Interface humano-computador com identificação de gestos da mão

Matheus Filipe Lourenço

Trabalho realizado sob a orientação de

Fernando Monteiro

Felipe Walter Dafico Pfrimer

Mestrado em Engenharia Industrial

2019-2020

Interface humano-computador com identificação de gestos da mão

Trabalho realizado sob a orientação de

Fernando Monteiro

Felipe Walter Dafico Pfrimer

Relatório Final do Trabalho de Projecto de Engenharia Industrial apresentado à Escola Superior de Tecnologia e de Gestão do Instituto Politécnico de Bragança e a Universidade Tecnológica Federal do Paraná de Toledo.

Matheus Filipe Lourenço

2019-2020

A Escola Superior de Tecnologia e de Gestão não se responsabiliza pelas opiniões expressas neste relatório.

Declaro que o trabalho descrito neste relatório é da minha autoria e é da minha vontade que o mesmo seja submetido a avaliação.

Matheus Filipe Lourenço

Agradecimentos

Primeiramente agradeço aos meus pais, Gilberto Lúcio Lourenço e Adriana Batista de Arruda Lourenço, que me deram todo o apoio necessário para eu chegar ao ponto que estou hoje, pois sem eles nada disso seria possível.

Agradeço também, a UTFPR que me deu a oportunidade de realizar o mestrado em Portugal, assim todo seu corpo docente que me ensinou e instruiu o caminho a ser seguido. Em especial o Professor Fábio Rizental Coutinho que me orientou e ajudou durante os anos letivos, e o Professor Felipe Walter Dafico Pfrimer que aceitou o desafio de me orientar do Brasil e me ajudou e se preocupou durante a escrita da tese.

Sou grato pela oportunidade que o IPB me proporcionou em participar de um mestrado na Europa, em uma instituição de renome e recursos. Em especial o Professor Fernando Monteiro que me orientou e auxiliou no processo da realização da tese.

Também agradeço ao amigos se sempre estiveram presentes, mesmo de longe, dando apoio moral quando preciso.

Por ultimo, mas não menos importante, agradeço a minha namorada, Jessica Sibila Guardezi, que esteve presente em todos os momentos difíceis e alegres dessa empreitada.

Resumo

A comunicação é considerada uma ferramenta de importante uso em todos os tipos de relações, que pode ser realizada de forma verbal ou não verbal. Tratando-se de uma pessoa surda, a linguagem gestual portuguesa (LGP - Portugal), ou linguagem brasileira de sinais (Libras - Brasil), contribui para a comunicação dos surdos, que enfrentam barreiras que dificultam sua comunicação, em locais de trabalho, vida social entre outros. Considerando a grande comunidade de surdos no mundo, que necessitam de ajuda para se comunicar e que tenham facilidade em realizar essa comunicação, e também, tendo em conta o avanço da tecnologia nos últimos anos, surgiu a possibilidade da criação de interfaces humano-computador, mais simples e de fácil acesso e compreensão. Neste contexto, e com o uso da Inteligência Artificial, o principal objetivo deste trabalho é estudar a possibilidade de criar uma ferramenta que possa realizar a identificação e tradução das letras do alfabeto para computador através da criação de um sistema composto pela captura e tradução de gestos a partir de um Kinect mais o Software MatLab. Com os estudos realizados neste trabalho, observou-se que o dispositivo possui precisão de 96.71% em relação ao gestos treinados e apresenta a possibilidade de realizar testes e tradução em tempo real.

Palavras-chave: Comunicação, LGP, Inteligência Artificial, Kinect.

Abstract

Communication is considered a tool of important use in all types of relationships, which can be carried out verbally or non-verbally. In the case of a deaf person, Portuguese sign language (LGP - Portugal), or Brazilian sign language (Libras - Brazil), contributes to their communication, who face barriers communication, in workplaces, life among others. Considering the large community of deaf people in the world who needs help to communicate and easily communicates, and also, taking into account the advancement of technology in recent years, the possibility of creating human-computer interfaces came up, simpler and easy to access and understand. In this context, and with the use of Artificial Intelligence, the main objective of this work was to study the possibility of creating a tool that can perform the identification and translation of the letters of the alphabet to a computer through the creation of a system composed by the capture and translation of gestures from a Kinect plus MatLab Software. With the studies carried out in this work, it was observed that the device has 96.71% accuracy in relation to the trained gestures and the possibility to carry out tests and real time translation.

Keywords: Communication, LGP, Artificial Intelligence, Kinect.

Conteúdo

Lista de Tabelas

Lista de Figuras

Siglas

Adam Adaptive Moment Optimization. 29, 31, 38, 39

ANTLR Another Tool for Language Recognition. 6

API Interface de Programação. 8

APS Associação Portuguesa de Surdos. 2

CAD Desenho assistido por computador. 7

CNN Convolutional Neural Network. 17, 18, 33

DSL Domain-Specific Language. 5

IDE Integrated Development Environment. 21

LED Light Emitting Diode. 7, 14

LEG Linguagem para Especificação de Gestos. 5

LGP Linguagem Gestual Portuguesa. 1–3, 5, 30, 31, 39, 43

NUI Interface Natural. 6

RGB Red, Green e Blue. 12, 14

RMSprop Root Mean Square Propogation. 29, 31, 36

SDK Kit de Desenvolvimento de Software. 6, 8, 21

SGD Stochastic Gradient Descent. 18

SGDM Stochastic Gradient Descent with Momentum. 29, 31, 33, 36, 38, 39

USB Universal Serial Bus. 9, 15

Capítulo 1

Introdução

Neste capítulo contextualiza-se a Linguagem Gestual Portuguesa (LGP) com os objetivos da dissertação, alem de descrever a estrutura deste documento.

1.1 Linguagem Gestual Portuguesa

A LGP tem grande importância na comunidade surda, e também por toda a comunidade envolvente, como familiares de surdos, educadores, professores, técnicos, entre outros. Pois através dessa linguagem, permite que pessoas se comuniquem através de gestos ou sinais, sem a necessidade da fala.

As primeiras referências à educação dos surdos reportam ao séc. VIII pela mão do Arcebispo de York, mas é o Frei Ponce de Léon (1520-1584), considerado o primeiro professor de surdos, que é confiada a educação de várias crianças surdas da nobreza espanhola. Naquela época os mudos/surdos não eram reconhecidos como pessoas nos termos da lei, e a aprendizagem de uma língua era essencial para que essas crianças pudessem herdar os títulos e as propriedades das suas famílias [1].

Atualmente, todos têm direito ao ensino com garantia do direito à igualdade de oportunidades de acesso e êxito escolar, e se deve proteger e valorizar a língua gestual portuguesa, enquanto expressão cultural e instrumento de acesso à educação e da igualdade de oportunidades, segundo o artigo 74º da constituição portuguesa [2].

Com o intuito de manter esse regulamento, nasceu a primeira Associação de Surdos em Portugal, com sede em Alvalade e tendo como primeira presidente Maria Madalena Pires. Em 1958, Portugal passava por uma época de grande destaque na educação dos surdos, pois aqueles que mais se inclinavam para a língua gestual, com o entusiasmo e a vontade de aprender, se reuniam em um espaço cedido pela Sociedade Filarmónica João Rodrigues Cordeiro, na Rua da Fé. A partir disso, foi criada a Associação Portuguesa de Surdos (APS), no dia 24 de Setembro do mesmo ano.

1.2 Objetivos

Tendo em vista, o aumento da tecnologia humano-computador, o presente trabalho tem como objetivo principal, realizar um estudo acerca da possibilidade e das vantagens de se utilizar um sistema que traduza a LGP para um computador. Isto possibilitará uma maior facilidade de estudo e pratica da LGP.

A LGP possui 26 gestos para todo o alfabeto, como mostra a figura 1.1.

Figura 1.1: Alfabeto da LGP [3].

Com base nisso, é estudado a possibilidade de fazer a tradução das letras do alfabeto para o computador através do periférico Kinect, um dispositivo inicialmente criado

para jogos, mas que possui uma câmera de profundidade que possibilita a captura dos movimentos do corpo humano.

Alem disso, também tem como objetivo identificar comandos para um computador, por sinais de mãos, utilizando em tarefas onde não seja possível o contacto dos dedos com o teclado ou com um ecrã táctil.

1.3 Estrutura do Documento

Este trabalho foi divido em 5 capítulos para uma melhor compreensão do mesmo.

- Capítulo 1: Introdução
  - Contextualização da LGP com o objetivo do trabalho.
- Capítulo 2: Estado da Arte:
  - É analisado os equipamentos existentes que realizam a captura de movimento do corpo humano. Assim como uma base teórica sobre Redes Neuronais.
- Capítulo 3: Materiais e Métodos
  - Explica o funcionamento do sistema, desde a sua montagem e inicialização, até a obtenção de resultados.
- Capítulo 4: Resultados e Discussões
  - Apresenta e explica os resultados obtidos do sistema.
- Capítulo 5: Conclusões
  - Discute se o trabalho atingiu os objetivos propostos, como também possíveis trabalhos futuros.

Capítulo 2

Estado da Arte

Neste capítulo é abordado as tecnologias existentes que utilizam um sistema capaz de capturar os movimentos do corpo humano, assim como a teoria por trás da identificação dos gestos proposto por este trabalho.

2.1 Trabalhos Semelhantes

Existem trabalhos com objetivos e propostas semelhantes a apresentada neste trabalho, porém há diferenças na metodologia e softwares empregados para resolução da mesma.

Foi visto no trabalho de [4] propostas de soluções para o reconhecimento automático de gestos da LGP através do sensor Kinect. Para resolução da proposta foi utilizado o software Microsoft Visual Studio 2013 com a linguagem de programação C# na versão. Net Framework 4.5 e também foi criado uma interface de utilização para visualização, captura e reconhecimento. Estas soluções deram resultados positivos, dado que gerou-se uma alta porcentagem de precisão.

Também foi visto em [5] a especificação e implementação da Linguagem para Especificação de Gestos (LEG), uma Domain-Specific Language (DSL) para a especificação e reconhecimento de gestos livres do corpo humano com suporte a diferentes dispositivos de profundidade. A LEG é uma linguagem declarativa, baseada na análise das interfaces gestuais para computador e no estudo das abstrações e representações do movimento

humano, a fim de reduzir a complexidade no desenvolvimento de aplicações baseadas em gestos. Este trabalho utilizou o Kinect com o auxilio do Another Tool for Language Recognition (ANTLR), um gerador de analisador de cima para baixo para um subconjunto de linguagens livres de contexto.

Foi proposto em [6] um sistema de captura de baixo custo destinado a determinação da cinemática humana. A fim de atender este requisito, foi utilizado o dispositivo Kinect, por ser barato, leve, portátil, de fácil uso e dispõe de dados tridimensionais com apenas um periférico. No trabalho desenvolvido utilizou-se um software utilizando a ferramenta QtCreator, que é um Kit de Desenvolvimento de Software (SDK) para produção rápida de programas com interface gráfica convencional.

O artigo [7] apresentou a possibilidade da utilização de redes neurais artificiais aplicadas no reconhecimento de gestos utilizando o Kinect para realizar o acompanhamento da mão direita do usuário, para que fosse possível assim fornecer comandos a computadores, aplicações, jogos, ou até mesmo comandar robôs à distância através de uma Interface Natural (NUI). Nele demonstrou a viabilidade de realizar o aprendizado e reconhecimento de gestos em tempo real através do uso de um Kinect e de redes neurais artificiais.

2.2 Equipamentos

Nesta secção será mostrado recursos disponíveis no mercado que possuem foco na captura de movimento do corpo humano, como:

- Leap Motion Controller;
- Xtion PRO LIVE;
- CyberGlove III.
- Microsoft Kinect;

2.2.1 Leap motion

Leap Motion Controller mostrado na figura 2.1, é um modulo óptico de rastreamento de mãos desenvolvido pela empresa Leap Motion Inc. e atualmente é fabricado e distribuído pela Ultraleap, hoje o produto é vendido por 89.95 €. Uma licença é necessária para poder utiliza-lo [8], sendo elas:

- Licença de desenvolvedor (gratuita): somente uso não comercial;
- Licença comercial: empresas que incorporam o hardware e/ou software de rastreamento Ultraleap em um produto revendido e qualquer empresa que desenvolve um aplicativo para equipamentos industriais, militares, comerciais, médicos ou Desenho assistido por computador (CAD);
- Licença corporativa: aplicativos apenas para uso comercial interno (por exemplo, ferramentas de CAD ou de produtividade).

Figura 2.1: Leap Motion Controller [9].

O Motion Controller, possui dois sensores de imagem, e três Light Emitting Diode (LED) infravermelhos que enviam dados para o computador para rastrear o movimento

das mãos [9].

O dispositivo pode ser usado para aplicativos de produtividade com computadores Windows ou Mac, integrados a soluções ou monitores de hardware corporativos, conectados a fones de ouvido de realidade virtual aumentada para prototipagem, pesquisa e desenvolvimento.

O controlador é capaz de rastrear mãos dentro de uma zona imaginaria interativa 3D que se estende até um raio de 60 cm (24 ") ou mais, como indicado na figura 2.2, estendendo-se do dispositivo em um campo de visão de 120x150º. O software da Leap Motion é capaz de discernir 27 elementos das mãos, incluindo ossos e articulações, e rastreia mesmo quando obscurecidos por outras partes da mão, além de possuir uma taxa de atualização de 120 Hz.

Figura 2.2: Leap Motion Controller em funcionamento [9].

Ele possui um SDK: Interface de Programação (API) de estilo C chamada LeapC e uma ligação oficial em C#. (Ligações mais antigas para C ++, Java, JavaScript, Python e Objective-C estão disponíveis, mas não são mais ativamente suportadas) [10].

2.2.2 Xtion PRO LIVE

Xtion PRO LIVE, mostrado na figura 2.3, é um dispositivo criado pela ASUS com o objetivo de que seus consumidores desenvolvam aplicativos e jogos com detecção de movimento. Atualmente é vendido por 300 €.

Figura 2.3: Xtion PRO LIVE [11].

Ele usa sensores infravermelhos, tecnologia adaptativa de detecção de profundidade, detecção de imagem em cores e fluxo de áudio para capturar a imagem, movimento e voz em tempo real dos usuários, tornando o rastreamento do usuário mais preciso. A solução de desenvolvimento do Xtion PRO LIVE vem com um conjunto de ferramentas de desenvolvedor para facilitar a criação de seus próprios aplicativos baseados em gestos, sem a necessidade de escrever algoritmos de programação complexos.

O produto realiza a detecção das mãos através de gestos pré-definidos para permitir que você pressione, clique, circule, acene e mais, além de detectar o corpo inteiro.

O Xtion PRO LIVE apresenta um design Universal Serial Bus (USB) plug and play, com a solução de desenvolvimento Xtion PRO LIVE compatível com o middleware OPNI NITE [11].

A tabela 2.1 apresenta as especificações do Xtio PRO LIVE.

Tabela 2.1: Especificações do Asus Xtion Pro Live

Distancia de uso	Entre 0.8m e 3.5m
Campo de visão	58º Horizontal, 45º Vertical
Sensores	RGB, Depth, Microphone
Resolução	RGB: 1280 X 1024, Depth: 640 x 480
Interface	USB 2.0/3.0
Software	OpenNI SDK bundled
Suporte OS	Win 32/64 : XP, Vista, 7, 8 Linux : X86, 32/64 bit Android
Linguagem de Programação	C++/C#(Windows) C++(Linux) Java

2.2.3 CyberGlove III

O CyberGlove III, como mostrado na figura 2.4, é um dispositivo capaz de realizar a captura do movimento das mãos e dedos através de 22 sensores, é capaz de transmitir dados sem fio através do Wi-Fi e sua bateria dura cerca de duas a três horas [12].

Figura 2.4: CyberGlove III

Este dispositivos precisa estar de alguma forma anexado ao corpo do usuário. Com isso, o usuário precisa constantemente se preocupar com peso do aparelho, se o acessório está fixo ao corpo para evitar lançá-lo para longe durante execução de um movimento

rápido. Desta forma, mesmo que bem projetado para o uso comum do corpo humano, este dispositivo tende a causar limitações de interação no usuário durante o uso. Diferente dos dispositivos que câmeras e sensores a longa distancia, deixando o corpo mais livre [5].

2.2.4 Microsoft Kinect

O Kinect apareceu em 4 de novembro de 2010 como um acessório para Console Xbox 360, ilustrado na figura 2.5. É um dispositivo desenvolvido pela empresa Prime Sense, em colaboração com a Microsoft. Em janeiro de 2012, mais de 18 milhões de unidades foram vendidas e em fevereiro de 2012, uma versão para Windows foi lançada [13].

Figura 2.5: Microsoft Kinect v1 [14].

Este dispositivo fornece quadros de imagem em cores e profundidade através da câmera RGB e depth, respectivamente, e dados de áudio da matriz do microfone para o SDK. Além de reconhecer movimentos do corpo humano através de suas articulações.

Quando lançado em sua primeira versão para XBOX 360, não havia uma interface de software para que ele fosse utilizado no computador. Pouco tempo após o lançamento, desenvolvedores independentes conseguiram criar uma biblioteca de funções que permitia acessar os dados do sensor, e através disso, foi possível utilizá-lo em computadores [5].

Diante disso, a própria Microsoft decidiu lançar uma segunda versão do Kinect especialmente para computadores. Essa versão é denominada Microsoft Kinect for Windows. Junto a ela, foi disponibilizado de forma oficial e gratuita, o Kit de Desenvolvimento de Software, com suporte da Microsoft aos desenvolvedores [5].

Em novembro de 2013, em conjunto com o lançamento do novo console XBOX ONE, foi apresentado a nova versão do Kinect, denominado Kinect for Windows v2 visto na figura 2.6. Diferente do seu antecessor, a mesma versão do Kinect v2 foi lançado para as duas plataformas, console e computador, sendo necessário apenas um adaptador para conexão ao computador. O novo Kinect avançou consideravelmente em relação a precisão, sendo possível identificar movimento das mãos através das articulações dos dedos. Além disso, considera os movimentos de até seis pessoas ao mesmo tempo, onde suas versões anteriores somente eram possível identificar duas [5].

Figura 2.6: Microsoft Kinect v2 [15].

O Hardware do Kinect v1 possui os seguintes componentes conforme mostra a figura 2.7

- Emissor infravermelho: este componente emite diversos pontos infravermelhos no ambiente, porém não é perceptível a olho nu, pois seu comprimento de onda não pertence ao espectro eletromagnético visível;

- Câmera Red, Green e Blue (RGB) e de profundidade: A câmera de cores possui 640 x 1280 pixels de resolução e a de profundidade 640 x 480 pixels, isto influência diretamente na qualidade da imagem e na quantidade de quadros por segundo que o Kinect consegue processar;

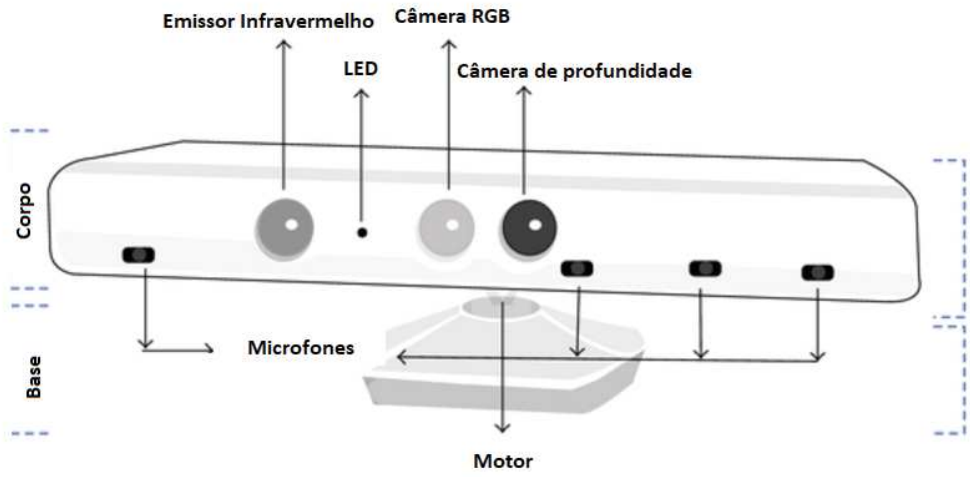


Figura 2.7: Componentes do Kinect v1 [16].

– Eixo motorizado: O Eixo motorizado é uma peça importante no Kinect, através dele é possível alterar o ângulo de elevação do componente onde ficam as câmeras do sensor, alterando assim a visão que o sensor possui do ambiente. Este eixo movimenta o sensor apenas para cima e para baixo. O Kinect possui um raio de visão de 57,5º na horizontal e 43,5º na vertical, contudo este eixo motorizado pode movimentar a visão vertical para 27º para cima ou para baixo, conforme ilustrado na figura 2.8.

– Acelerômetro: Um acelerômetro é um sensor para medir a aceleração sobre um objeto. O Kinect possui um acelerômetro capaz de fazer medidas nas três dimensões: X, Y e Z. Além disso, ele está configurado para suportar um intervalo de 2g, onde g é a força que a gravidade implica sobre o sensor, isso permite que este pequeno sensor consiga informar a sua orientação em relação à gravidade. Este sensor de aceleração possui uma sensibilidade para temperaturas, o valor de seus eixos pode variar em até três graus positiva ou negativamente.

– Conjunto de microfones: O sensor Kinect também conta com quatro microfones e com uma grande capacidade para identificar os sons do ambiente.

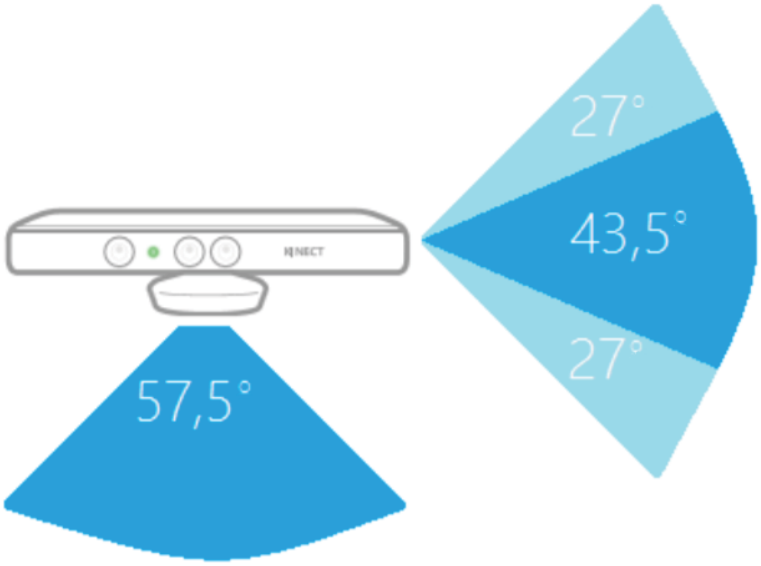


Figura 2.8: Ângulos do eixo motorizado do Kinect v1 [14].

– LED : Um LED é colocado entre a câmera e o projetor de infravermelho. É usado para indicar o status do dispositivo Kinect. A cor verde do LED indica que os drivers do dispositivo foram carregados corretamente. Se você estiver conectando o Kinect a um computador, o LED começará com uma luz verde assim que o sistema detectar o dispositivo. Contudo, para obter a funcionalidade completa do dispositivo, é necessário conectá-lo a uma fonte externa de energia. [16].

Houve mudanças significativas em relação a tecnologia empregada entre o Kinect v2 e o v1. Em sua versão mais recente é possível realizar capturas de imagens RGB com uma resolução de 1920x1080. Além disso ele possui seis microfones embutidos com um melhor sistema de redução de ruído e identificação de voz, mas, ele não possui o acelerômetro que permita identificar a orientação do Kinect, era necessário que isso fosse feito manualmente. Também há uma maior distância disponível para identificar a profundidade, onde agora é possível ver até 4,5 m [5].

Embora “Kinect for Windows” e “Kinect for Xbox” sejam semelhantes em muitos aspectos, existem várias diferenças sutis do ponto de vista de um desenvolvedor. O Kinect

para Windows é principalmente um dispositivo em desenvolvimento e não para fins de jogos como o de Xbox. É possível desenvolver aplicativos que usem o sensor Kinect for Windows ou o Sensor Kinect para Xbox. O sensor Kinect para Xbox foi construído para rastrear jogadores que estão a até 12 pés (4,0 m) de distância do sensor, mas não consegue rastrear objetos que estão muito próximos (80 cm), e talvez seja necessário rastrear objetos a uma distância muito curta para aplicações diferentes. O sensor Kinect para Windows possui um novo firmware, que ativa o rastreamento no modo local. Usando o Modo Próximo, o Kinect para Windows suporta o rastreamento de objetos a uma distância de 40 cm na frente do dispositivo sem perder a precisão. Em termos de alcance, ambos os sensores se comportam da mesma forma [16].

Os sensores Kinect para Windows e Kinect para Xbox precisam de energia adicional para trabalhar com o computador. Isso pode não ser necessário quando conectado ao dispositivo Xbox, pois o Xbox tem energia suficiente para operar o dispositivo. Não há diferença entre o Xbox Kinect e o Kinect for Windows a esse respeito. Contudo No Kinect para Windows, o cabo USB é pequeno e aprimorado para permitir mais confiabilidade e portabilidade em uma ampla variedade de computadores. E, finalmente, o sensor Kinect para Windows é para aplicações comerciais, que significa que, se você estiver desenvolvendo um aplicativo comercial, deverá usar o Kinect para dispositivo Windows para produção, enquanto você pode usar o Kinect for Xbox para fins de desenvolvimento, aprendizado e pesquisa [16].

Com base nesses dados e com o equipamento disponível em laboratório foi escolhida a versão do Xbox 360 para ser utilizado no desenvolvimento da dissertação.

2.3 Redes Neuronais Artificiais

Ao longo dos anos, o ser humano buscou aprimorar suas tecnologias em diversas áreas, descobrindo e criando novos dispositivos.

Em especifico, na área de inteligência artificial, sempre houve um grande interesse no estudo do cérebro humano e seu funcionamento. A fim de simular seu comportamento

e se aproximar do que um neurônio é capaz de fazer, a área de estudo de redes neurais artificiais foi criada [17].

A primeira referência dessa área se deu pelos pesquisadores Warren McCulloch e Walter Pitts que por sua vez inspirou outros cientistas a se interessarem pelo assunto, como Frank Rossenblatt que desenvolveu os perceptrons, neurónios artificiais que simulam neurónios reais.

Um perceptron recebe várias entradas binárias x_j, e produz uma única saída binária u [18], como demonstrado na figura 2.9. Para calcular a saída, Rossenblatt propôs introduzir pesos p_j em suas entradas, de forma que representem sua importância. A saída do neurônio 0 ou 1, é determinada se a soma ponderada $\sum_j x_j.p_j$ é menor ou maior que algum valor limite, dessa forma chegamos a expressão:

$$u = \begin{cases} 0, & \text{se } \sum_j x_j.p_j \leq limiar \\ 1, & \text{se } \sum_j x_j.p_j > limiar \end{cases} \tag{2.1}$$

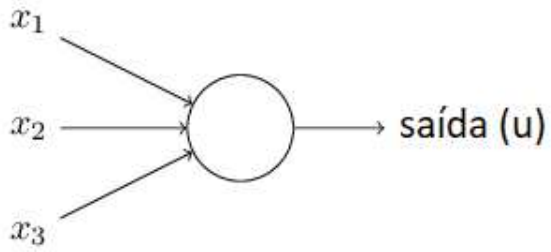


Figura 2.9: Perceptron [18].

Se a única saída binária u_{kj} for introduzida em outros perceptrons, de forma que a camada k anterior esteja toda ligada como demonstrado na figura 2.10 , é criado uma rede de perceptrons.

Dessa forma, a cada camada, um perceptron pode tomar decisões mais complexas baseado na camada anterior, tomando decisões mais precisas e confiáveis [18].

Ou seja, a rede neural artificial é um método da inteligência artificial capaz de gerar conhecimento e determinar o comportamento de um conjunto de dados através de

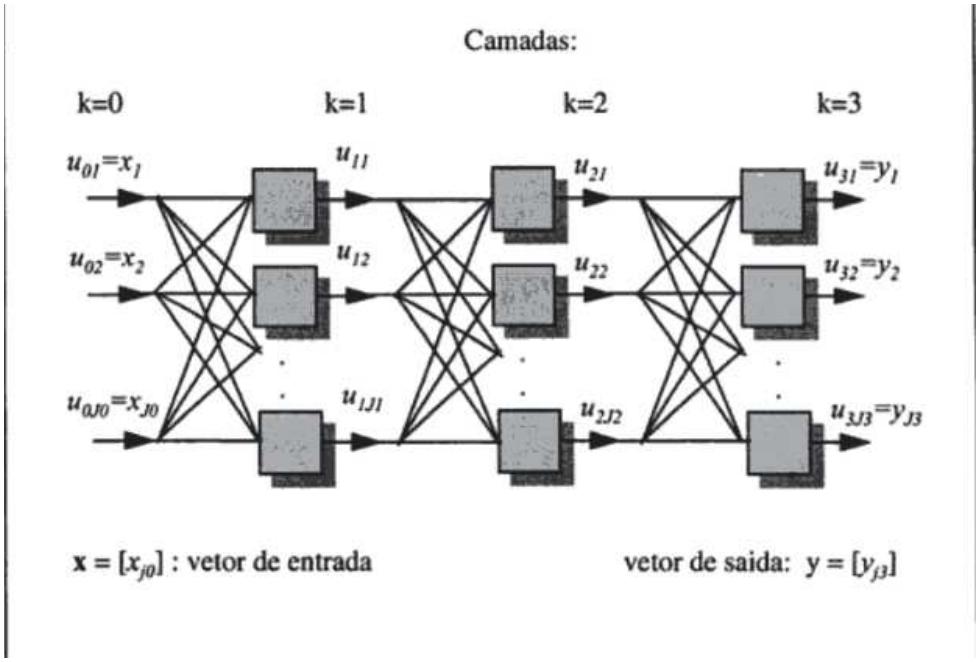


Figura 2.10: Rede de Perceptrons [19].

treinamentos e simulações [20].

2.3.1 Redes Neuronais Convolucionais

Uma Rede Neural Convolucional (ou Convolutional Neural Network (CNN)), se originou da rede de perceptrons de múltiplas camadas. De forma semelhante aos processos tradicionais de visão computacional, uma CNN é capaz de aplicar filtros em dados visuais, mantendo a relação de vizinhança entre os pixels da imagem ao longo do processamento da rede, sendo muito utilizado em classificação de imagem e vídeo [21].

As CNN são arquiteturas multi estágios capazes de serem treinadas. Os campos receptivos são altamente correlacionados à localidade do estímulo na imagem capturada. As CNN utilizam este conceito forçando um padrão de conectividade entre as camadas de neurônios artificiais [22].

Um mapa de característica é obtido efetuando a convolução de uma imagem de entrada por um filtro linear seguido da adição de um termo de viés e da aplicação de uma função não linear. Sendo a camada k, os filtros determinados por um conjunto de pesos W^k e

um termo de viés b_k e o operador de convolução *, a Equação 2.2 mostra a obtenção do mapa de característica h^k para uma função não linear f [23].

$$(h^k)_{ij} = f((W^k * x)_{ij} + b_k) \tag{2.2}$$

Cada estágio é composto por três etapas, filtragem (filter bank layer), etapa não linear (non-linearity layer) e etapa de redução (feature pooling layer) que representa o campo receptivo. Uma CNN pode ser composta de um ou mais estágios onde cada um contém as três etapas. A Figura 2.11 mostra uma CNN com um único mapa de características de entrada (ex: uma imagem em tons de cinza) com dois estágios convolucionais C1+S1 e C2+S2 como mostra a figura 2.11.

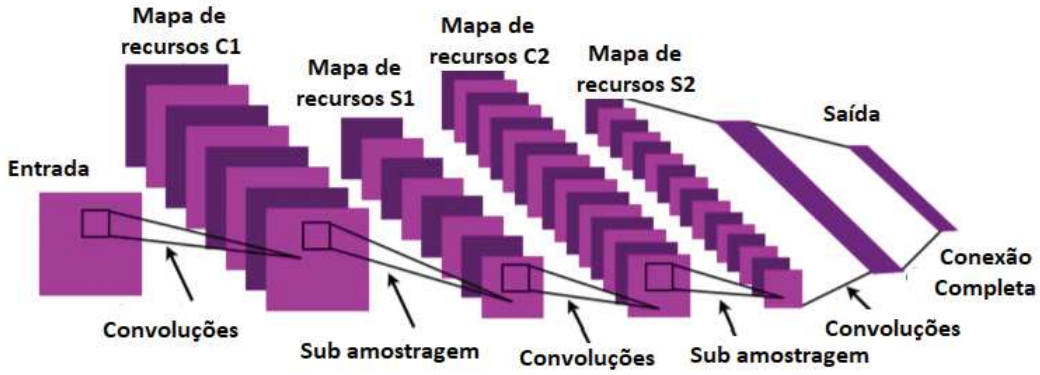


Figura 2.11: Uma CNN com dois estágios convolucionais [23].

Uma das formas de efetuar o treinamento da rede, é utilizando o algoritmo de gradiente descendente estocástico ou stochastic gradient descent Stochastic Gradient Descent (SGD) para minimizar a diferença entre a saída da CNN e a saída desejada. O algoritmo SGD procura encontrar o valor mínimo de uma função utilizando um processo iterativo com base no gradiente. O gradiente de uma função é definido por um vetor de derivadas parciais. Como o gradiente sempre aponta para o valor máximo, e o objetivo é encontrar o valor mínimo da função, atualizam-se os valores com base no valor negativo do gradiente. Esta estratégia eventualmente levará ao valor mínimo global de uma função convexa [22].

O algoritmo abaixo demonstra as etapas do SGD.

1. Inicialização: Escolha um ponto X_0 como tentativa inicial

2. Gradiente: Calcule o gradiente da estimativa $V_I = \bigtriangledown f(x_i)$ onde $\bigtriangledown f(x_i)$ corresponde às derivadas parciais.

3. Atualização: Atualizar os valores utilizando $x_{i+1} = x_i - \alpha v_i$. Onde α é um parâmetro de taxa de aprendizado (ou learning rate) definido manualmente para acelerar a atualização dos novos valores a cada iteração.

4. Iteração: Repetir as etapas 2 e 3 até a função chegar próxima a zero [22].

Capítulo 3

Materiais e Métodos

As duas versões do Kinect, v2 e v1 podem ser utilizados em diferentes plataformas e sistemas operacionais, como Windows, GNU/Linux e Mac OS X, além de ter funcionalidades e pacotes para inúmeros softwares, entre eles estão o Windows Software Development Kit (SDK), Visual Studio e Matllab. Porém, o Kinect de Xbox 360 tem problemas de compatibilidade com esses softwares, impedindo de extrair todo o potencial do dispositivo. Após testar os softwares citados, o que teve a melhor funcionalidade e que foi possível fazer capturas de imagens foi o Matlab. Com base nisso, ele foi escolhido para ser utilizado como ferramenta de software para o sistema.

A montagem do sistema é simples, ele é composto por apenas dois componentes, o kinect, e o computador, como mostrado na figura 3.1.

3.1 Matlab

O MatLab é uma poderosa ferramenta matemática e Integrated Development Environment (IDE) de desenvolvimento. Seu processamento é composto em suma por equacionamentos matriciais.

Sua interface lhe permite trabalhar com programação de alto nível por linguagem de blocos através do Simulink e, lhe permite também, trabalhar com uma linguagem derivada do C/C++ em suas interfaces de código.

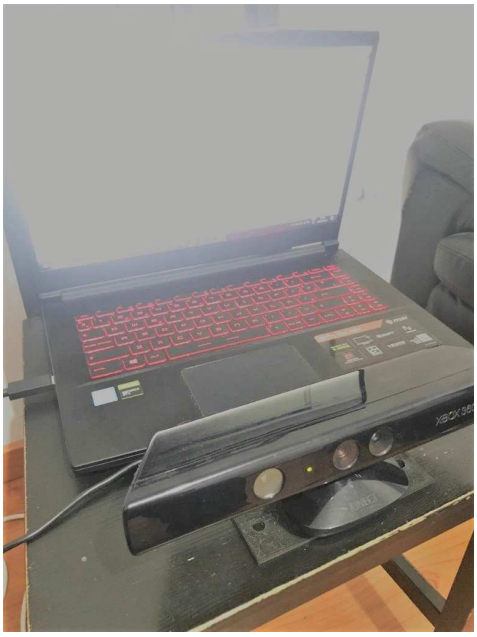


Figura 3.1: Componentes do sistema.

Sua sintaxe é um pouco diferenciada, sua flexibilidade e sua capacidade de expansão, utilizando as conhecidas ToolBoxes, permite ao MatLab se tornar especialista em diversas áreas de conhecimento, desde análise de elementos finitos, inteligência artificial e análise de sinais até a depuração de processamento em tempo real [15].

Para que o Kinect se torne funcional no MatLab, é necessário instalar o pacote Microsoft Kinect for Windows Support From Image Acquisition Toolbox.

Devido a falta de compatibilidade do Kinect para Xbox 360 com os softwares mais recentes, foi utilizado a versão de 2015 do Matlab para realizar a captura das imagens, e a versão de 2019 para realizar o treinamento da rede convolucional e classificação dos gestos.

3.2 Método de Captura

Afim de realizar a aquisição das imagens de gestos, foi utilizado o esquema apresentado na Figura 3.2.

Primeiramente, é necessário a inicialização do Kinect, após isso o programa irá emitir

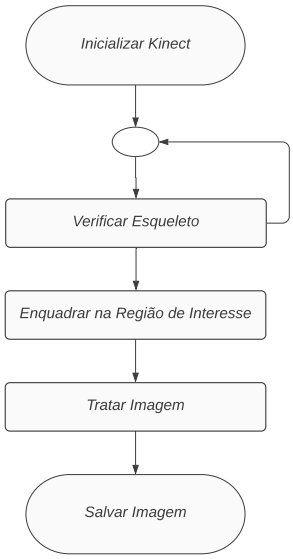


Figura 3.2: Fluxograma da aquisição das imagens.

luzes infrafermelhas no ambiente. Dessa forma, o seu software consegue informações de presença e distância de objetos, pois ele possui um diferencial que consegue identificar o corpo humano através de coordenadas cartesianas de 20 articulações como mostra a tabela 3.1 e na imagem 3.3.

E para que isso ocorra de forma otimizada e com mínimo de erros, o usuário deve se posicionar (como mostra a imagem 3.4) de forma que a área destacada seja a localização ideal.

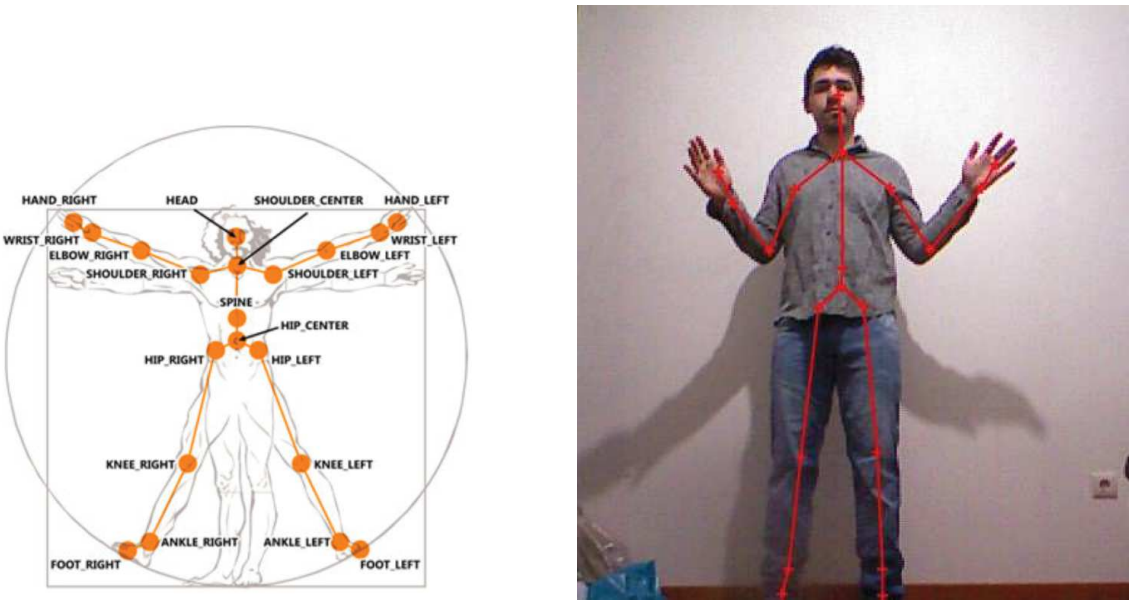
Assim que encontrado o corpo humano, ele automaticamente irá focalizar na mão direita para obtenção do gesto.

Para localizar a região das mãos, independentemente do local onde ela se encontra, desde que esteja a vista, é utilizada a função de localização das articulações do Kinect, como mostrado na figura 3.3, através do índice 12. Assim, é obtido as coordenadas da mão direita e focalizamos a região onde ela se encontra e é realizada a captura do gesto, como mostrado na figura 3.5.

Após isso ocorre o tratamento da imagem. Inicialmente a imagem é normalizada e

Tabela 3.1: Valores das juntas retornadas pelo Kinect [24].

Membro	**Valor**	**Descrição**
Hip_Center	01	Centro do Quadril
Spine	02	Coluna
Shoulder_Center	03	Centro do Ombro
Head	04	Cabeça
Shoulder_Left	05	Ombro Esquerdo
Elbow_Left	06	Cotovelo Esquerdo
Wrist_Left	07	Pulso Esquerdo
Hand_Left	08	Mão Esquerda
Shoulder_Right	09	Ombro Direito
Elbow_Right	10	Cotovelo Direito
Wrist_Right	11	Punho direito
Hand_Right	12	Mão Direita
Hip_Left	13	Quadril Esquerdo
Knee_Left	14	Joelho Esquerdo
Ankle_Left	15	Tornozelo Esquerdo
Foot_Left	16	Pé Esquerdo
Hip_Right	17	Quadril Direito
Knee_Right	18	Joelho direito
Ankle_Right	19	Tornozelo Direito
Foot_Right	20	Pé direito



(a) Teórico [25] (b) Real

Figura 3.3: Juntas Captadas pelo Kinect

transformada em uma escala de cinza, para que seja realizada a transformação em preto e branco utilizando o valor médio da imagem como limiar para binarização. E por fim a imagem é salva em uma base de dados.

3.3 Base de Dados

O processo de captura das imagens descrito anteriormente permanece em um loop, até que seja realizado a captura da quantidade de imagens que o usuário definiu.

Estas imagens são salvas em uma pasta atribuída ao gesto. Todas as imagens capturadas de mão aberta serão salvas na primeira pasta e as de mão fechada na segunda, e ambas devem permanecer juntas em uma única pasta superior, pois facilita na criação de um banco de dados de imagens.

É necessário observar que deve se realizar a obtenção das imagens de um gesto de cada vez, pois cada gesto irá para sua respectiva pasta.

Figura 3.4: Limites do Kinect [26].

Figura 3.5: Mão Focalizada

3.4 Alexnet

AlexNet é uma rede neural convolucional com 8 camadas de profundidade. É possivel carregar uma versão pré-treinada da rede em mais de um milhão de imagens do banco de dados ImageNet. A rede pré-treinada pode classificar imagens em 1000 categorias de objetos, como teclado, mouse, lápis e muitos animais. Dessa forma, a rede aprendeu representações ricas de recursos para uma ampla variedade de imagens. A rede tem um tamanho de entrada de imagem de 227x227 pixels. [27].

O Alexnet possui uma arquitetura que consiste em 8 camadas de peso, incluindo 5 camadas convolucionais e 3 camadas totalmente conectadas, e três camadas de pool máximo são usadas após a primeira, segunda e quinta camadas convolucionais. A primeira camada convolucional possui 96 filtros de tamanho 11 x 11 com um passo de 4 pixels e preenchimento com 2 pixels. O passo e o preenchimento de outras camadas convolucionais são definidas como 1 pixel. A segunda camada convolucional possui 256 filtros de tamanho 5 x 5. A terceira, quarta e quinta camadas convolucionais têm 384, 384 e 256 filtros com tamanho de 3 x 3, respectivamente [28].

Também é possível utilizar uma rede de classificação de imagens pré treinadas que tem informações de como extrair recurso e informações de imagens naturais e utilizá-la como ponto de partida para aprender uma nova forma de reconhecimento, esta técnica é chamada de transfer learning. Dessa forma, utilizar uma rede pré treinada com transferência de aprendizado é muito mais rápido e fácil do que treinar uma rede desde o início [29].

3.5 Método de Treinamento

Foi utilizado o esquema conforme a figura 3.6 para fim de compreensão de como o treinamento é realizado.

Para realizar o treinamento, é necessário utilizar o Matlab 2016 ou acima, pois o pacote do Alexnet não é compatível com as versões mais antigas, dessa forma foi utilizado o

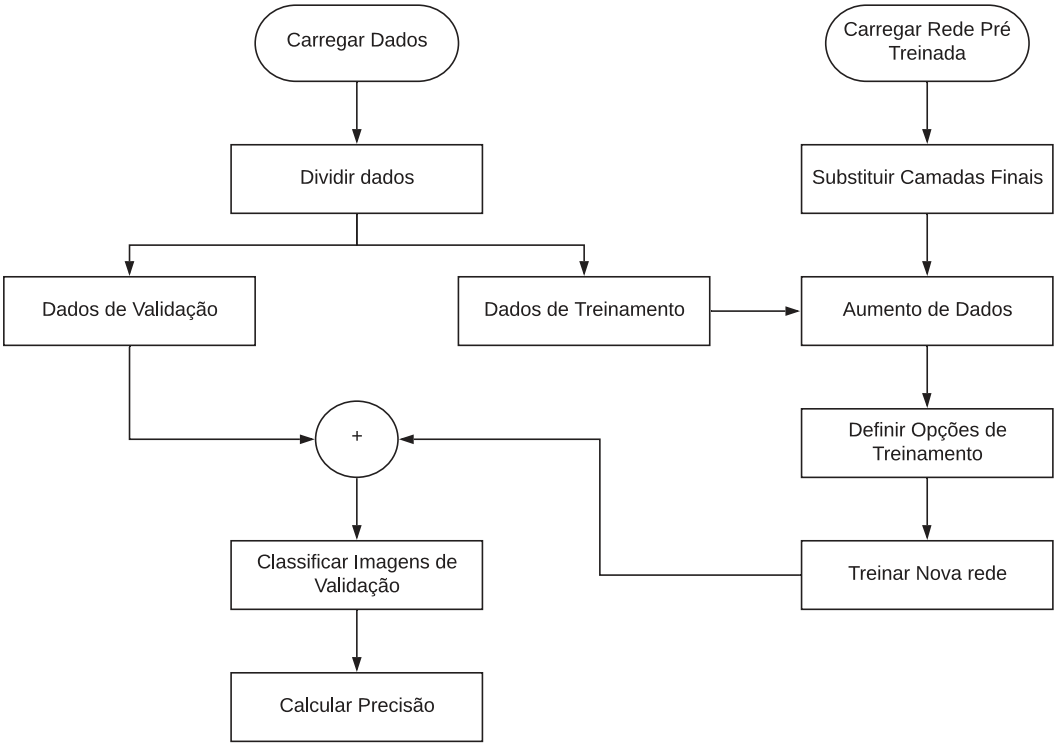


Figura 3.6: Fluxograma do Treinamento das Imagens.

Matlab 2019 como descrito na seção 3.1. A partir disso, seguindo o esquema do diagrama, inicialmente o programa carrega as imagens dos gestos. Nesse ponto, é importante que cada gesto esteja salvo em sua respectiva pasta, e todos estejam unidos em uma pasta superior, pois é utilizado a função imageDatastore, criando um banco de dados de imagens com todos os gestos. Observando que cada pasta de gesto se tornará uma label da rede treinada posteriormente.

Após isso, os dados são divididos randomicamente, sendo que 70% das imagens é utilizada para o treinamento da rede, e os outros 30% servirão para validar o treinamento em relação a sua taxa de acerto.

A seguir, a rede pré treinada Alexnet é carregada e as suas últimas três camadas são substituídas por uma camada totalmente conectada, uma camada softmax e uma camada de saída de classificação. Levando em consideração que essa camada conectada deve ter o mesmo tamanho que o número de classes nos novos dados.

A rede requer imagens de entrada de tamanho 227x227x3, mas as imagens nos armazenamentos de dados de imagem têm tamanhos diferentes. Dessa forma é utilizado um armazenamento de dado aumentado para redimensionar automaticamente as imagens de treinamento. As imagens de treinamento são invertidas aleatoriamente ao longo do eixo vertical e são convertidas em até 30 pixels horizontal e verticalmente. Este aumento de dados ajuda a impedir que a rede se ajuste demais e memorize os detalhes exatos das imagens de treinamento [30].

Assim, tem-se um ponto muito importante para o treinamento, onde é definido o tipo de treinamento utilizado através do argumento de entrada, a taxa de aprendizado e o número de épocas de interação. O AlexNet permite que sejam inseridos tres tipos de argumentos de entrada, o Stochastic Gradient Descent with Momentum (SGDM), Root Mean Square Propogation (RMSprop) e o Adaptive Moment Optimization (Adam), e todos foram testados para obter o melhor desempenho do sistema. Já os outros fatores citados a cima foram variados algumas vezes de forma empírica.

Após todos os procedimentos descritos anteriormente, inicia-se o treinamento da rede. Durante o treinamento é possível acompanhar o nível de precisão e a curva de perda

conforme o tempo.

Após o término do treinamento, é possível identificar gestos com base nos que foram treinados, possibilitando realizar ações com gestos específicos e decifrar a LGP.

3.6 Treinamentos Realizados

Para verificar se o sistema realmente funciona, os primeiros treinamentos foram realizados com 7 gestos: mão aberta(1), mão fechada(2), polegar(3), um dedo(4), dois dedos(5), três dedos(6) e quatro dedos(7), como mostra a figura 3.7.

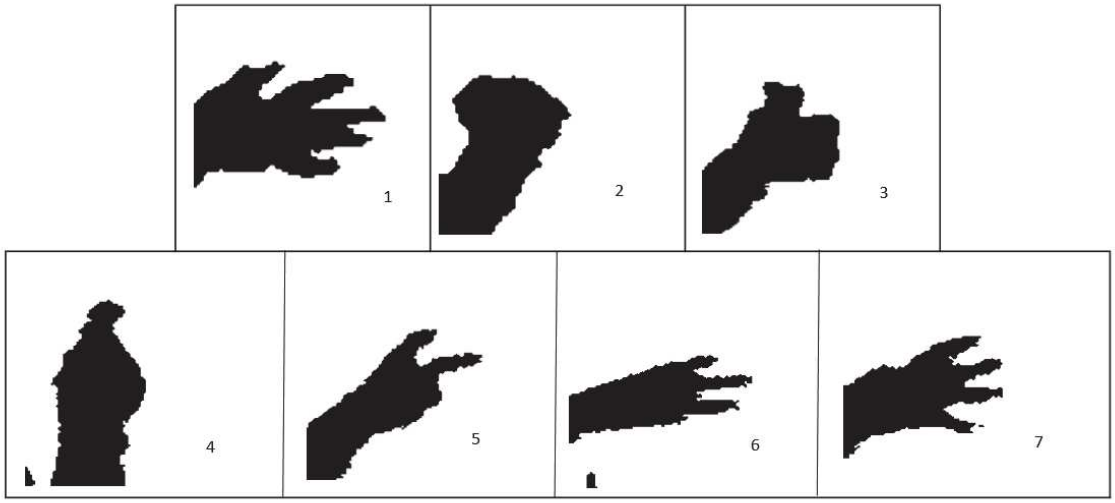


Figura 3.7: 7 gestos da mão.

Estes treinamentos foram realizados em 3 situações diferentes para se aprofundar nos parâmetros de treino ideal para a aplicação estudada.

– Na primeira situação, utilizamos um computador de processador Intel® Core(TM) i7-9750H CPU @ 2.60 GHz, 2601 Mhz, 6 Núcleos e 12 Processadores Lógicos e 16GB de memória RAM. Foram utilizadas 1000 imagens para cada gesto, de forma em que o gesto estava em diferentes regiões do espaço, do lado esquerdo, direito e a frente do corpo, assim como mais elevada e mais abaixo como mostra a figura 3.8.

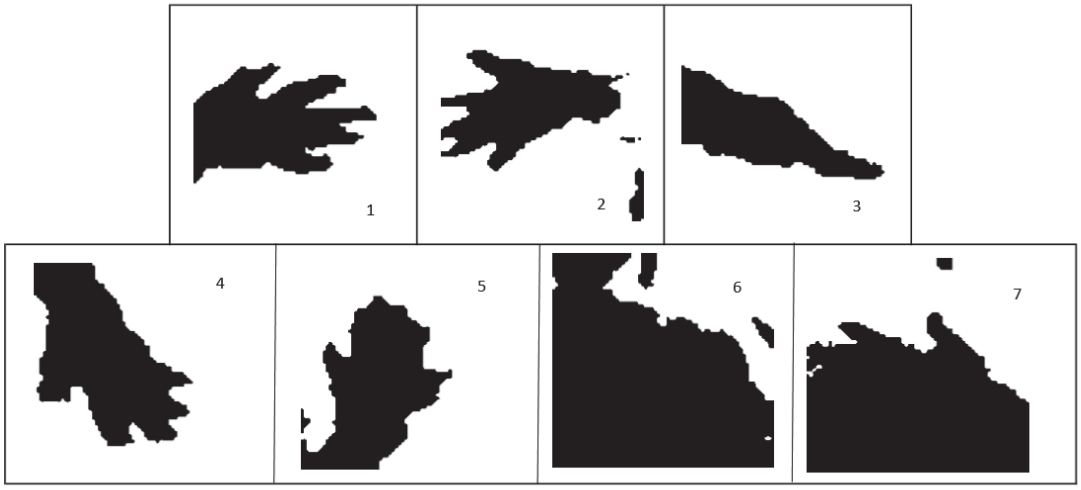


Figura 3.8: Mão aberta em diferentes ângulos.

Neste treinamento foi utilizado um número de 4 épocas, variável que altera o numero total de interações durante o treinamento , com uma taxa inicial de aprendizado(em inglês learning rate) de $1e^{-4}$, ou seja, esse parâmetro é utilizado para definir o tamanho do passo que será dado na direção apontada pelo gradiente.

– Na segunda situação, foi utilizado os mesmos parâmetros do teste anterior, porém foi alterada o número de épocas 4 para 10.

– Na terceira situação, foi utilizado o número de épocas em 10. Porém, foi alterado o computador utilizado, a fim de verificar o desempenho do sistema. O computador utilizado neste teste possui um processador Intel® Core™ i5-4210U CPU @ 1.70GH, 1701 Mhz, 2 Núcleos, 4 processadores Lógicos e 4GB de memória RAM.

Alem disso, todas estas situações foram realizadas nos três tipos de argumentos de entrada, em SGDM, RMSprop e Adam.

Com base nos resultados dos treinamentos anteriores que serão apresentados posteriormente no capítulo 4, foi comprovado que o sistema funciona, dessa forma foi feito a captura dos 26 gestos do alfabeto LGP, e para cada gesto foram salvas 1000 imagens. Diferentes dos testes anteriores os gestos foram realizados apenas do lado direito do corpo,

em diferentes alturas, mas não em frente ao corpo, pois graças aos testes anteriores foi possível observar que as imagens perdiam muita qualidade quando o gesto estava à frente do corpo, como é visto no item 6 e 7 da figura 3.8, tornando impossível de identificar os gestos. Para esse teste também foi utilizado o primeiro computador de processador Intel® Core(TM) i7 pelo seu maior desempenho em relação ao tempo. Segue o exemplo de algumas letras na figura 3.9:

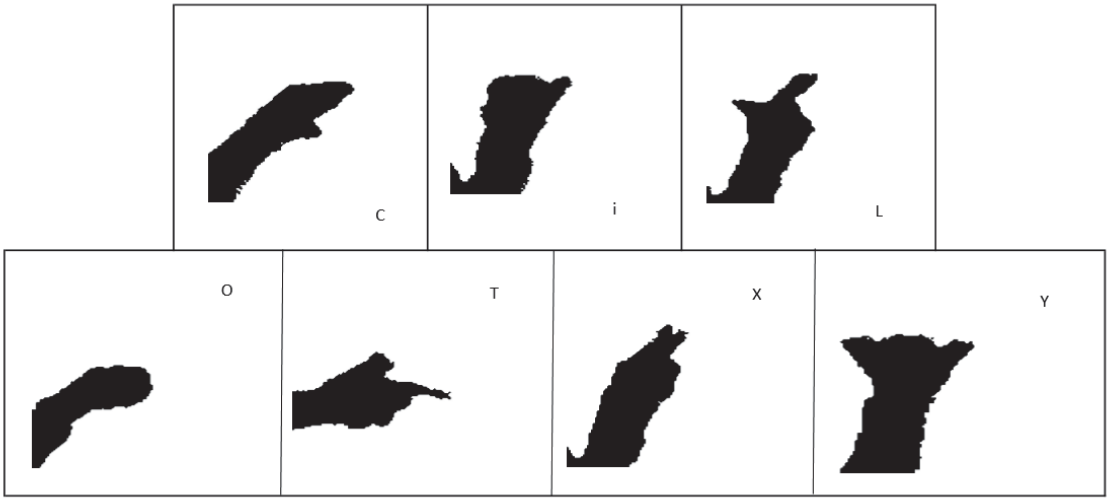


Figura 3.9: Letras C, I, L, O, T, X e Y do alfabeto LGP.

Capítulo 4

Resultados e Discussões

Nesta seção, será discutido os resultados dos testes realizados no capitulo 3, comparando os diferentes métodos da CNN, assim como viabilidade do sistema.

4.1 SGDM

Esta seção aborda apenas os treinamentos realizados com o SGDM como argumento de entrada. Sendo assim,no primeiro treinamento de rede com os 7 gestos diferentes, a época de interações foi fixada em 4 e foi utilizado o computador de melhor desempenho, foi obtido as seguintes curvas de precisão e perda como mostrado na figura 4.1.

Durante o treinamento ocorreu um total de 1960 interações, obteve 89.29% de precisão e demorou 55 minutos e 28 segundos para ser realizada.

O segundo treinamento realizado também com 7 gestos, porém, com 10 épocas de interação, gerou o seguinte gráfico 4.2.

Este treinamento obteve um total de 4900 interações, com 94.10% de precisão e demorou 137 minutos e 31 segundos para ser realizado.

Ao observar os dois primeiros treinamentos, é possível perceber que graças ao aumento do número de épocas houve um ligeiro aumento de 4.81% de precisão do segundo para o primeiro, porém ficou duas vezes e meio mais lento. Isso se deve ao fato de que ao aumentar o número de épocas, também aumenta o número de interações durante o treinamento.

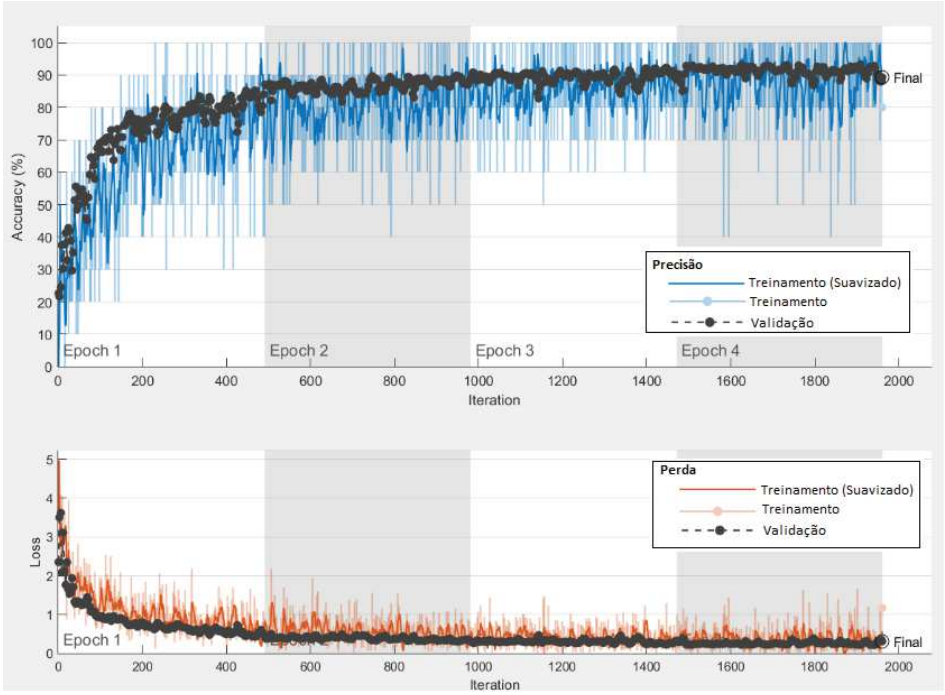


Figura 4.1: Gráfico de Precisão e Perda do treinamento com 4 épocas de interação (SGDM).

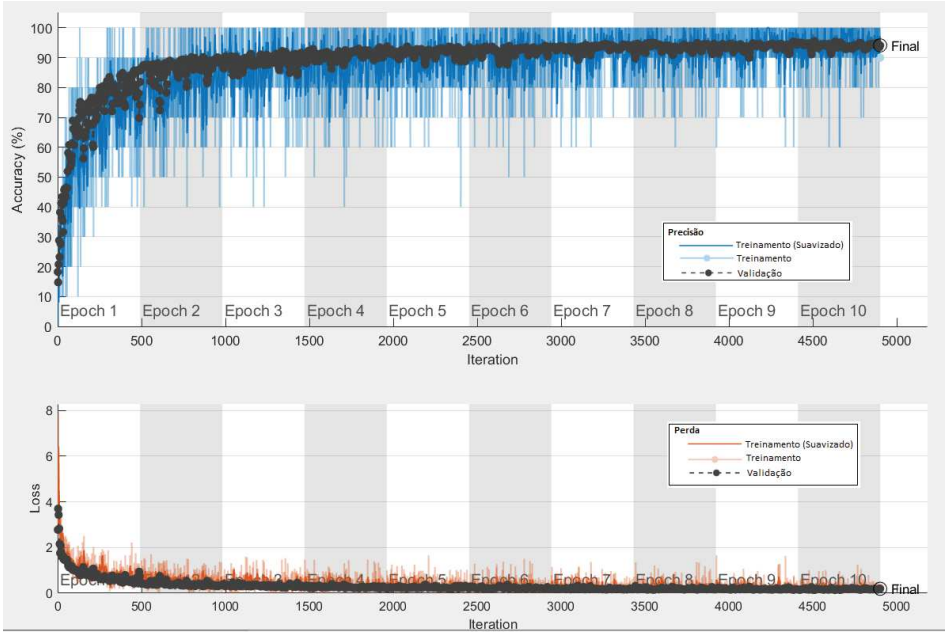


Figura 4.2: Gráfico de Precisão e Perda do treinamento com 10 épocas de interação (SGDM).

Esses resultados indicam que é possível aumentar a precisão do seu sistema apenas aumentando o número de interações. Porem, se a aplicação desejada não requer muita precisão, é possível deixar o treinamento mais rápido e ainda manterá um bom nível de precisão, vendo que a partir da segunda época, em ambos os casos, a curva se mantêm próximo dos 90%.

O terceiro treinamento se manteve nos 10 ciclos de interação, porém, foi realizado em um computador diferente dos anteriores, segue os gráficos 4.3 com os resultados.

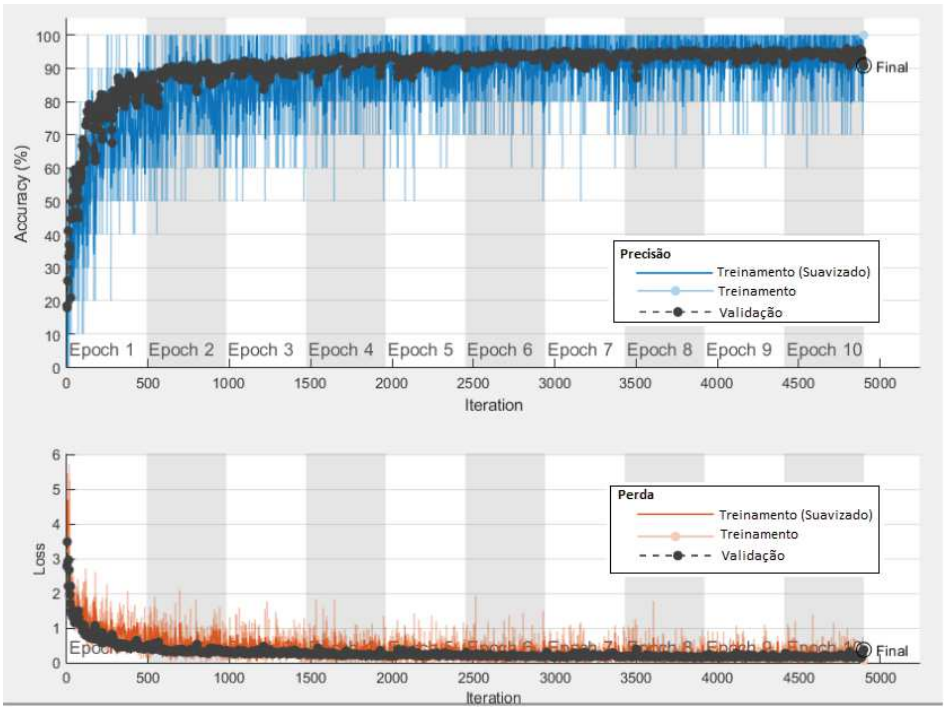


Figura 4.3: Gráfico de Precisão e Perda do treinamento com 10 épocas de interação em um computador de menor desempenho (SGDM).

Este treinamento se manteve nas 4900 intereções, e obteve 90.81% de precisão, mas demorou 34 horas, 51 minutos e 30 segundos para ser finalizado.

O segundo e o terceiro treinamento, possuem configurações idênticas, porém, a máquina em que se aplica o treinamento é diferente. É visto claramente que o computador de menor tecnologia em hardware, possui um desempenho muito lento comparado ao superior, cerca de 15 vezes mais lento e gerou uma precisão de 3.29% menor.Isto torna inviável

realizar o treinamento em máquinas de baixo desempenho, por questão de tempo.

Teoricamente a precisão nos dois casos deveriam ser semelhantes por utilizarem o mesmo método de treinamento, entretanto, para realizar o treinamento foi dividido aleatoriamente 70% dos dados para treino e 30% para validação. Esta aleatoriedade pode gerar pequenas variações de resultados com dados equivalentes.

Vale salientar que apenas a função de treinamento requer um computador que tenha um bom desempenho. Pois assim que sua rede estiver treinada, basta salvá-la e carregá-la onde precisar. E com apenas um comando é possível identificar um gesto.

4.2 RMSprop

Esta seção aborda apenas os treinamentos realizados com o RMSprop como argumento de entrada. Dessa forma, foi utilizado as mesmas condições da seção anterior, um treinamento de rede com 7 gestos diferentes, a época de interações foi fixada em 4 e foi utilizado o computador de melhor desempenho. Foi obtido as seguintes curvas de precisão e perda como mostrado na figura 4.4.

Durante o treinamento ocorreu um total de 1960 interações, obteve 87.76% de precisão e demorou 53 minutos e 58 segundos para ser realizada.

O segundo treinamento realizado também com 7 gestos, porém, com 10 épocas de interação, gerou o seguinte gráfico 4.5.

Este treinamento obteve um total de 4900 interações, com 86.10% de precisão e demorou 135 minutos e 15 segundos para ser realizado.

Ao comparar os gráficos 4.4 e 4.5 com os 4.1 e 4.2 do treinamento com SGDM, é possível observar uma maior descontinuidade nas curvas de validação e treinamento, afetando diretamente em sua precisão final, já que em ambos os casos, a precisão do treinamento realizado com o RMSprop foi inferior ao SGDM.

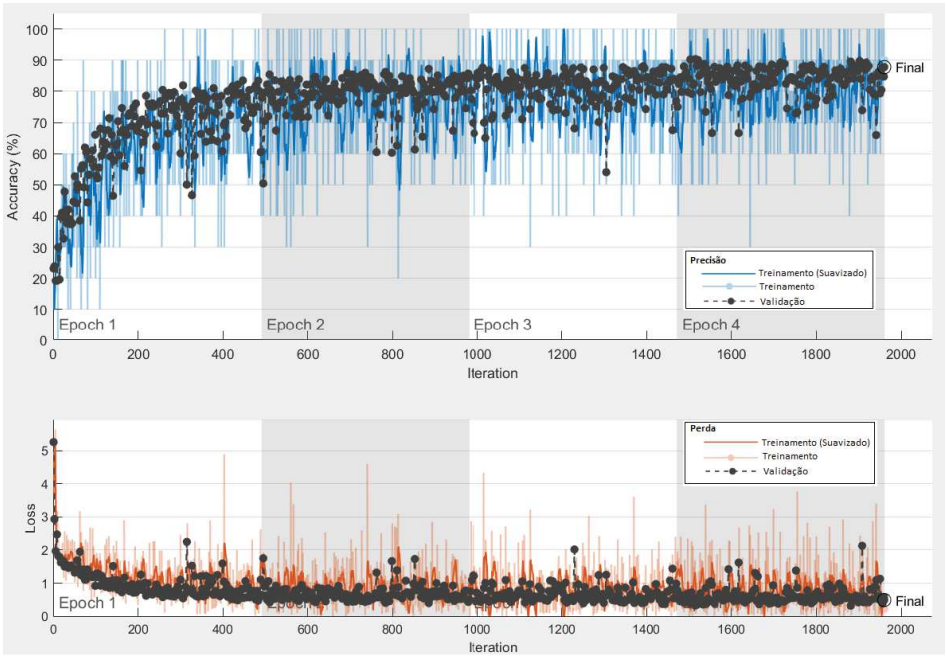


Figura 4.4: Gráfico de Precisão e Perda do treinamento com 4 épocas de interação (RMSprop)

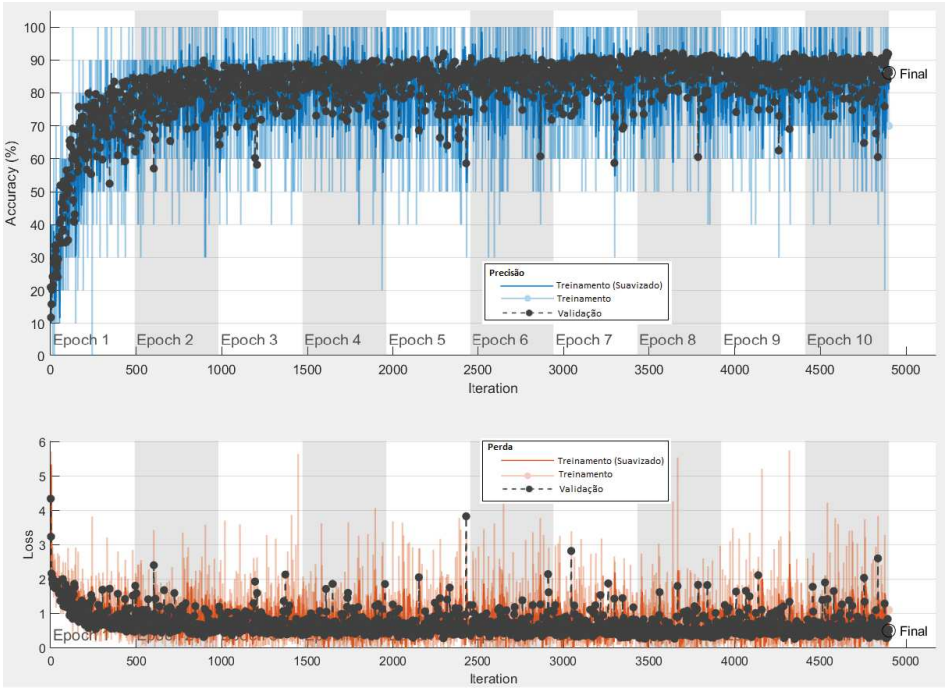


Figura 4.5: Gráfico de Precisão e Perda do treinamento com 10 épocas de interação (RMSprop)

4.3 Adam

Esta seção aborda apenas os treinamentos realizados com o Adam como argumento de entrada. Dessa forma, foi utilizado as mesmas condições das duas seções anteriores, um treinamento de rede com 7 gestos diferentes, a época de interações foi fixada em 4 e foi utilizado o computador de melhor desempenho. Foi obtido as seguintes curvas de precisão e perda como mostrado na figura 4.6.

Figura 4.6: Gráfico de Precisão e Perda do treinamento com 4 épocas de interação (Adam)

Durante o treinamento ocorreu um total de 1960 interações, obteve 86.95% de precisão e demorou 51 minutos e 39 segundos para ser realizada.

O segundo treinamento realizado também com 7 gestos, porém, com 10 épocas de interação, gerou o seguinte gráfico 4.7.

Este treinamento obteve um total de 4900 interações, com 94.00% de precisão e demorou 128 minutos e 56 segundos para ser realizado.

Ao comparar os métodos SGDM e Adam, é possível observar que ambos possuem

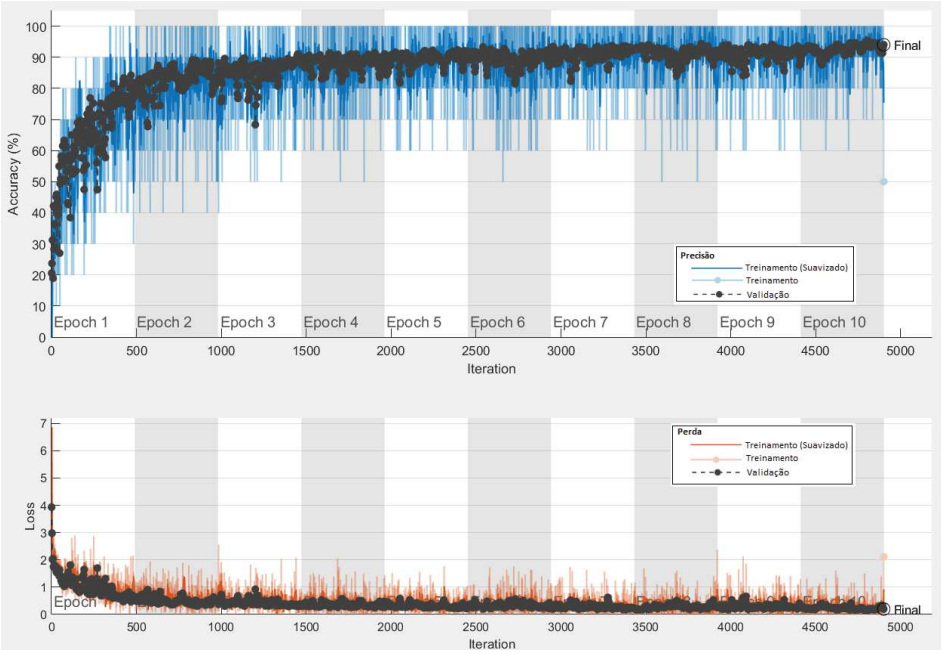


Figura 4.7: Gráfico de Precisão e Perda do treinamento com 10 épocas de interação (Adam)

resultados muito semelhantes para esta aplicação. Porem, analisando os gráficos 4.6 e 4.1, nota-se que com quatro ciclos de interação os pontos de validação do argumento Adam são levemente mais descontínuos comparados ao SGDM, provando que mesmo com poucas épocas de treinamento, o método mais preciso para o problema proposto é o SGDM, pois é um método que ajuda a acelerar vetores de gradientes nas direções corretas, levando a uma convergência mais rápida. É um dos algoritmos de otimização mais populares e muitos modelos de ponta são treinados usando-o [31].

4.4 Alfabeto LGP

A fim de fazer a tradução do alfabeto LGP, foi realizado também o treinamento dos 26 gestos do alfabeto. Este treinamento foi realizado com o método SGDM, 10 épocas de interação e com o computador de melhor desempenho.

Mesmo tendo concluído anteriormente que não era necessário tantos ciclos de interação, foi utilizado um valor considerável, pois o interesse é que o sistema tenha a maior precisão

possível, sendo apresentado na Figura 4.8 o resultado do treinamento.

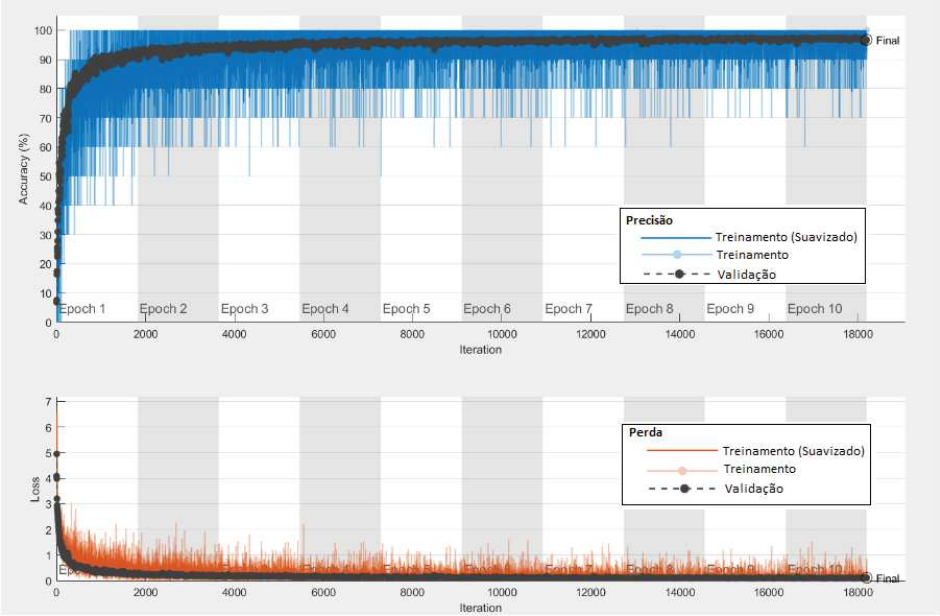


Figura 4.8: Gráfico de Precisão e Perda do treinamento do alfabeto LGP.

Este treinamento requisitou de um total de 18200 interações e resultou em uma precisão de 96.71% com um tempo de 1689 minutos e 35 segundos.

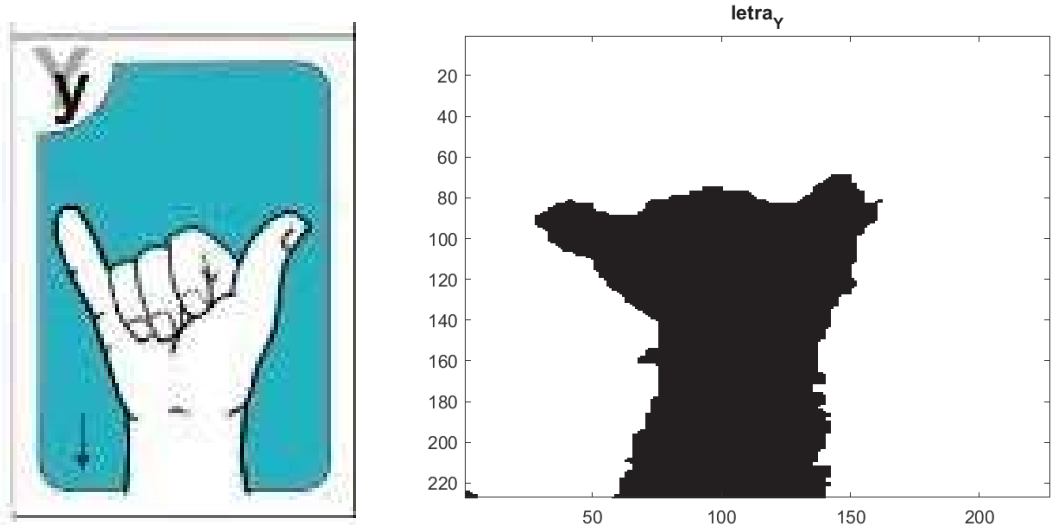
4.5 Classificação

Após ter realizado o treinamento de uma rede, a rede se tornará uma variável de formato ".mat". Dessa forma, basta salva-lá e será possível utiliza-lá em qualquer máquina que possua um matlab 2016 ou superior.

Devido a falta de compatibilidade do Kinect para Xbox 360 com as versões mais recentes do matlab, não foi possível realizar a tradução do gesto em tempo real. Porém, se os gestos que deseja traduzir forem gravados anteriormente, basta carregar as imagens no matlab 2019 e chamar o comando classify, como segue o exemplo no código 4.9 com uma imagem da letra Y na Figura 4.10. Esta classificação pode ocorrer entre 4 e 10 ms.

```
load netTransferAlfabeto
net = netTransferAlfabeto;
A=imread('img_202032017425037.png'); %Imagem com a letra Y
label = classify(net, A);        % Classificação da imagem
image(A);     % Mostra a imagem
title(char(label)); % Insere o titulo com a respectiva letra
```

Figura 4.9: Exemplo de código para classificar uma letra do alfabeto LGP.



(a) Teórico [3] (b) Classificada

Figura 4.10: Letra Y do alfabeto LGP.

Capítulo 5

Conclusões

Com a ascenção da tecnologia em inteligência artificial, este trabalho, teve como objetivo, proporcionar uma nova ferramenta para a comunidade LGP, facilitando a comunicação entre pessoas interessadas no assunto, e até mesmo no treinamento para as que tem o interesse de aprender.

Através dos testes realizados, foi observado que o dispositivo possui uma alta precisão de acerto em relação aos gestos treinados, tornando o sistema utilizável na prática.

Infelizmente, devido a falta de compatibilidade dos equipamentos disponíveis com a tecnologia mais recente, não foi possível realizar testes em tempo real, onde a tradução seria extremamente rápida, mas é possível verificar que existe a possibilidade para tal.

Vale salientar que este trabalho não serve apenas para a tradução da linguagem LGP, pois com a identificação de um gesto especifico é possível gerar comandos para qualquer ação que se deseje realizar, como controlar um robô, mandar instruções para outro dispositivo ou utilizar em tarefas onde não seja possível o contacto dos dedos com o teclado ou com um ecrã táctil.

Tendo em vista que este sistema teria grande utilidade se fosse possível realizar traduções em tempo real, para trabalhos futuros, seria essencial a utilização do Kinect v1 para Windows ou a versão v2 que possui um melhor desempenho.

Além disso, o sistema detecta apenas o alfabeto de forma estática, dessa forma, acrescentar palavras e gestos dinâmicos também seria um ótimo incremento ao sistema.

Bibliografia

[1] H. M. M. Ferreira, *A Língua Gestual: Uma forma de Comunicação*, https://www.porsinal.pt/index.php?ps=artigos&idt=artc&cat=7&idart=46. (acedido em 13/05/2020).

[2] *Constituição da República Portuguesa*, https://www.parlamento.pt/Legislacao/Paginas/ConstituicaoRepublicaPortuguesa.aspx. (acedido em 13/05/2020).

[3] *Linguagem Gestual*, https://sites.google.com/site/pessoasportadorasdeficiencias/linguagem-gestual. (acedido em 13/05/2020).

[4] A. de Oliveira Monteiro Castanheira Costa, "Reconhecimento de Língua Gestual", tese de mestrado, Instituto Superior de Engenharia de Lisboa, Lisboa, Portugal, set. de 2014.

[5] D. L. Viana, "Uma linguagem de domínio específico para descrição e reconhecimento de gestos usando sensores de profundidade", tese de mestrado, Universidade Federal de Pernambuco, Recife, ago. de 2015.

[6] E. G. X. Moura, "Desenvolvimento de um sistema de captura de baixo custo destinado a determinação da cinemática humana", tese de mestrado, Universidade Federal do Rio Grande do Norte, Natal, Brasil, 2015.

[7] F. S. O. Matheus Lin Truglio Alvarenga Diogo S. Ortiz Correa, "Redes Neurais Artificiais aplicadas no Reconhecimento de Gestos usando o Kinect", em *Computer on the Beach*, Florianópolis, Brasil, mar. de 2012, ISBN: 978-85-7696-078-2.

[8] *Licensing*, https://www.ultraleap.com/product/leap-motion-controller/#pricingandlicensing. (acedido em 15/05/2020).

[9] *Meet your digital hands*, https://www.ultraleap.com/tracking/#how-it-works. (acedido em 15/05/2020).

[10] *UH-003206-TC Issue 2 LeapMotion Controller Data Sheet*, Ultrahaptics Ltd, The West Wing, Glass Wharf, Bristol, BS2 0EL, United Kingdom, dez. de 2019.

[11] *Use Xtion PRO developer solution to make motion-sensing applications and games*, https://www.asus.com/3D-Sensor/Xtion_PRO_LIVE/overview/. (acedido em 15/05/2020).

[12] *CyberGlove III*, http://www.cyberglovesystems.com/cyberglove-iii. (acedido em 17/05/2020).

[13] D. L. L. Cruz e L. Velho, "Kinect and RGBD Images: Challenges and Applications", em *25th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials*, Ouro Preto, Brazil, 2012.

[14] G. S. Cardoso, *Microsoft Kinect - Criando aplicações interativas com o Microsoft Kinect*. Brasil: Casa do Código, ISBN: 978-85-66250-21-3.

[15] *Xbox One: Microsoft esclarece regras de privacidade do Kinect*, https://www.techtudo.com.br/noticias/noticia/2013/11/xbox-one-microsoft-esclarece-regras-de-privacidade-do-kinect.html. (acedido em 15/05/2020).

[16] A. Jana, *Kinect for Windows SDK Programming Guide*, 1rd. United States: PACKT, 2012, ISBN: 9781849692380.

[17] J. A. Fabro, "Redes Neurais Artificias", tese de mestrado, Curso em Especialização em Inteligência Computacional, ago. de 2001.

[18] M. Nielsen, *Using neural nets to recognize handwritten digits*, 1rd. Book Online, 2019.

[19] Z. L. Kovács, *Redes Neurais Artificiais - Fundamentos e Aplicações*, 4rd. São Paulo: Livraria da Física, 2006, ISBN: 85-88325-14-4.

[20] T. B. L. Antônio de Pádua Braga André Ponce de Leon F. de Carvalho, *Redes neurais artificiais: teoria e aplicações*. Rio de Janeiro: LTC, 2000.

[21] A. M. P. C. e. C. N. V. Ana Caroline Gomes Vargas, "Um estudo sobre Redes Neurais Convolucionais e sua aplicação em detecção de pedestres", em *Electronic Proceedings of the 29th Conference on Graphics, Patterns and Images (SIBGRAPI'16)*, F. A. M. Cappabianco, F. A. Faria, J. Almeida e T. S. Körting, eds., São José dos Campos, SP, Brazil, out. de 2016. URL: http://gibis.unifesp.br/sibgrapi16.

[22] G. D. Juraszek, "Reconhecimento de produtos por imagem utilizando palavras visuais e redes neurais convolucionais", tese de mestrado, Universidade do Estado de Santa Catarina, Santa Catarina, Brasil, dez. de 2014.

[23] K. K. Yann LeCun e C. Farabet, "Convolutional Networks and Applications in Vision", em *Proceedings of 2010 IEEE International Symposium on Circuits and Systems*, Paris, France, mai. de 2010. DOI: 10.1109/ISCAS.2010.5537907.

[24] *BodyPosture Joint Indices*, https://www.mathworks.com/help/supportpkg/kinectforwindowsruntime/ug/acquire-image-and-skeletal-data-using-kinect-v1.html. (acedido em 17/05/2020).

[25] *Kinect Sensor*, https://docs.microsoft.com/en-us/previous-versions/microsoft-robotics/hh438998%28v%3dmsdn.10%29. (acedido em 17/05/2020).

[26] *Kinect for Windows: Human Interface Guidelines v2.0*, Microsoft Corporation, 2014.

[27] *Transfer Learning Using AlexNet*, https://www.mathworks.com/help/deeplearning/ug/transfer-learning-using-alexnet.html?s_tid=mwa_osa_a. (acedido em 17/05/2020).

[28] W. Yu, K. Yang, Y. Bai, H. Yao e Y. Rui, *Visualizing and Comparing Convolutional Neural Networks*, 2014. arXiv: 1412.6631 [cs.CV].

[29] *Pretrained Deep Neural Networks*, https://www.mathworks.com/help/deeplearning/ug/pretrained-convolutional-neural-networks.html. (acedido em 17/05/2020).

[30] *imageDataAugmenter*, https://www.mathworks.com/help/deeplearning/ref/imagedataaugmenter.html. (acedido em 17/05/2020).

[31] *Stochastic Gradient Descent with momentum*, https://towardsdatascience.com/stochastic-gradient-descent-with-momentum-a84097641a5d. (acedido em 17/05/2020).