinado, a rede neural artificial pode não convergir ao resultado esperado.

Conclui-se que no momento em que for possível mapear as coordenadas das *joints* referente aos dedos das mãos, será possível reduzir o tamanho da entrada, tornando mais rápido e eficaz o processamento.

Como melhorias e sugestões para trabalhos futuros, citam-se: Rastreamento da mão no ambiente, ao invés de utilizar uma área específica como no trabalho desenvolvido; O acionamento automático do *script* de reconhecimento quando a mão estiver em uma

determinada posição; O desenvolvimento de uma etapa de captura para treinamento da rede neural; Possibilidade de treinar a rede com múltiplas imagens para uma saída. Validar a aplicação com usuários de LIBRAS.

6. REFERÊNCIAS

BARROS, Jussara. Libras – Forma de comunicação. Disponível em: http://www.escolakids.com/libras-forma-de-comunicacao.htm. Acesso em 27 mar. 2015.

CANDESCENT NUI. Hand and finger tracking with Kinect depth data. Disponível em: https://candescentnui.codeplex.com/. Acesso em 21 jun. 2015.

CAPOVILLA, F. C.; RAPHAEL, W. D. Dicionário Enciclopédico Ilustrado Trilíngue da Língua de Sinais Brasileira LIBRAS. São Paulo, Brasil: Editorada Universidade de São Paulo, 2001.

CEFET-RJ, 2013. Construção de ambiente para desenvolvimento de jogos educacionais baseados em interface de gestos. Disponível em: http://www.upf.br/seer/index.php/rbca/article/view/3270. Acesso em 21 jun. 2015.

COSTA E SILVA, Luiz Fernando. Modelo de Rede Neural Artificial Treinada com o Algoritmo Backpropagation. UFJF. Juiz de Fora, MG, Fevereiro de 2003 – Disponível em: http://ufjf.googlecode.com/svn/trunk/T%C3%B3picos%20em%20Computa%C3%A7%C3%A3o%20Cient%C3%ADfica%20II%20-%20Data%20Mining/MonogLuizFernando.pdf. Acesso em 29 nov. 2015.

CRAWFORD, S. - How Microsoft Kinect Works - 13 July 2010. HowStuffWorks.com. Disponível em: http://electronics.howstuffworks.com/microsoft-kinect.htm.Acesso em 21 jun. 2015.

DOMINGOS, Maria Cristina da Silva. Cartilha de Libras, UNIFENAS, Alfenas, 2010. Disponível em: http://www.unifenas.br/extensao/cartilha/CartilhaLibras.pdf. Acesso em 27 mar. 2015.

FIALHO, A. R. S. Estudo de Técnicas de Rastreamento das Mãos para o Desenvolvimento de Interfaces Homem-Máquina. Campo Grande, Brasil, 2004.

FISHER, M. Kinect. Disponível em https://graphics.stanford.edu/~mdfisher/Kinect.html. Acesso em 21 jun. 2015.

FREEMAN, J.A.; SKAPURA, D.M. Neural Networks: Algorithms, Applications and Programming Techniques. Addison-Wesley, 1992.

FROZZA, R. Redes Neurais Artificiais: Material de Apoio. Universidade de Santa Cruz do Sul Departamento de Informática, Curso de Ciência da Computação, 2014.

GASSEN, Edner dos Reis Gassen. O uso do Kinect na acessibilidade de pessoas com deficiência visual, UNISC, Santa Cruz do Sul, 2013.

HELLAND, Tanner. Seven grayscale conversion algorithms (with pseudocode and VB6 source code). Disponível em: http://www.tannerhelland.com/3643/grayscale-image-algorithm-vb6/. Acesso em 27 nov. 2015.

JANA, A. Kinect for Windows SDK Programming Guide. Disponível em http://lirec.ict.pwr.wroc.pl/~jkedzier/urbi/Kinect%20for%20Windows%20SDK%20Programming%20Guide.pdf. Acesso em 21 jun. 2015.

JANELATECH. REVIEW – Review – Prodeaf Aplicativo de Libras. Disponível em: http://www.janelatech.com.br/ultimos-posts/windows-phone/review-prodeaf-aplicativo-de-ibras/. Acesso em 21 jun. 2015.

KHOSHELHAM, K. Accuracy and Resolution of Kinect Depth Data for Indoor Mapping Applications. Disponível em http://www.mdpi.com/1424-8220/12/2/1437/htm. Acesso em 21 jun. 2015.

KINECTDTW. Kinect SDK Dynamic Time Warping (DTW) Gesture Recognition. Disponível em: http://kinectdtw.codeplex.com/. Acesso em 21 jun. 2015.

LIBRAS – Disponível em: http://www.cbsurdos.org.br/libras.htm. Acesso em 21 jun. 2015.

MICROSOFT. Kinect for Windows – Meet Kinect. Disponível em: http://www.microsoft.com/en-us/kinectforwindows/meetkinect/default.aspx. Acesso em 27 mar. 2015.

MORRISSEY, S. Data-Driven Machine Translation for Sign Languages. Tese (Doutorado) —Dublin City University, Dublin, Irlanda, 2008

MSDN MICROSOFT - Kinect for Windows Sensor Components and Specifications. Disponível em https://msdn.microsoft.com/en-us/library/jj131033.aspx. Acesso em 21 jun. 2015.

PISTORI, H.; NETO, J.J. Tecnologia Adaptativa em Engenharia de Computação: Estado da Arte e Aplicações. Tese de Doutorado. USP. São Paulo, Brasil, Dezembro, 2003 – Disponível em: http://www.gpec.ucdb.br/pistori/f_publications.htm. Acesso em 21 jun. 2015.

ROCKEMBACH, Marcelo. Utilização do Microsoft Kinect na captação de características corporais de usuários, UNISC, Santa Cruz do Sul, 2014.

RODRIGUES, Telma Tietre. Libras – Língua Brasileira de Sinais, Universidade Estácio de Sá, Rio de Janeiro, 2007. Disponível em: http://portal.estacio.br/media/1868413/cartilha_lingua_brasileira_de_sinais.pdf Acesso em 31 mar. 2015.

TAFNER, M. Redes Neurais Artificiais: Aprendizado e Plasticidade. Universidade Estadual de Campinas, Revista Cérebro & Mente, 1998.

TANG, M. Recognizing Hand Gestures with Microsoft's Kinect - Department of Electrical Engineering Stanford University. Disponível em: http://uran.donetsk.ua/~masters/2012/fknt/sobolev/library/article5.pdf. Acesso em 21 jun. 2015.