CENTRO UNIVERSITÁRIO INTERNACIONAL UNINTER
ESCOLA SUPERIOR POLITÉCNICA
BACHARELADO EM ENGENHARIA DA COMPUTAÇÃO

HERDNEY SOUZA DOS SANTOS
LEILA FABIOLA FERREIRA
POLIANA GONÇALVES LEITE ALVES

LUVA TRADUTORA DA LÍNGUA BRASILEIRA DE SINAIS

TRABALHO DE CONCLUSÃO DE CURSO

CURITIBA
2019

HERDNEY SOUZA DOS SANTOS
LEILA FABIOLA FERREIRA
POLIANA GONÇALVES LEITE ALVES

LUVA TRADUTORA DA LÍNGUA BRASILEIRA DE SINAIS

Trabalho de Conclusão de Curso apresentado como requisito parcial à obtenção do título de Bacharel em Engenharia da Computação pela Escola Superior Politécnica do Centro Universitário Internacional Uninter.

Orientador: Prof. Me. Charles Way Hun Fung.

CURITIBA
2019

Dedicamos este trabalho aos nossos familiares, amigos e professores que sempre nos apoiaram nesta jornada.

AGRADECIMENTOS

Agradecemos primeiramente a Deus por nos permitir estar sempre evoluindo como ser humano, nos dando sabedoria para enfrentar os desafios impostos e nos fortalecendo continuamente.

Também registramos o nosso reconhecimento aos professores do curso que sempre se mostraram dispostos a nos orientar nos mais diversos assuntos e principalmente naqueles pertinentes a este projeto, em especial ao nosso professor orientador Charles que detém um vasto conhecimento em diversas áreas, sendo muito solícito nas necessidades do grupo.

Aos nossos familiares e amigos que sempre estiveram ao nosso lado nos apoiando também a nossa gratidão, pois são nossos pilares e nos mantém em pé diante das adversidades da vida.

“Seja um padrão de qualidade. As pessoas não estão acostumadas a um ambiente onde o melhor é o esperado.”
(JOBS, Steve)

RESUMO

SANTOS, Herdney Souza dos, FERREIRA, Leila Fabiola, ALVES, Poliana Gonçalves Leite Alves. **Luva Tradutora da Língua Brasileira de Sinais:** Trabalho de Conclusão de Curso (Bacharelado em Engenharia da Computação) – Centro Universitário Internacional Uninter. Curitiba, 2019.

Levando-se em consideração a necessidade de inclusão social por parte da população com alguma deficiência auditiva ou da fala, que a Libras é a linguagem amplamente adotada por essa parcela da população bem como daquelas que estão em seu convívio social ou também interessadas nesse tema, este trabalho apresenta um projeto desenvolvido por acadêmicos da Engenharia da Computação do Centro Universitário UNINTER, com o principal objetivo de construir um protótipo para tradução da língua brasileira de sinais (Libras ou LSB), por meio de uma luva de dados. Para tal propósito foram utilizados sensores flexíveis, sensor acelerômetro, dispositivo de comunicação *bluetooth* e interface com o usuário via aplicativo *Android*, sendo a aquisição de dados gerenciada por um microcontrolador da família PIC modelo 18F4550. O reconhecimento dos padrões da soletração digital foi realizado por um modelo de classificação previamente treinado. Para o embasamento teórico e técnico, foi realizada uma extensa pesquisa bibliográfica no que tange aos dispositivos de hardware, software e também assuntos de cunho social acerca dos principais interessados no uso da língua brasileira de sinais, destacando a atual situação da educação dos surdos no Brasil. Diante dos resultados obtidos nos testes, verifica-se a possibilidade do projeto ser aplicado no aprendizado de Libras e a gama de melhorias possíveis futuramente.

Palavras-chave: Libras. Microcontroladores. Mobile. Inteligência Artificial. Luva de Dados.

ABSTRACT

SANTOS, Herdney Souza dos, FERREIRA, Leila Fabiola, ALVES, Poliana Gonçalves Leite Alves. **Brazilian Sign Language Translator Glove**: Course Completion Work (Bachelor of Computer Engineering) - Uninter International University Center. Curitiba, 2019.

Considering the need for social inclusion by the population with some hearing or speech disability, and Libras being the language widely adopted by this portion of population as well as those in their social life, or also interested in this theme, this paper presents a project developed by Computer Engineering academics of the UNINTER University Center, with the main objective of building a prototype for the translation of Brazilian sign language (Libras or LSB), through a data glove. For this purpose, flexible sensors, accelerometer sensor, bluetooth communication device and user interface via Android application were used. Data acquisition was managed by a PIC model 18F4550 family microcontroller. Recognition of digital spelling patterns was performed by a previously trained classification model. For the theoretical and technical basement, an extensive bibliographic research was carried out regarding hardware devices, software and also social issues about the main stakeholders in the use of Brazilian sign language, highlighting the current situation of deaf education in Brazil. Given the results obtained in the tests, there is the possibility of the project being applied to learning Libras and the range of possible improvements in the future.

Keywords: Libras. Microcontrollers. Mobile. Artificial Intelligence. Data Glove.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS

ACK	Reconhecimento *(Acknowledgement)*
Char	Caractere *(Characte*r)
NACK	Não Reconhecimento (*Not Acknowledgment*)

LISTA DE SIGLAS

AD	Analógico-Digital (*Analog-to-digital)*
ADC	Conversor Analógico/Digital (*Analog-to-digital Converter*)
CM	Configuração de Mão
CPU	Unidade Central de Processamento (*Central Processing Unit*)
CS	Seletor de Chip (*Chip Select*)
DMP	Processador de Movimento Digital (*Digital Motion Processor*)
EF	Expressão Facial
FS-USB	Barramento Serial Universal de Velocidade Máxima (*Full Speed Universal Serial Bus*)
IA	Inteligência Artificial
IBGE	Instituto Brasileiro de Geografia e Estatística
I^2C	Circuito Inter Integrado (*Inter-Integrated Circuit*)
IDE	Ambiente de Desenvolvimento Integrado (*Integrated Development Environment*)
IO	Entrada e Saída (*Input/Output)*
KNN	K- Vizinhos Mais Próximos (K-*Nearest Neighbors*)
LSB	Língua de Sinais Brasileira
LCD	Display de Cristal Líquido (*Liquid Crystal Display*)
M	Movimento
MCLR	Reset Mestre (*Master Clear*)
MISO	Entrada do Mestre Saída do Escravo (*Master Input Slave Output*)
MOSI	Saída do Mestre Entrada do Escravo (*Master Output Slave Input*)
PA	Ponto de Articulação
PIC	Controladores de Interface Programável (*Programmable Interface Controller*)
PNS	Pesquisa Nacional de Saúde
RX	Recebimento (*Receive*)
SCK	Serial *Clock*
SCLK	Serial *Clock*
SDI	Entrada de dados seriais (*Serial Data In*)
SDO	Saída de dados seriais (*Serial Data Out)*

SPI	Interface Periférica Serial (*Serial Peripheral Interface*)
SPP	Protocolo de Porta Serial (*Serial Port Protocol*)
SRAM	Memória de Acesso Randômico Estática (*Static Random-Access Memory*)
SS	Seletor de Escravo (*Slave Select*)
TX	Transmissão (*Transmit)*

LISTA DE ACRÔNIMOS

ANATEL	Agência Nacional de Telecomunicações
CISC	Computador de Instruções Complexas (*Complex Instruction Computer)*
EEPROM	Memória Somente de Leitura Programável e Apagável Eletricamente (*Electrically Erasable Programmable Read-Only Memory*)
FENEIS	Federação Nacional de Educação e Integração dos Surdos
INEP	Instituto Nacional de Estudos e Pesquisas Educacionais Anísio Teixeira
LED	Diodo Emissor de Luz (*Light Emitter Diode)*
LIBRAS	Língua Brasileira de Sinais
MEC	Ministério da Educação
MIPS	Milhões de Instruções por Segundo (*Million Instructions Per Second*)
RAM	Memória de Acesso Randômico (*Random-Access Memory)*
RISC	Computador com Conjunto de Instruções Reduzido (*Reduced Instruction Set Computer)*
ROM	Memória Somente de Leitura (*Read-Only Memory)*
UART	Receptor/Transmissor Assíncrono Universal *(Universal Asynchronous Receive/Transmitter)*
USART	Receptor/Transmissor Síncrono Assíncrono Universal *(Universal Synchronous Asynchronous Receive/Transmitter)*

SUMÁRIO

1 INTRODUÇÃO

A comunicação é algo essencial entre as pessoas, seja através da fala, da escuta, por mensagens, e-mails, cartas, gestos e afins. Além de inúmeros idiomas e dialetos existentes no mundo, há também diversos outros tipos de expressões, gestuais, ou não, que são utilizadas para o entendimento entre os seres humanos. Segundo a Empresa Ágil (2015, p.6), é “através da comunicação que os seres humanos partilham diferentes informações entre si, tornando assim o ato de se comunicar uma atividade essencial para a vida em sociedade.”

Entretanto, algumas deficiências físicas as quais o ser humano é passível, vindas desde o nascimento, ou adquiridas ao longo do tempo, podem interferir na qualidade da comunicação em diversos níveis, e neste trabalho, interessa aquelas que afetam diretamente a audição e fala.

Em se tratando da surdez, segundo o Instituto Brasileiro de Geografia e Estatistica (IBGE, 2015) na PNS de 2013 (Pesquisa Nacional de Saúde), considerando os 200,6 milhões de pessoas residentes em domicílios particulares permanentes naquele ano, 6,2% possuía ao menos uma deficiência entre intelectual, física, auditiva e visual, das quais a surdez nos dois ouvidos, surdez em um ouvido e audição reduzida no outro, ou ainda audição reduzida de ambos os ouvidos ocupa 1,1% da população.

Ainda de acordo com as estatísticas do IBGE (2015) constantes na PNS (2013):

> Em 2013, 20,6% da população com deficiência auditiva apresentou grau intenso ou muito intenso de limitações ou não conseguia realizar as atividades habituais. No Brasil, 8,4% da população com deficiência auditiva frequentava algum serviço de reabilitação. (Não paginado)

Segundo Felipe (2005), que cita em seu artigo os dados do Censo Demográfico realizado pelo IBGE no ano 2000 e do Censo Escolar do ano 2003 realizado pelo MEC/INEP, o Brasil aparecia com estimados 5.750.805 surdos brasileiros. Desse total, 766.344 eram jovens entre 0 e 24 anos de idade, dos quais 2.041 (3%) possuíam ensino médio completo e apenas 344 haviam iniciado um curso superior.

Já os dados estatísticos apontados pelo IBGE (2010) em seu Censo Demográfico mais atual, de um total de 190.755,799 residentes permanentes brasileiros, 9.717,318 pessoas apresentam alguma dificuldade, grande dificuldade ou não consegue ouvir de modo algum, o que corresponde a 5,1% do total populacional.

Vale destacar também, a utilização de Libras por muitas dessas pessoas com deficiência auditiva, linguagem que é reconhecida como nacional por meio da Lei 10.436/2002 e regulamentada pelo Decreto 5626/2005 (QUADROS, 2013).

De acordo com as autoras Martins e Nascimento (2012, p.58), mesmo com todos os pontos deficitários relativos às políticas inclusivas advindas pós decreto, pode-se “(...) notar que desde 2005 houve uma crescente visibilidade no campo da surdez.”

1.1 JUSTIFICATIVA

Diante das informações expostas, o projeto tem como justificativa a melhoria nas comunicações entre pessoas que tem alguma deficiência física relativa à audição e/ou fala, bem como facilitar e incentivar o aprendizado de Libras aos que se interessam ou necessitam dessa linguagem. É uma importante iniciativa trabalhar com dispositivos que colaborem com a inclusão social, digital e acessibilidade, seja para aqueles que necessitem ou para os interessados em ingressar na pesquisa acerca desses temas.

1.2 OBJETIVO GERAL

Desenvolver um protótipo na configuração de luva de dados, para que a partir da leitura dos sensores flexíveis, acelerômetro e também da aplicação de um método de inteligência artificial, sejam reconhecidos e apresentados ao usuário via aplicativo *mobile*, os caracteres alfanuméricos correspondentes aos movimentos e configuração da mão que representam a datilologia da Língua Brasileira de Sinais.

Para se tornar possível a realização do objetivo principal do projeto, foi necessário atingir alguns objetivos específicos constantes no próximo tópico.

1.3 OBJETIVOS ESPECÍFICOS

- Projetar o protótipo nas melhores condições estruturais e de funcionamento, com precisão e consistência;
- Realizar a comunicação via *bluetooth* entre o protótipo e o aplicativo *mobile*;
- Aplicar um método de inteligência artificial para classificação dos caracteres alfanuméricos, visando maior leveza e acurácia possível;
- Desenvolver um aplicativo mobile com interface amigável e com funcionalidades úteis para comunicação e resposta ao usuário;

- Interligar de maneira objetiva todos os módulos do projeto, entendendo seus conceitos e funcionamento de maneira que possibilite a evolução da aplicação.

1.4 ORGANIZAÇÃO DO TRABALHO

Este trabalho compõe-se de cinco capítulos no qual o primeiro traz uma introdução ao tema justificando o desenvolvimento da pesquisa e objetivos do projeto. O segundo capítulo aborda o referencial teórico onde se tem uma revisão bibliográfica dos assuntos que envolvem este trabalho. O terceiro capítulo apresenta a metodologia aplicada e os materiais necessários para a construção do protótipo. O quarto capítulo descreve quais resultados foram obtidos, além de uma análise sobre o projeto como um todo. Por fim no quinto capítulo têm-se as conclusões obtidas com a versão final do projeto, onde se destaca o que foi produzido e propostas para futuros projetos. No sexto capítulo constam as referências bibliográficas utilizadas no desenvolvimento do projeto.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão apresentados os temas que esta pesquisa abrange, contextualizando desde o tema foco que é a Língua Brasileira de Sinais (Libras ou LSB), explorando um pouco sobre os trabalhos e pesquisas já realizadas acerca do tema proposto, bem como os demais assuntos técnicos que corroboraram para o desenvolvimento deste projeto.

2.1 LÍNGUA BRASILEIRA DE SINAIS

Segundo Castro e Carvalho (2011), a Libras é considerada a língua materna dos surdos, sendo assim de extrema importância seu aprendizado como forma de comunicação dos surdos e/ou mudos em sociedade. De acordo com Martins (2011, p.61), pesquisadores brasileiros defendem a aquisição de Libras o mais breve possível para o desenvolvimento educacional dos surdos.

Uma abordagem mais técnica sobre a língua brasileira de sinais pode ser descrita como:

> A Libras é composta por sinais que correspondem, em português, a palavras, entretanto não se trata simplesmente de uma substituição, uma palavra por um sinal correspondente, ela tem suas peculiaridades, além de ser independente da língua portuguesa. Ou seja, ela não se reflete na estrutura gramatical da língua oral, mas possui uma estrutura própria (...) (ZILIO, 2012, p.29).

Ainda citando Zilio (2012, p.37), é interessante ressaltar que “(…) em cada país uma língua de sinais diferente é utilizada: na Argentina, a língua argentina de sinais; na Bolívia, a língua boliviana de sinais; no Equador, a língua equatoriana de sinais”, podendo-se inferir que as línguas de sinais independem da língua falada de origem (no exemplo o espanhol), ou seja, são línguas naturais dos surdos. A criação de uma língua universal, seja ela oral ou de sinais, é um anseio de muitos, mas é também um objetivo difícil de concretizar (ZILIO, 2012). Diante disso, deve-se focar nas características histórico culturais que permeiam cada país e entender os desafios da

educação inclusiva dos surdos na atualidade. De acordo com Soares[1], (1999, apud Martins, 2011), antes da década de 1960, a educação dos surdos consistia em sistemas segregados de ensino nas residências em grandes capitais brasileiras, tendo caráter assistencialista e filantrópico, desconsiderando a integração escolar.

A mudança inicial veio apenas na década seguinte, conforme exposto por Martins (2011, p.33), “Tem-se o conhecimento de que somente a partir da década seguinte a lei 4.024/71 propõe que a educação dos deficientes seja enquadrada no sistema geral da educação, com a finalidade de integrá-los à comunidade ouvinte.”

Essa integração é um fator de extrema importância para o acesso de pessoas com deficiência a escolarização. Porém, outras questões também relevantes devem ser levadas em consideração quando se fala em educação dos surdos, mesmo porque, as dificuldades na comunicação já aparecem desde cedo no âmbito familiar, onde muitas vezes os pais demonstram forte resistência para aceitação da língua de sinais para seus filhos (MARTINS, 2011), crentes sobre um possível desenvolvimento da oralidade. Por isso, muitos autores discutem questões pedagógicas e sociais associadas aos surdos em uma tentativa de encontrar o melhor método de inclusão, conforme exemplo analisado por Martins (2011):

> Os professores acreditavam que ao simplificar os materiais impressos oferecidos aos surdos, estes aprenderiam a ler e a escrever mais facilmente. Uma das justificativas para tal medida refere-se às dificuldades dos surdos em compreender e utilizarem-se da língua majoritária. Embora as crianças surdas convivam em uma sociedade letrada, na qual a leitura e a escrita são valorizadas socialmente, a poucas tem sido conferida chance para manusear portadores de textos autênticos que circulam fora das esferas escolares. (p.200).

Levando em consideração esse histórico e com o advento da evolução da tecnologia, os dispositivos podem auxiliar nas relações de convívio de pessoas com deficiência na sociedade.

---

[1] SOARES, M. a. l. **A educação do surdo no Brasil**. Campinas: autores associados; Bragança Paulista: Edusf, 1999.

2.1.1 Estrutura da Libras e sua Datilologia

Levando-se em consideração a estrutura que compõe a Libras, de acordo com o exposto por Brito et. al (1997):

> A LIBRAS é dotada de uma gramática constituída a partir de elementos constitutivos das palavras ou itens lexicais e de um léxico (o conjunto das palavras da língua) que se estruturam a partir de mecanismos morfológicos, sintáticos e semânticos que apresentam especificidade, mas seguem também princípios básicos gerais. (BRITO et.al., 1997, p.23).

As palavras em Libras, possuem um sinal com estrutura distinta das palavras em português (BRITO et. al., 1997), bem como apresenta diferença entre soletração digital e sinal, como se pode perceber na Figura 1:

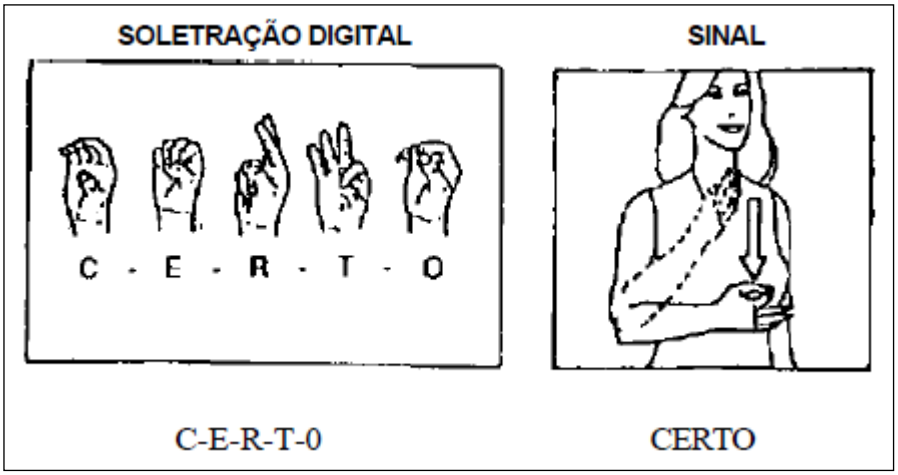


Figura 1 – Diferença entre soletração digital e sinal
Fonte: Brito et. al (1997)

Uma importante característica da LSB é o fato desta não abranger somente os sinais que compõe a configuração de movimento da mão para representação de palavras ou expressões, mas também possui a aplicação da datilologia (soletração digital representada na Figura 1) que corresponde a “soletrar” palavras com os sinais manuais usando caracteres alfanuméricos (CASTRO E CARVALHO, 2011), sendo algumas palavras apenas representadas por datilologia como, por exemplo: nomes de pessoas, localidades ou palavras que não possuem sinal específico atribuído

(BRITO et. al., 1997), fazendo uso do alfabeto manual que consta no Anexo A deste documento.

O sinal em Libras representa uma palavra ou expressão, não sendo articulado linearmente como as soletrações ou como as palavras em português, que se formam pela justaposição linear de suas unidades mínimas, os fonemas. Em Libras, as unidades mínimas são espaciais (forma da mão ou do sólido, movimento linear e com retenção, vetores orientacionais da mão, etc.) (BRITO et. al., 1997). Sendo assim podemos distinguir as unidades mínimas em Libras considerando os parâmetros de configuração da mão (CM), pontos de articulação (PA), movimento/orientação (M) e expressão facial (EF) conforme ilustrado a seguir:

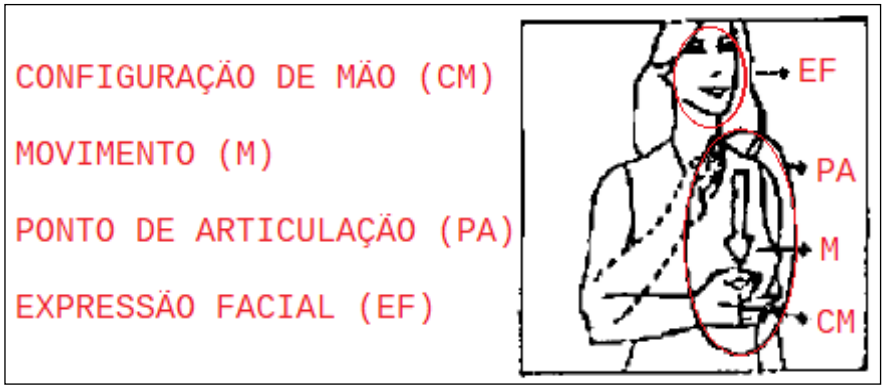


Figura 2 – Parâmetros do sinal em Libras
Fonte: Adaptado de Brito et. al. (1997)

O projeto baseou-se então em sinais datilológicos, ou seja, são reconhecidos movimentos e configuração de mão que expressam caracteres alfanuméricos em Libras isoladamente.

2.2 ESTADO DA ARTE

Como explanado anteriormente, cada país possui uma língua de sinais própria e, portanto, as configurações de mão apresentarão diferenças. Para discutir acerca dos projetos já realizados em diferentes países, foram analisados diversos trabalhos anteriores com propostas semelhantes. Com intuito de tornar mais fácil a compreensão das características de cada pesquisa a análise do material foi disposta em um quadro comparativo (Quadro 14). Analisando o quadro citado, percebe-se que as aplicações contam predominantemente com dispositivos que realizam a aquisição

de dados através de luvas de dados, por visão computacional, fazendo-se o processamento das imagens obtidas por câmeras e ainda os processos de aquisição híbridos que utilizam essas duas técnicas.

Alguns trabalhos utilizam luvas de dados prontas, por exemplo: As luvas da *CyberGlove® System* ilustradas abaixo (WANG; GAO; SHAN., 2002) (OZ; LEU, 2011), que já contam com diversos sensores acoplados e são fabricadas para projetos que necessitem fazer a leitura dos movimentos das mãos, sendo disponibilizados também duas aplicações, a *VirtualHand* SDK para captura de movimento da mão e CAD *eValuator* para avaliação de alcance e ergonomia em 3D (CYBERGLOVE, 2017):

Figura 3 – Modelos de luva da *CyberGlove® System*
Fonte: http://www.cyberglovesystems.com[2]

Em outros projetos (LAZZAROTTO, 2016; SWEE et al., 2007), a luva e/ou os sensores foram fabricados pelo(s) autor(es), utilizando diferentes tipos de sensoriamento e processamento de dados, como no caso deste projeto, onde a luva de dados foi projetada com 6 sensores. Abaixo, é ilustrado o circuito final de Fahn e Sun (2005), na qual os sensores redutores baseados em indução magnética utilizados, foram confeccionados pelos próprios autores:

[2] CyberGlove® System: Disponível em <http://www.cyberglovesystems.com/> Acessado em 13 nov. 19.

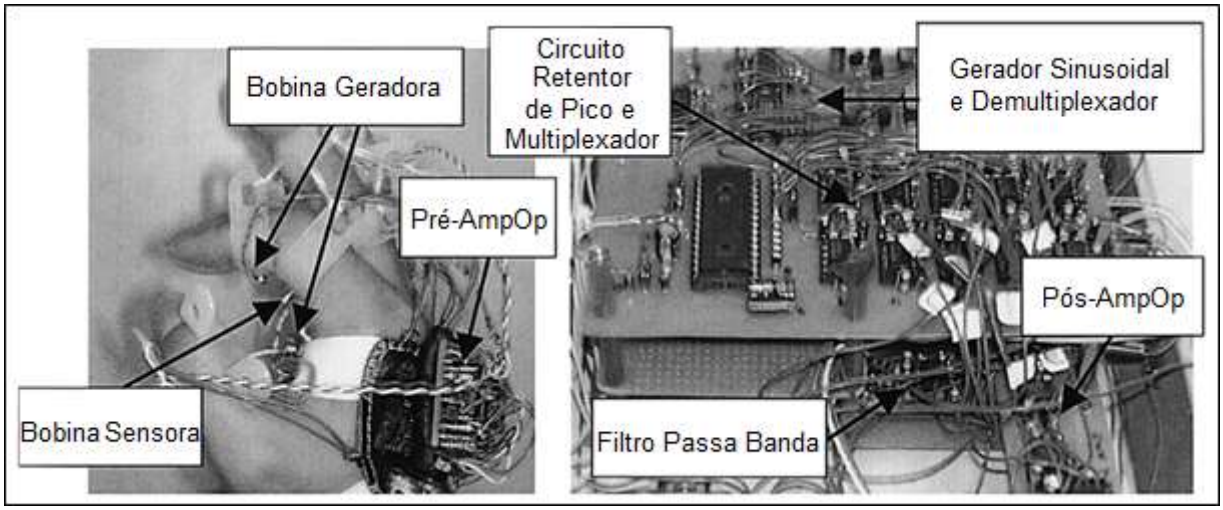


Figura 4 – Luva de dados e sua unidade de controle
Fonte: Adaptado de Fahn e Sun (2005)

É importante salientar que todos os fatores relacionados à construção do protótipo e processamento dos dados irão influenciar diretamente no desempenho, efetividade e acurácia. Por exemplo, quanto mais sensores utilizados, maior a precisão da leitura da configuração e movimento da mão, porém demanda-se maior capacidade de processamento e gerenciamento do consumo de energia (FAHN; SUN, 2005).

2.3 MICROCONTROLADORES

Primeiramente é necessário explanar sobre a definição de um microcontrolador de modo geral. Sendo assim, pode-se definir um microcontrolador como:

> (...) um computador de chip único e barato. Computador com chip único significa que todo o sistema está dentro dos limites do chip do circuito integrado. O microcontrolador na tira de silício encapsulada possui características semelhantes às do computador pessoal padrão. (IOVINE, 2000, p.1, tradução nossa).

Ainda segundo Iovine (2000), a principal característica do microcontrolador é o fato de ser capaz de armazenar e executar um programa, contendo uma CPU (Unidade Central de Processamento), memória RAM (Memória de Acesso

Randômico), memória ROM (Memória Somente de Leitura), pinos de E/S (Entrada/Saída), portas seriais e paralelas, *timers* e, às vezes, outros periféricos internos, como conversores AD (Analógico-Digital).

Existem dois tipos de arquiteturas convencionais nos microcontroladores. A Von Neumann que é usada em grande parte dos microcontroladores, disponibiliza todo o espaço de memória no mesmo barramento; enquanto que na arquitetura de Harvard o código e os dados se encontram em barramentos distintos, permitindo que sejam consultados simultaneamente e resultando em maior desempenho (IBRAHIM, 2008), conforme ilustrado a seguir:

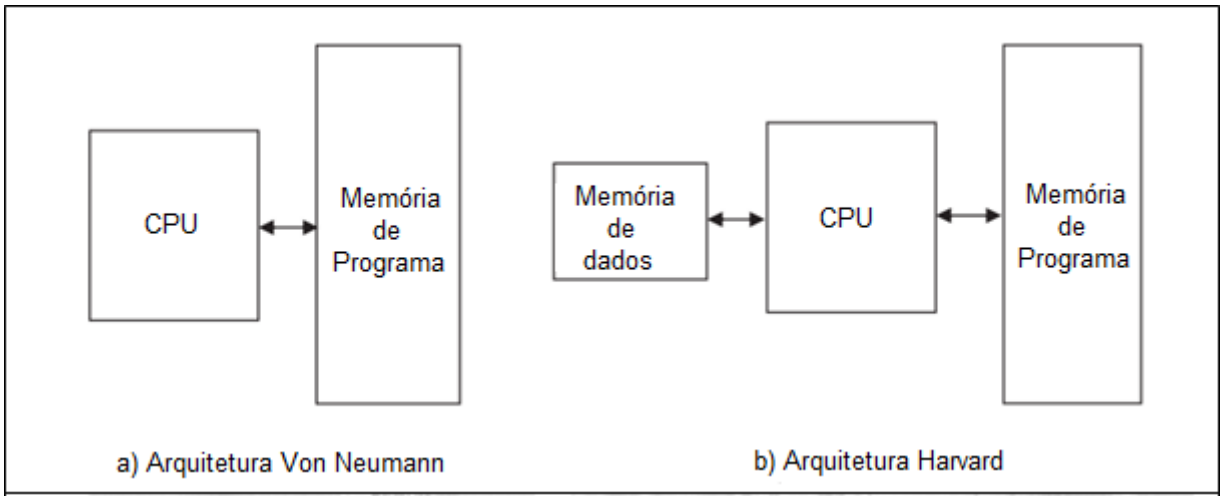


Figura 5 – Arquiteturas de Von Neumann e Harvard

Fonte: Adaptado de Ibrahim (2008)

Levando-se em consideração o conjunto de instruções de um microcontrolador, este pode ser classificado como RISC[3] (Computador com Conjunto de Instruções Reduzido) e CISC[4] (Computador de Instruções Complexas). Um microcontrolador RISC com 8 bits, possui dados com 8 bits de largura, porém os dados das instruções têm geralmente 12, 14 ou 16 bits ocupando um endereço de memória do programa. Deste modo, as instruções são buscadas e executadas a cada ciclo, melhorando o desempenho. Já nos microcontrolador CISC, tanto dados quanto as instruções tem a

[3] RISC - (*Reduced Instruction Set Computer)* - (IBRAHIM, 2008).

[4] CISC (*Complex Instruction Computer*) - (IBRAHIM, 2008).

mesma largura, tendo dados e código no mesmo barramento e por isso não podem ser acessados simultaneamente (IBRAHIM, 2008).

Ao iniciar um projeto, para escolha correta de qual microcontrolador aplicar, algumas características devem ser avaliadas (BATES, 2008):

- Número de portas de entradas e saídas;
- Tamanho da memória de programa;
- Tamanho da RAM de dados;
- Memória de dados não voláteis;
- Velocidade máxima do timer.
- Interfaces;
- Suporte ao sistema de desenvolvimento;
- Custo e disponibilidade.
- Consumo de energia.

2.3.1 Microcontrolador PIC 18F4550

O microcontrolador PIC 18F4550 da Microchip® de arquitetura Harvard e tecnologia RISC, conta com um conjunto de 75 instruções, possui 40 pinos e é indicado para aplicações de baixa energia. Fornece conectividade através de três portas seriais, sendo elas a FS-USB (Barramento Serial Universal de Máxima Velocidade) com taxa de transmissão de 12 Mbit por segundo, a I^2C (Circuito Inter Integrado) e SPI (Interface Periférica Serial) com taxas de transmissão de até 10 Mbit por segundo (MICROCHIP, 2019).

Possui memória de programa *flash* de 32 *Kbytes* e 16384 palavras de instruções. Sua memória de dados conta com uma SRAM (Memória de Acesso Randômico Estática) de 2 *Kbytes* e EEPROM (Memória Somente de Leitura Programável e Apagável Eletricamente*)* de 256 *bytes* (DATASHEET, 2009).

As portas configuráveis como entrada ou saídas digitais são 35, dentre elas 13 podem ser configurados como entradas analógicas, sendo os canais AN0 ao AN12 com conversor analógico digital integrado de 10 bits (DATASHEET, 2009). Uma visão mais detalhada dos 40 pinos deste microcontrolador e suas funções é ilustrado a seguir:

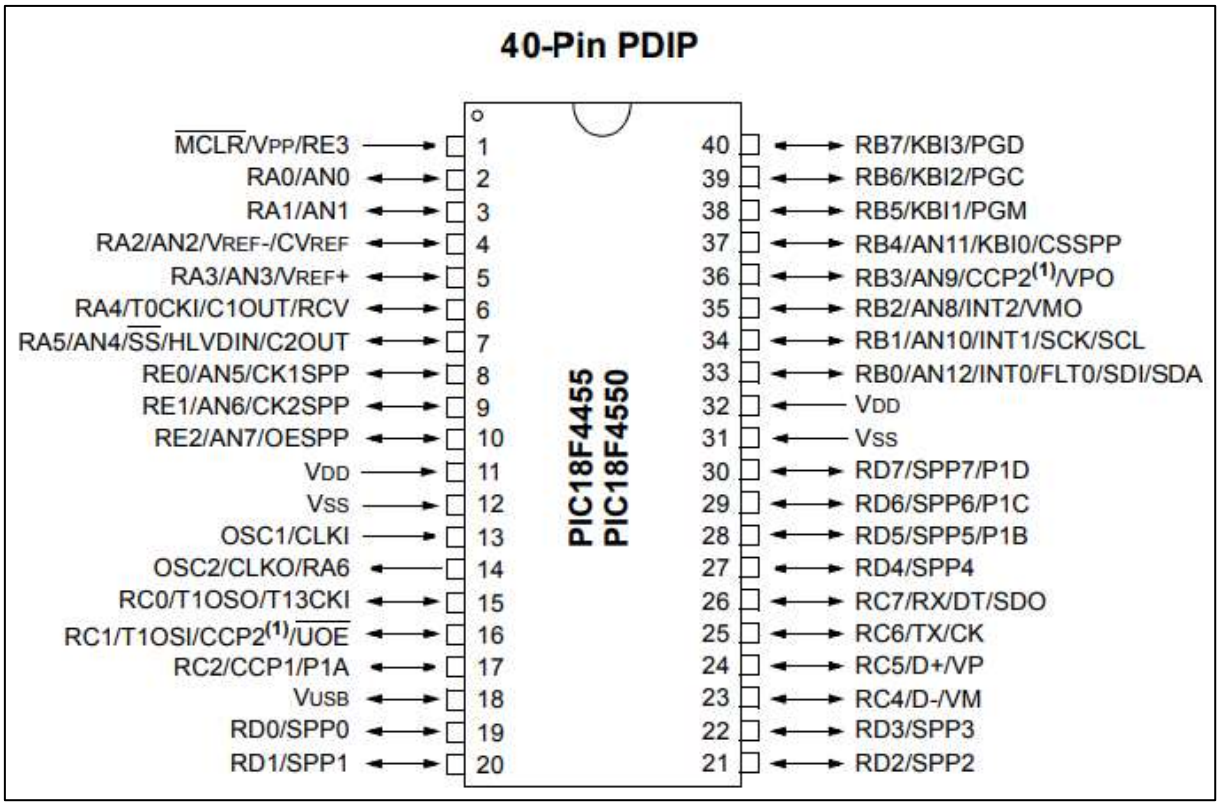


Figura 6 – Diagrama de Pinagem do PIC18F4455 e PIC18F4550 PDIP

Fonte: Datasheet (2009)

A frequência máxima de operação deste microcontrolador é 48 Mhz, chegando a 12 MIPS (Milhões de Instruções por Segundo) (MICROCHIP, 2019). Todas as configurações necessárias para o uso apropriado das funcionalidades disponíveis devem ser feitas observando-se os registradores específicos.

2.4 COMUNICAÇÃO SERIAL

Muitas aplicações necessitam que o microcontrolador se comunique com um ou mais dispositivos externos, estando eles na mesma placa do circuito ou a muitos metros de distância (PEREIRA, 2007). Para que essa comunicação seja possível, existem duas principais categorias: a serial e a paralela. Comparado às conexões paralelas, que contém ao menos oito linhas de dados mais sinais de controle, as conexões seriais utilizam apenas um ou dois fios de sinal sendo extremamente úteis para aplicações, reduzindo a necessidade de várias linhas de dados (BATES, 2008).

Os microcontroladores da família PIC oferecem alguns tipos de interfaces de comunicação serial, cada uma com suas especificações, devendo ser avaliado qual

melhor se adapta a aplicação de acordo com a distância entre os dispositivos, velocidade ou taxa de comunicação e número de conexões necessárias (BATES, 2008). Além disso, devem ser considerados os dois tipos de comunicação serial, classificados de acordo com seu sincronismo: comunicação assíncrona e síncrona que serão discutidas posteriormente.

Um dispositivo USART (Transmissão e Recebimento Universal Síncrono / Assíncrono) normalmente é aplicado em seu modo assíncrono para implementar conexões externas. Isso significa que não é necessário nenhum sinal de *clock* externo para coordenar o envio e o recebimento de dados (BATES, 2008), porém existem também os modos de transmissão síncrona, os quais possuem um sinal de *clock* para sincronizar o envio e recebimento de dados, necessitando portanto de mais fios em sua conexão.

2.4.1 Comunicação Serial Assíncrona

Como citado anteriormente, a comunicação serial assíncrona não possui um sinal de *clock* para controlar as transmissões de dados. Sendo assim, é necessária apenas uma conexão para envio de dados, uma conexão para recebimento de dados e um fio de aterramento (BATES, 2008). Segundo Sousa e Souza (2012, p.181), “(...) o sincronismo é feito pelo próprio dado, isto é, os sistemas conseguem saber quando cada dado começa”.

No protocolo o sinal possui 8 *bits* de dados, um *start bit* e um *stop bit,* opcionalmente também pode conter um bit de paridade, assim, se a paridade do byte não coincidir com esse bit, é detectado um erro na transmissão. A quantidade de *bits* enviados por segundo é definido pela taxa de transmissão ou *baud rate*. Um valor bem comum de *baud rate* é 9600 (cerca de 9600 *bits* por segundo) (BATES, 2008).

A imagem a seguir ilustra as conexões necessárias para uma comunicação serial assíncrona bem como a estrutura de um sinal:

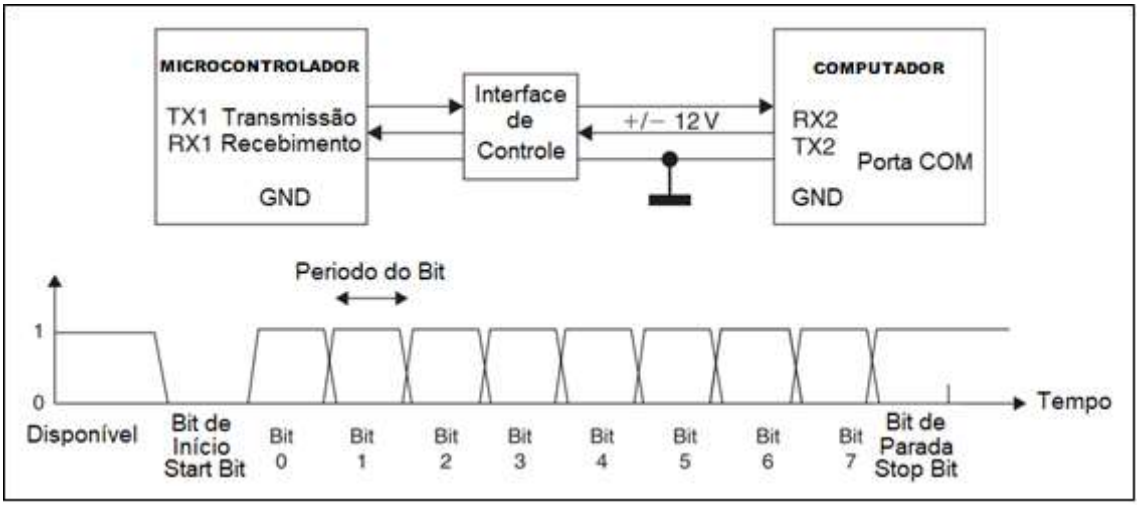


Figura 7 – Conexão e sinal Serial

Fonte: Adaptado de Bates (2008)

Para que ocorra a transmissão dos dados de maneira correta, os dispositivos em comunicação serial devem estar operando na mesma taxa de transmissão de bits (*baud rate*), ou seja; o dispositivo receptor deve ser inicializado com as mesmas configurações do transmissor (BATES, 2008).

2.4.2 Comunicação Serial Síncrona SPI (Interface Periférica Serial)

O SPI é um protocolo de comunicação síncrono de alta velocidade, que utiliza o modelo mestre/escravo, desenvolvido originalmente pela Motorola e atualmente adotado por diversos fabricantes (PEREIRA, 2007) (BATES, 2008).

Sua interface física é formada pelas linhas CS ou SS (Seletor de Chip ou Seletor de Escravo) que seleciona o dispositivo escravo o qual o mestre deseja se comunicar e também para encerrar os comandos transmitidos pelo mestre, a linha de *clock* no pino SCK ou SCLK (fonte de clock) que é utilizado para sincronização da comunicação entre mestre e escravo, o SDI ou MOSI[5] (entrada de dados seriais) e o SDO ou MISO[6] (saída de dados seriais) (PEREIRA, 2007). A seguir é ilustrado as conexões no modelo SPI:

[5] Saída do Mestre Entrada do Escravo (*Master Output Slave Input*).

[6] Entrada do Mestre Saída do Escravo (*Master Input Slave Output*).

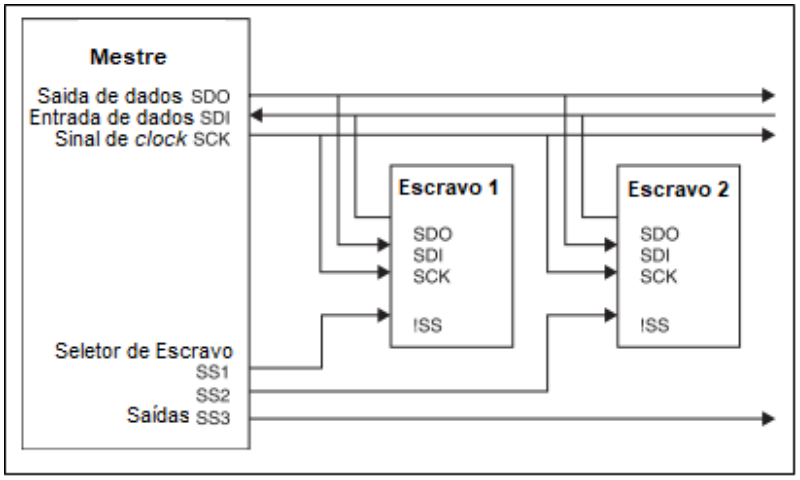


Figura 8 – Conexão SPI

Fonte: Adaptado de Bates (2008)

Para iniciar a transferência de dados, o mestre faz a seleção do dispositivo escravo para comunicar-se deixando em nível baixo a linha SCK e SS. Assim, oito bits de dados são enviados ao escravo ou recebidos pelo mestre, não sendo necessário *bits* de início e parada (*start* e *stop bits*) (BATES, 2008) conforme a Figura 9:

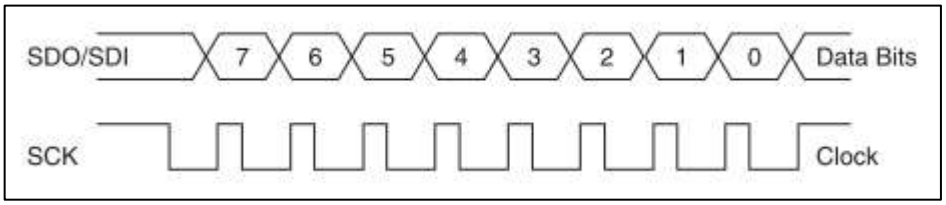


Figura 9 – Sinal SPI

Fonte: Bates (2008)

2.4.3 Comunicação Serial Síncrona I^2C (Circuito Inter Integrado)

O I^2C é um protocolo bidirecional de comunicação criado pela Philips com intuito de facilitar o desenvolvimento de sistemas modulares para televisores e outros aparelhos eletrônicos para consume geral. É um protocolo síncrono que utiliza duas linhas, sendo uma delas o *clock* (linha SCL), e a outra linha de dados (linha SDA) (PEREIRA, 2007). Foi projetado para comunicações de curto alcance entre dispositivos que estão no mesmo circuito ou equipamento eletrônico usando endereçamento de software (BATES, 2008).

Segundo Pereira (2007, p.275) “Graças à especificação de saídas em coletor (ou dreno) aberto, o protocolo permite a ligação de diversos dispositivos nas mesmas

linhas, formando um autêntico barramento de comunicação serial, ou uma rede de dispositivos.”

Assim como os demais protocolos síncronos, o I^2C utiliza o modelo mestre/escravo, porém suporta o modo “*multimastering*”, onde podem existir diversos mestres no mesmo barramento, no entanto, durante a comunicação permite somente um mestre estar ativo por vez para não ocorrer colisões (PEREIRA, 2007).

O modo de operação é dito por “reconhecimento”, ou seja, um dispositivo reconhece uma requisição de comunicação a ele endereçada através do envio do bit “ACK” (abreviação de *acknowledgement*), permitindo então que os dados solicitados sejam enviados em uma direção no barramento I^2C e, por fim, outro bit "ACK" enviado indica que os dados foram recebidos (MICROCHIP, 2001).

Cada dispositivo na rede I^2C deve possuir um endereço único, para que seja possível a comunicação, visto que somente dois dispositivos podem trocar dados por vez. A Figura 10 ilustra a conexão e estrutura do sinal no protocolo I^2C:

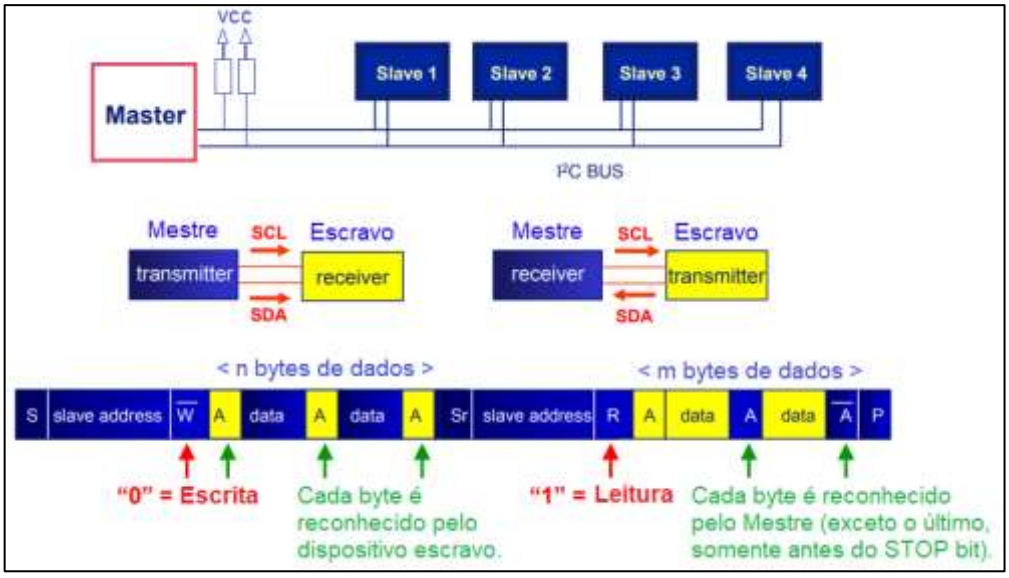


Figura 10 – Conexão e componentes do sinal I^2C

Fonte: Adaptado de Irazabal e Blozis (2003)

Segundo Pereira (2007), existem três princípios a serem seguidos para se estabelecer uma comunicação utilizando protocolo I^2C, são eles:

1. As informações constantes na linha SDA, somente serão lidas durante a fase alta da linha de *clock* (SCL);

2. O nível da linha de dados SDA, pode ser alterado somente na fase baixa da linha de *clock*;

3. Quando o barramento não estiver em uso, as linhas de SDA e SCL permanecem desligadas, portanto, em nível lógico alto devido aos resistores de *pull-up*.

O início e fim de uma transmissão ocorrem violando o segundo princípio, ou seja, forçando a linha SDA para nível lógico baixo durante a fase alta do *clock* (*start condition*) que indica o início de uma transmissão, ou forçando a linha SDA para nível lógico alto também durante a fase alta do *clock,* o que indica o fim da transmissão (*stop condition*) (PEREIRA, 2007), conforme ilustrado na Figura 11. Após a condição de início, são enviados 7 ou 10 bits que correspondem ao endereço do escravo a ser acessado e um bit indicando se a operação é de leitura (nível lógico 1) ou escrita (nível lógico 0). Caso o dispositivo a ser acessado, retorne um bit “ACK” (nível lógico 0), ou seja, reconhecendo a comunicação, os 8 bits de dados serão enviados ou recebidos pelo Mestre. Cada *byte* de dados trafegados deve ser reconhecido com um bit “ACK” em seguida, exceto o último *byte* enviado, que precede um bit “NACK” (nível lógico 1) encerrando a transmissão, conforme Figura 11 (IRAZABAL; BLOZIS, 2003).

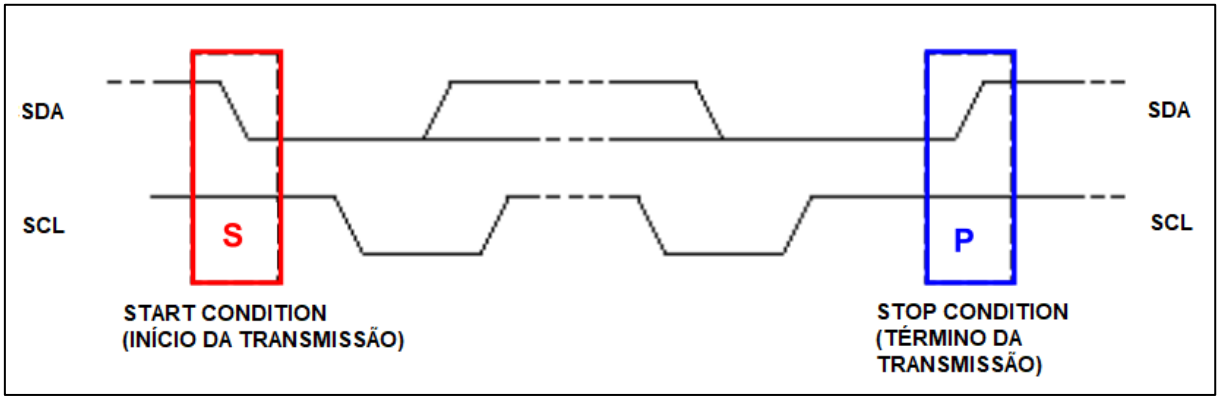


Figura 11 – Início e Término de uma comunicação I^2C

Fonte: Adaptado de Irazabal e Blozis (2003)

2.5 SENSORES FLEXÍVEIS

Esse tipo de sensor é resistivo, ou seja, sua aplicação baseia-se na alteração da resistência do material conforme ângulo de flexão do sensor, aumentando a

resistência quando o sensor é flexionado. São aplicados de modo geral, para medição de efeitos de torção ou medição de movimentos dos dedos como luva de dados, mas também são úteis em aplicações da robótica, equipamentos médicos, periféricos de computador, instrumentos musicais entre outros (BELL, 2013) (SPECTRASYMBOL, 2014).

O sensor flexível de 2,2 polegadas (5,5 cm) de comprimento ativo, em sua posição plana, apresenta uma resistência média de 25 kOhm conforme Figura 12, podendo chegar a 125 kOhm quando flexionado (DATASHEET).

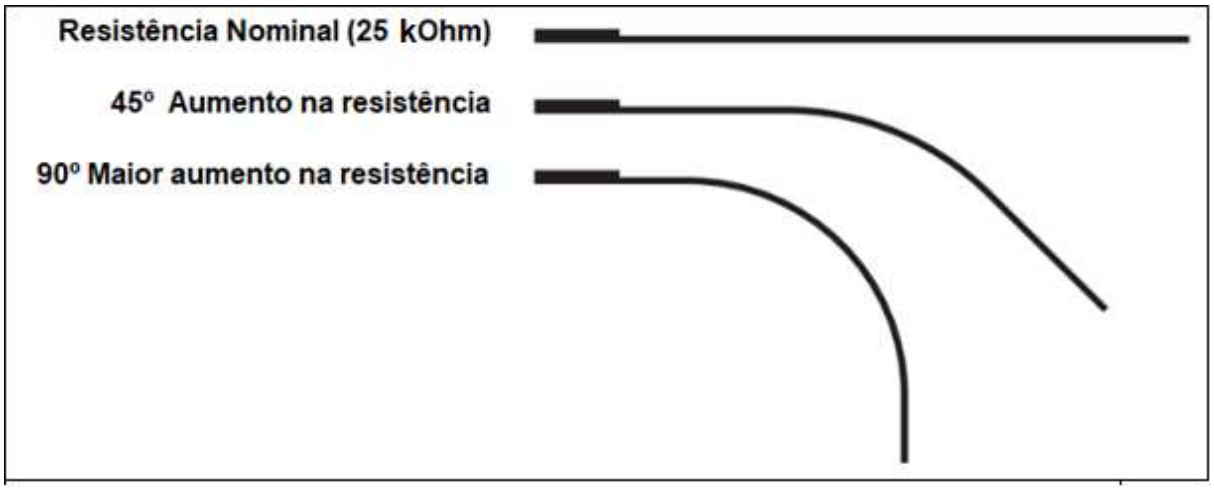


Figura 12 – Funcionamento do Sensor Flexível

Fonte: Adaptado de Datasheet

Para realizar a aquisição dos dados fornecidos por esse sensor, é necessário aplicá-lo em um circuito divisor de tensão, onde as alterações na flexão do sensor irão gerar mudanças na tensão no ponto de saída do divisor. Estas mudanças de tensão são entradas para os conversores AD. A Figura 13 demonstra um circuito divisor de tensão com o uso do sensor flexível:

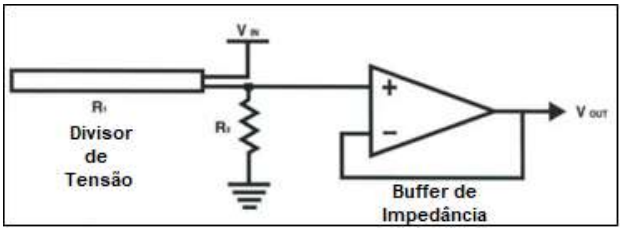


Figura 13 – Circuito Básico Sensor Flexível

Fonte: Adaptado de *Datasheet*

Segundo o *Datasheet* do sensor, o uso do amplificador operacional (*Buffer* de Impedância) é opcional, porém quando usado, sua baixa corrente de polarização, reduz o erro do sensor devido à impedância da fonte do sensor em um circuito divisor de tensão.

Para calcular o valor da tensão de saída em um divisor de tensão, utiliza-se a equação a seguir:

$$Vout = Vin.\left(\frac{R1}{R1+R2}\right) \quad (1)$$

Onde:
Vout é a tensão de saída;
Vin é a tensão de entrada;
R1 e *R2* são os resistores em série no divisor de tensão.

2.5.1 Conversor AD

Os microcontroladores são dispositivos digitais, sendo assim, só entendem e processam dados binários (zero e um), enquanto que grande parte dos sensores medem grandezas analógicas (distância, temperatura, pressão, volume entre outros) que são valores contínuos no tempo. Devido a essa diferença, os valores de tensão lidos em dispositivos analógicos precisam ser convertidos em um padrão digital para que possa ser processado pelo microcontrolador (MATIC, 2000). Para tanto, segundo Matic (2000):

> Esta tarefa é realizada por um bloco para conversão de analógico para digital ou por um ADC[7]. Esse bloco é responsável por converter informações sobre algum valor analógico em um número binário e enviá-las para um bloco da CPU para que possa processá-lo ainda mais (p. 12, tradução nossa).

[7] ADC: *Analog to Digital Converter* ou Conversor Analógico/Digital

As aplicações muitas vezes requerem a conversão de mais de um sinal analógico, sendo necessário o uso de um multiplexador, que seleciona apenas um sinal por vez para conversão em casos onde existe apenas um conversor (IBRAHIM, 2008). A seguir é ilustrado o modo de operação com multiplexador, 4 entradas de sinal analógico e um canal ADC:

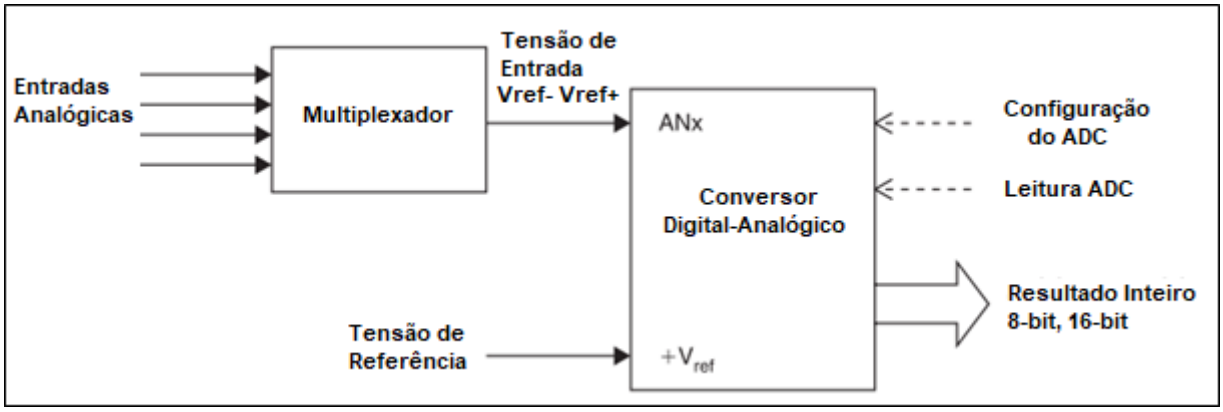


Figura 14 – Operação ADC com Multiplexador

Fonte: Adaptado de Bates (2008)

Ao finalizar o processo, o conversor gera uma *flag* de sinalização ou uma interrupção, indicando o término da conversão, assim os dados de saída do ADC podem então ser processados pelo dispositivo digital conectado a ele (IBRAHIM, 2008).

O número de bits de um conversor AD define sua resolução e o menor valor de tensão que pode ser detectado correspondendo a:

$$V_{LSB} = \frac{(V_{REF+} - V_{REF-})}{2^n} \quad (2)$$

Onde:

V_{LSB} é o menor valor de tensão detectado;

V_{REF} - é a tensão de referência negativa;

V_{REF} + é a tensão de referência positiva;

2^n define a resolução do conversor AD, sendo n o número de bits.

Por exemplo, em um circuito onde as tensões de referência V_{REF-} e V_{REF+} são respectivamente 0 V e 5 V, e utilizando um conversor AD de 10 bits de resolução, teremos:

$$V_{LSB} = \frac{(5V - 0V)}{2^{10}} = \ 0{,}00488V \ ou \ 4{,}88 \ mV \qquad (3)$$

Assim, a cada 4,88 mV, aumenta-se em uma unidade a grandeza decimal que corresponde à conversão resultante. Para calcular o valor decimal que resultará de uma tensão específica de entrada, basta dividir o valor da tensão de entrada pelo valor que corresponde a menor tensão que pode ser detectada nesse circuito. Por exemplo, com uma entrada de 4,5V no exemplo anterior, tem-se 4,5V/4,88 mV, que resulta em 922 decimais (resposta que o conversor AD retorna de acordo com a quantidade de bits de resolução). Fazendo a operação inversa (922 multiplicado por 4,88 mV), o valor resultante seria 4,49 V. Percebe-se que existe um pequeno erro entre o valor original da entrada e o valor correspondente à operação inversa. Esse erro ocorre até mesmo em conversores AD ideais, sempre quando um valor contínuo (infinito) é quantizado em um número finito de amostras (HEDAYATI, 2011)

2.6 SENSOR ACELERÔMETRO/GIROSCÓPIO (MPU6050)

O MPU-6050 é uma unidade de processamento de movimento que possui um acelerômetro, um giroscópio, um sensor de temperatura e o recurso Processador de Movimento Digital (*Digital Motion Processor* ou DMP) (DATASHEET, 2013).

Para a comunicação com o processador do sistema, é necessário utilizar o protocolo I^2C, por meio dos pinos SDA e SCL. O MPU-6050 sempre atua como escravo e “(...) o nível lógico para comunicações com o mestre é definido pela tensão no VLOGIC” (DATASHEET, 2013, p.25). Possui seis conversores analógico-digitais de 16 bits com finalidade de digitalizar as saídas nos eixos x, y e z do giroscópio e acelerômetro (DATASHEET, 2013). A Figura 15 ilustra o chip do MPU-6050:

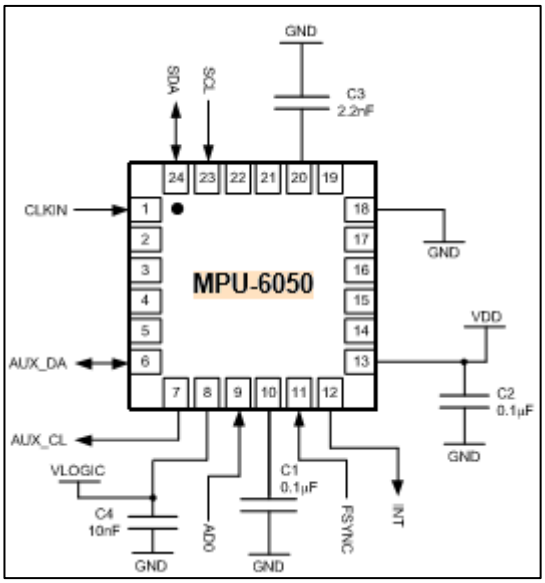


Figura 15 – Pinagem Mpu6050

Fonte: Datasheet (2013)

O MPU-6050 opera na faixa 2,375V-3,46V na fonte de alimentação, conforme especificado no quadro abaixo. Ele também fornece um pino de referência VLOGIC, que define os níveis lógicos de sua interface I2C (DATASHEET, 2013).

Partes do item	**MPU-6050**
VDD	2.375V-3.46V
VLOGIC	1.71V to VDD
Interface Serial Suportada	I^2C
Pino 8	VLOGIC
Pino 9	AD0
Pino 23	SCL
Pino 24	SDA

Quadro 1 – Pinagem MPU6050 e suas funções

Fonte: Datasheet (2013)

2.7 DISPOSITIVO *BLUETOOTH* HC-05

Segundo Sampei e Freitas (2014, p. 11) “A tecnologia e o padrão Bluetooth fornecem o meio para a substituição do cabo que conecta um dispositivo ao outro, através de um enlace universal de rádio de curto alcance”.

De acordo com a descrição apresentada no datasheet, o módulo HC-05 é um módulo Bluetooth SPP (*Serial Port Protocol*) projetado para configuração de conexão

serial sem fio. A comunicação entre o *bluetooth* e o microcontrolador se dá através do barramento UART. Ambos dispositivos, microcontrolador e módulo *bluetooth* devem estar configurados com a mesma taxa de transmissão para que seja possível a troca de dados. Além disso, deve-se fazer a instalação cruzada, ou seja, o RX do dispositivo máster deve ser conectado ao TX do dispositivo escravo, conforme demonstrado na Figura 16:

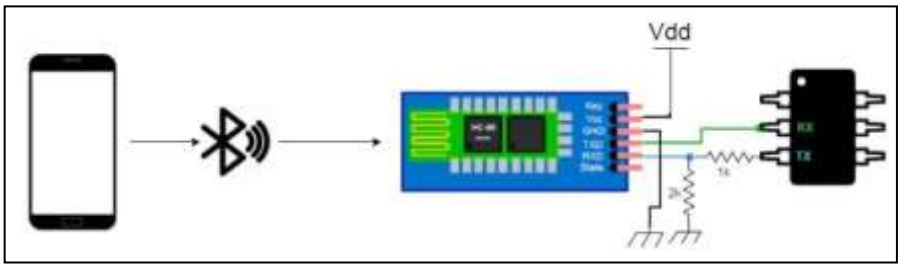


Figura 16 – Circuito de conexão *Bluetooth* com dispositivo HC-05
Fonte: Magdy (2019)

2.8 SISTEMA ANDROID E APLICATIVOS *MOBILE*

A utilização do celular só tem crescido durante os últimos anos. Segundo a ANATEL (Agência Nacional de Telecomunicações), no Brasil existem cerca de 228,5 milhões de acessos dos serviços móveis pessoais em julho de 2019. Comparando com a quantidade de pessoas que estão no Brasil, de acordo com o IBGE (Instituto Brasileiro de Geografia e Estatística), em 2019 há cerca de 210,6 milhões de habitantes, ou seja, há 1,08 celular por habitante no país. Essa quantidade de aparelhos se justifica pela comodidade da utilização em prol das atividades diárias, pois muitas ações podem ser realizadas diretamente do dispositivo móvel, tais como: movimentações bancárias, acesso a documentos digitais, envio e recebimento de mensagens ou e-mails, ligações, entretenimento, compra e venda, navegação na internet, entre outros.

Assim como há diversas ações possíveis de serem efetuadas através de um aparelho móvel, para que elas sejam executadas, existem vários aplicativos que foram desenvolvidos e disponibilizados para o uso. Neste projeto, está sendo utilizado o sistema operacional Android, o qual pode ser definido como segue:

> O Android é um sistema operacional móvel do Google e atualmente é líder mundial nesse segmento. No entanto, o sucesso do Android não se deve apenas a força do Google – por trás, do desenvolvimento de toda a plataforma estão gigantes do mercado de mobilidade, como fabricantes de celulares e operadoras. Esse grupo que ajuda no desenvolvimento da plataforma é chamado de OHA (*Open Handset Alliance*) e conta com nomes de peso como Intel, Samsung, LG, Motorola, Sony, Ericsson, HTC, Sprint Nextel, ASUS, Acer, Dell, Garmin, etc (...). (LECHETA, 2015, p.25).

Corroborando o autor Lecheta (2015) seguem dados sobre o sistema operacional Android e a participação do grupo que ajuda no desenvolvimento, de acordo com Cardle (2016):

> O Android é um sistema operacional móvel de código aberto. É uma variante do Linux, oferecendo ampla segurança, modularidade e produtividade no nível do dispositivo móvel. O Android é desenvolvido e mantido pela organização chamada OHA. A OHA foi criada em 2007, com o Google sendo seu principal membro. A OHA inclui muitas empresas de hardware e software de destaque. (p. 9).

Já foram lançadas diversas versões do Android. Ao criar o projeto é necessária atenção na escolha da versão mínima que a aplicação será compatível, para que o aplicativo não fique obsoleto. Um ambiente de desenvolvimento para aplicações é o Android Studio. Segundo Lecheta (2015, p. 28), “O Android é a primeira plataforma para aplicações móveis completamente livre e de código aberto (*open source*) (...)” No Android Studio é possível optar entre a linguagem de programação Kotlin ou a Java.

Na plataforma Android Studio foi utilizada a linguagem de programação orientada a objetos Java. O início do desenvolvimento da linguagem foi em 1990, por uma equipe da empresa *Sun Microsystems*. Antes de se chamar Java a linguagem era denominada Oak (Carvalho). Em 1994, a equipe estava desenvolvendo o projeto *Web Runner* que tinha como proposta a interação da linguagem com a *Web*. Contudo, ao registrar o projeto, o nome Oak já estava em uso, logo foi necessário muda-lo e a alteração foi para Java, o *Web Runner* também foi modificado para *HotJava*. De acordo com Cosmina (2018):

> Tudo começou em 1990, quando uma empresa americana que liderava a revolução na indústria de computadores decidiu reunir seus melhores engenheiros para projetar e desenvolver um produto que lhes permitisse se tornar um participante importante no novo mundo emergente da Internet. Entre esses engenheiros estava James Arthur Gosling, um cientista da computação canadense que é reconhecido como o "pai" da linguagem de programação Java. Levaria cinco anos de design, programação e uma renomeação (de Oak para Java por causa de problemas de marca registrada), mas finalmente em 1996, o Java 1.0 foi lançado para Linux, Solaris, Mac e Windows. (p. 1)

O Java tem dois pacotes básicos. Para o Android Studio é utilizado o pacote JRE (*Java Runtime Environment*), pois ele é o utilizado para executar as aplicações da plataforma Java, de acordo com Cardle (2017, p. 15) “O JRE é usado para executar software escrito na linguagem de programação Java, enquanto o JDK é utilizado para o desenvolvimento de software Java".

O paradigma da programação são as diversas formas de desenvolver um código e ele está divido em três principais vertentes: programação estruturada, programação funcional e programação orientada a objetos. A linguagem Java tem sua base na programação orientada a objetos. Segundo Barros (2011):

> A metodologia de Orientação a Objetos constitui um dos marcos importantes na evolução mais recente das técnicas de modelagem e desenvolvimento de sistemas de computador, podendo ser considerada um aprimoramento do processo de desenhar sistemas. Seu enfoque fundamental está nas abstrações do mundo real.”. (p. 85)

Barros (2011) ainda relata que, os princípios básicos da programação orientada a objetos são: abstração, encapsulamento, herança e polimorfismo.

2.9 INTELIGÊNCIA ARTIFICIAL

A Inteligência Artificial (IA) já é realidade e está presente nos smartphones, veículos, e-commerce, comércio, entre outras atividades. Essa expansão da utilização da Inteligência Artificial, conforme Russell e Norvig (2010), se deve aos estudos da função dos neurônios, análise da lógica proposicional e a teoria da computação de Turing realizados por Warren McCulloch e Walter Pitts em 1943. Porém, a Inteligência Artificial que vivenciamos atualmente nasceu no workshop de 1956 em Dartmouth e, segundo Russell e Norvig (2010, p. 17), estes estudos se baseavam “...na conjectura

de que todos os aspectos de aprendizado ou qualquer outro recurso da inteligência pode, em princípio, ser descrito com tanta precisão que pode ser feita uma máquina para simulá-la...".

Mas enfim, o que é Inteligência Artificial? De acordo com Nagy (2018, p. 45) "é uma ciência usada para construir inteligência usando soluções de hardware e software". Já Negnevitsky (2005, p. 219) conceitua como "capacidade de um sistema adaptar seu comportamento a um ambiente em constante mudança".

Para aplicar a Inteligência Artificial pode-se utilizar ou não técnicas de *Machine Learning* (Aprendizado de Máquina) e conforme Nagy (2018, p. 45) "é um campo de estudo preocupado em oferecer computadores à capacidade de aprender sem ser explicitamente programado". Para que os computadores possam aprender são necessários dados, pois segundo Russell (2018, p. 8), "O aprendizado de máquina é a prática de programar computadores para aprender com os dados.".

O aprendizado de máquina pode ser classificado conforme a interação humana, onde de acordo Géron (2017), podem ser: supervisionado, não-supervisionado, semi-supervisionado e por reforço. No aprendizado supervisionado conforme Nagy (2018, p. 58) "é baseado em dados rotulados e funções inferidas de dados de treinamento".

2.9.1 Métodos de Classificação

Os sistemas de aprendizado podem ser classificados em quatro categorias, conforme Russel (2018, p. 13), "supervisionado, não-supervisionado, semi supervisionado e aprendizado por reforço."

No aprendizado supervisionado procura-se prever rótulos desconhecidos a partir de comparações com base de dados treinados que de acordo com Russel (2018, p. 14) "os dados que você alimenta no algoritmo, com a solução desejada, são chamados de rótulos".

Neste tipo de aprendizado o ser humano mapeia as variáveis que os classificam (rotulam) para obter uma saída. Após esse mapeamento é realizado o treinamento e logo após submete novos dados para comparar com o treinamento para encontrar a classificação correta. Os algoritmos supervisionados mais importantes: K vizinhos mais próximos (KNN – K-*nearest neighbors*); Regressão Linear; Rede Neural;

Máquina de Vetores de Suporte; Regressão Logística; Árvores de Decisão e Florestas aleatórias

Na aprendizagem não-supervisionada o próprio algoritmo cria os *cluster* ou associações sem a intervenção do ser humano sendo utilizado para obter esclarecimentos através dos dados não visualizados anteriormente e de acordo com Muller e Guido (2017, p. 131) “o algoritmo não supervisionado solicita dados de entrada e extrai os conhecimentos desses dados”. Os algoritmos não-supervisionados mais importantes: K-médias; Análise hierárquica cluster; Eclat; Apriori; Análise de Componentes Principais do Kernel (PCA); Distribuição T; Análise de Componentes Principais.

Enquanto que no aprendizado por reforço, conforme Russel (2018, p. 17), “(...)o sistema de IA observará o ambiente, executará determinadas ações e, em seguida, recebe t recompensas em troca. Com esse tipo, o agente deve aprender sozinho.” De acordo com Sutton e Barto (2018, p.1) “O aprendizado por reforço é aprender o que fazer - como mapear situações para ações – para maximizar um sinal de recompensa numérico”.

2.9.1.1Método de classificação KNN (K- Vizinhos mais próximos)

O algoritmo K-*Nearest Neighbor* é um método de classificação supervisionado, que segundo Rothman (2018, p. 169) “porque usa rótulos fornecidos para treinar seu algoritmo”. Ainda considerando Rothman (2018, p. 171) “o KNN irá calcular a distância euclidiana entre o ponto P e todos os outros pontos P usando a fórmula da distância Euclidiana”. A distância Euclidiana pode ser calculada conforme a equação abaixo:

$$P(x, y) = \sqrt{(x1 - x2)^2 + (y1 - y2)^2} \quad (4)$$

Onde x e y, correspondem às coordenadas de cada ponto, onde estão sendo calculadas a distância Euclidiana.

Após o cálculo da distância entre todos os pontos, o algoritmo localiza os pontos mais próximos e infere a classificação do ponto atribuindo o rótulo que teve mais frequência entre todos os pontos. De acordo com Coppin (2018) no KNN “os dados de treinamento são armazenados e, quando uma nova instância for encontrada, ela

será comparada aos dados de treinamento para encontrar seus vizinhos mais próximos”. (COPPIN, 2010, p. 247).

2.9.1.2 Redes neurais artificiais

As redes neurais artificiais, conforme Negnevitsky (2005, p. 166), é um “...modelo de raciocínio baseado no cérebro humano”, composto por sinais de entrada, camadas de entrada, camadas ocultas, sinais de saída. A Figura 17 abaixo, adaptada de Negnevitsky (2005), apresenta o comparativo entre rede neural biológica (à esquerda) e rede neural artificial (à direita).

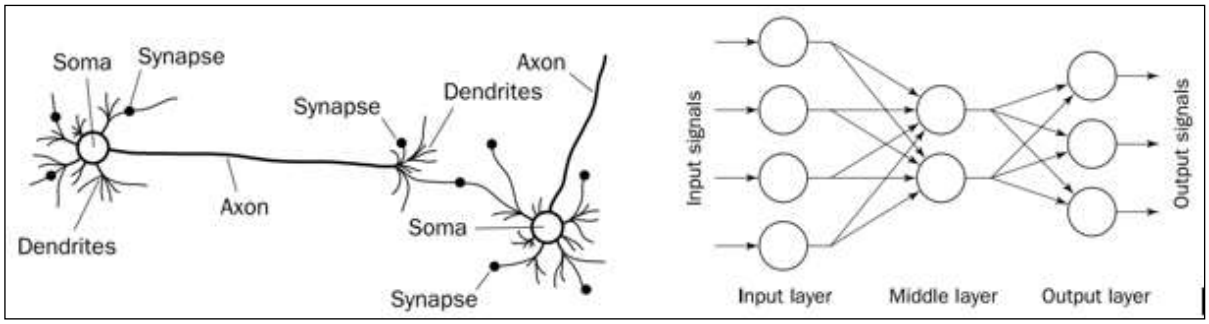


Figura 17 – Rede neural biológica x Rede neural artificial
Fonte: Adaptado de Negnevitsky (2005)

Portanto, conforme a ilustrado acima, em uma rede neural artificial são inseridos os dados (sinais de entrada), sendo processados pelas camadas ocultas e geram um resultado (sinais de saída).

3 MATERIAIS E MÉTODOS

O presente capítulo traz as informações acerca dos materiais utilizados para o desenvolvimento da aplicação e do protótipo. Além disso, explana sobre a metodologia e tecnologias adotadas em todas as fases do projeto, detalhando as principais características e funcionalidades do sistema e finalizando com a etapa de testes realizados.

3.1 MATERIAIS

Para ser possível analisar e classificar as letras, a partir da leitura dos movimentos da mão, foram necessários alguns materiais específicos que serão dispostos neste tópico, apresentando também como foram feitas suas aplicações no projeto.

3.1.1 Luva para Sensoriamento

Para acoplar os sensores necessários para aquisição de dados, utilizou-se uma luva da marca Kipsta® do tipo Keepwarm (mantém o calor corporal) tamanho 10 ilustrada na Figura 18, a qual é fabricada em tecido poliéster (93%) e elastano (7%) o que permitiu um encaixe confortável na mão, sendo essa característica essencial para melhor precisão na leitura dos movimentos.

Figura 18 – Luva Kipsta® *Keepwarm* com sensores flexíveis
Fonte: dos Autores (2019)

3.1.2 Sensores Flexíveis e MPU-6050

A leitura dos movimentos dos dedos foi feita através de sensores flexíveis de 2,2 polegadas fabricados pela Spectrasymbol®. Estes sensores são resistivos e para fazer a medição da flexão com esse tipo de sensor é necessário inseri-lo em um circuito divisor de tensão conforme apresentado na Figura 13. Neste projeto foram utilizados resistores de 47 kOhm para o circuito divisor de tensão conforme o diagrama esquemático na Figura 20. A fixação dos sensores foi feita com costura e fita dupla face, posicionados de modo a ler ao menos dois graus de flexão de cada dedo conforme Figura 18. Devido ao tamanho da área ativa dos sensores flexíveis que é de 5,5 cm, apenas os movimentos de dobra entre as falanges distal, média e proximal são lidos, descartando-se a flexão entre os ossos metacarpais e falange proximal.

O sensor MPU-6050 utilizado para leitura do posicionamento linear dos eixos X, Y e Z, foi posicionado no dedo indicador para ser possível diferenciar algumas letras que possuem configurações de mão semelhantes, porém com pequenas diferenças na posição ou com movimentos diferentes, como por exemplo: as letras U e V que se diferem apenas pela distância entre os dedos indicador e médio. O Quadro 2 a seguir mostra as localizações dos sensores na luva:

	Localização na Luva
Sensores Flexíveis	1 sensor no dedo Polegar
	1 sensor no dedo Indicador
	1 sensor no dedo Médio
	1 sensor no dedo Anelar
	1 sensor no dedo Mínimo
MPU6050	1 sensor no dedo Indicador

Quadro 2 – Disposição dos sensores na luva

Fonte: dos Autores (2019)

3.1.3 Módulo B*luetooth* HC-05

Como a comunicação será diretamente com um *smartphone*, foi utilizado o envio serial via *Bluetooth* HC-05 por ser um módulo sem fio (*wireless)* muito utilizado para pareamento e comunicação entre dispositivos. O diagrama esquemático pode ser consultado na Figura 20 deste documento, o qual ilustra a conexão do microcontrolador com o módulo HC-05 para comunicação serial.

3.1.4 Microcontrolador PIC18F4550

O projeto demandou o uso de 5 pinos analógicos com conversor analógico-digital integrado (conversor AD), pinos de comunicação serial assíncrona RX e TX, além da disponibilidade de pinos para comunicação via protocolo I^2C para leitura do sensor acelerômetro/giroscópio MPU 6050 e pinos digitais para envio de dados de resposta em um LCD 16x2 ou LEDs para testes iniciais. A partir disso, a aplicação foi desenvolvida com o uso do Microcontrolador PIC 18F4550 que atendeu todas as necessidades do projeto.

3.2 HARDWARE

Todos os dados obtidos na leitura dos sensores acoplados na luva são processados pelo microcontrolador PIC 18F4550, o qual coleta, pré-processa e então converte os dados iniciais gerados em formato inteiro (sensores flexíveis) e real (MPU-6050) para variáveis do tipo *string* (vetor de caracteres ou *char),* para então serem enviados serialmente via *bluetooth* a cada 0.8 ms aproximadamente.

O circuito conta com um LED verde que pisca intermitentemente para indicar o envio de dados, botão de *reset* do microcontrolador através do pino MCLR (*Master Clear*), suporte para pilha tipo A23 de 12 V com regulagem de tensão para 5 V por meio de um LM7805 e também um botão ON/OFF. A Figura 19 mostra a placa final do circuito:

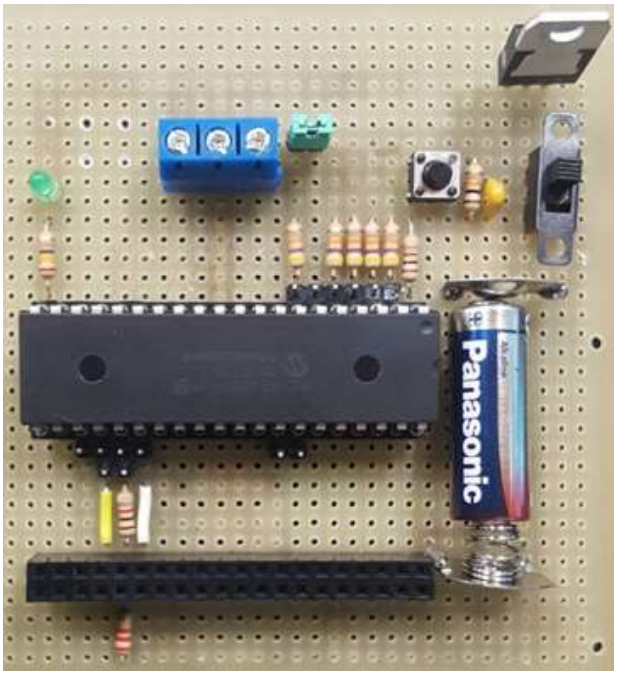


Figura 19 – Placa do Circuito Final
Fonte: dos Autores (2019)

A Figura 20 apresenta o diagrama esquemático do circuito feito no software Proteus®, mostrando as conexões feitas no microcontrolador e também os componentes eletrônicos utilizados no projeto. O Proteus® não possui sensores flexíveis em sua lista de componentes, por isso foram utilizados sensores LDR no diagrama por possuírem o mesmo princípio de funcionamento dos sensores flexíveis

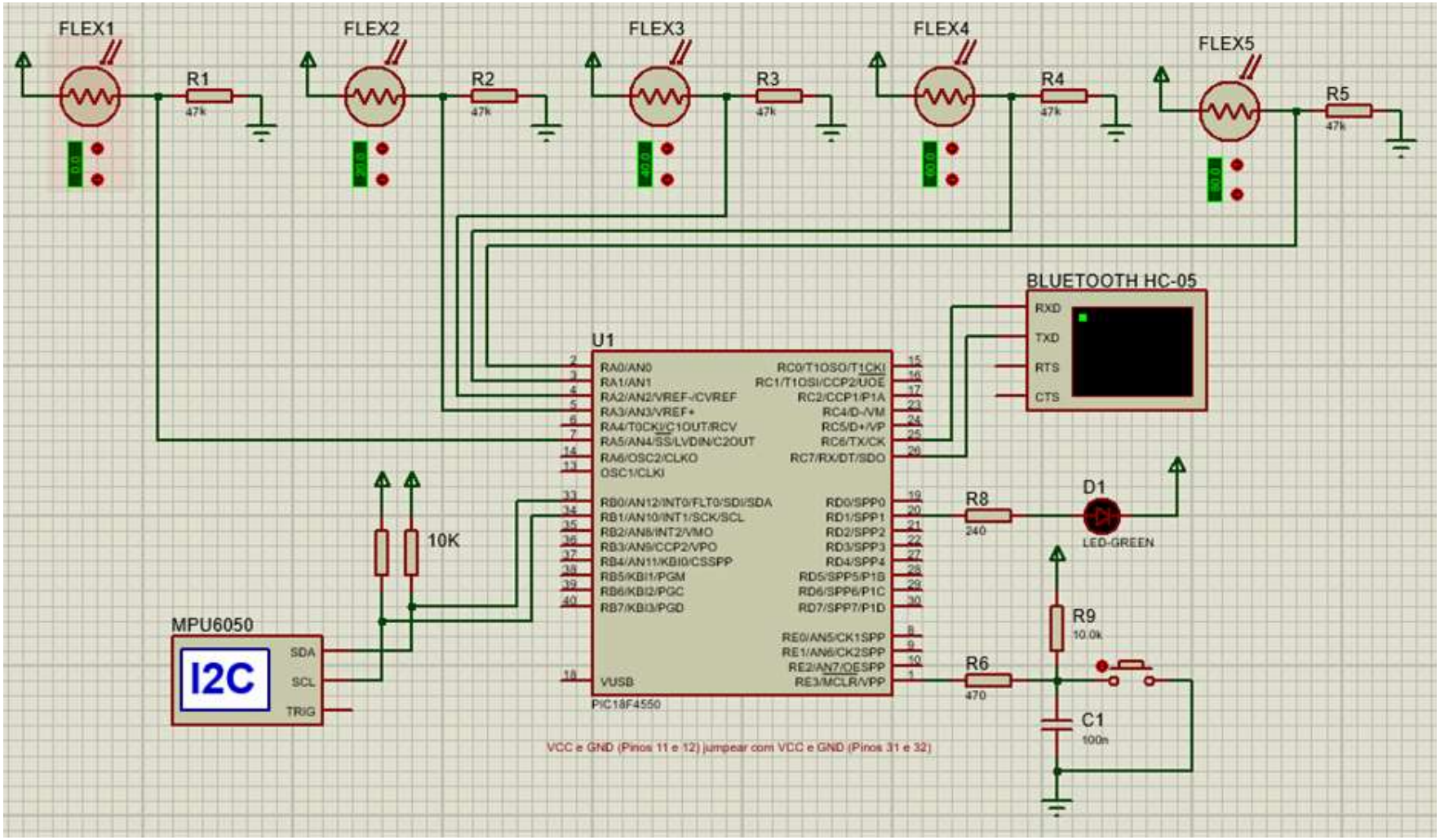


Figura 20 – Diagrama Esquemático do Circuito no Proteus®

Fonte: dos Autores (2019)

3.2.1 Normalização de Leitura dos Sensores Flexíveis

Para uma maior precisão das leituras dos sensores flexíveis, é aplicada uma normalização por média aritmética conforme equação a seguir:

$$y[n] = \frac{1}{M}\sum_{j=0}^{M-1} x[i+j] \quad (5)$$

Onde:

y[n] é o sinal de saída normalizado;

M é o número de amostras;

x[ ] são os sinais de entrada ou amostras.

A Figura 21 mostra o comportamento da leitura de um sensor flexível parado em posição plana, sem e com a aplicação da normalização, iniciando pela amostragem sem tratamento:

Pode-se perceber na Figura 21 (A) que a leitura oscila mesmo com o sensor não sendo flexionado. A Figura 21 (B) mostra o sinal de saída após aplicação de média aritmética com uso de cinco amostras, onde o sinal já apresenta menor oscilação:

Na Figura 21 (C) nota-se que quanto maior o número de amostras utilizadas, maior a estabilidade de leitura, neste exemplo com dez amostras, porém há um aumento no tempo de processamento pela necessidade de se coletar mais leituras a cada sinal de saída.

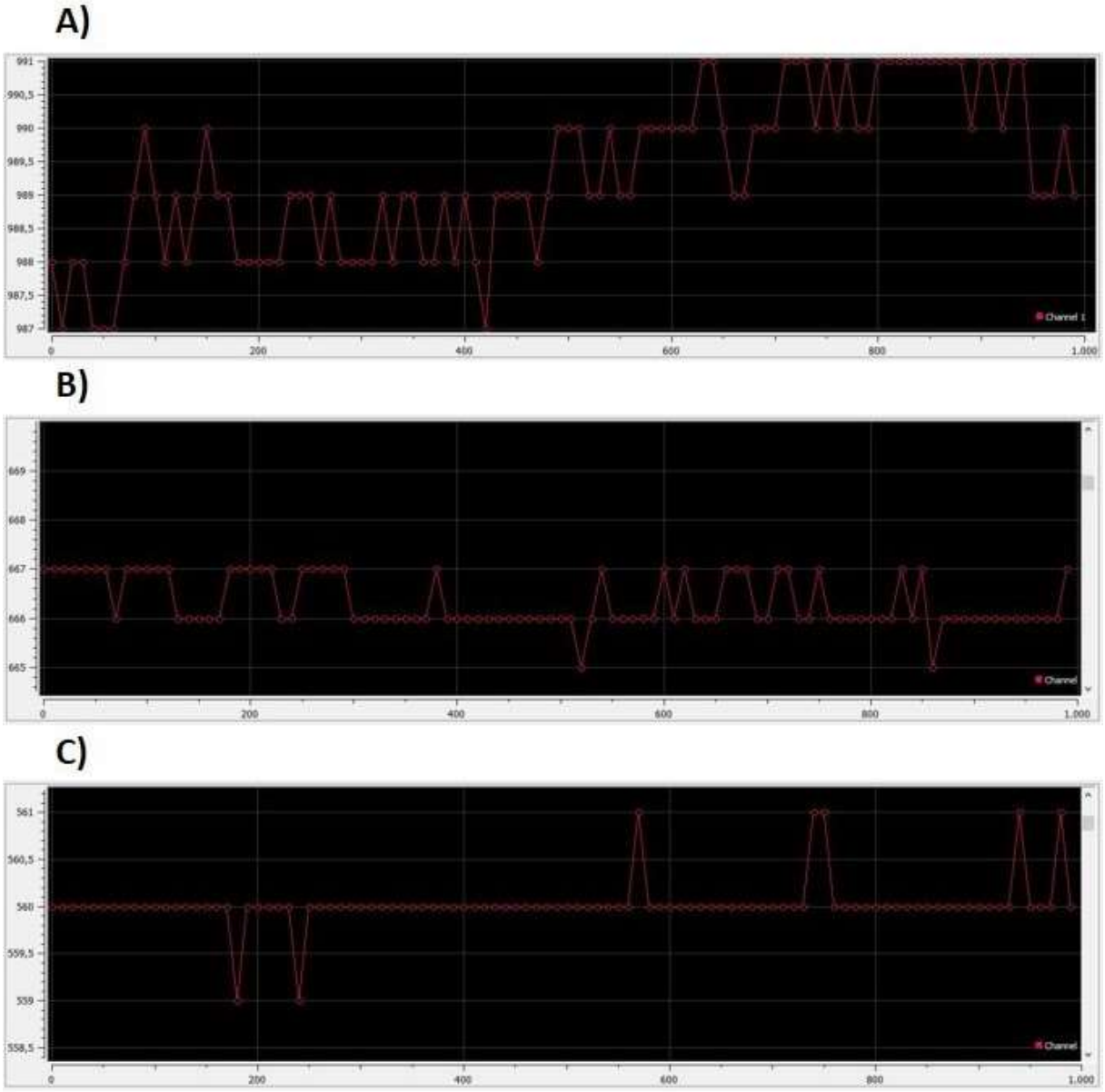


Figura 21 – Comparativo entre sinais com e sem normalização
Fonte: dos Autores (2019)

Nas leituras dos sensores flexíveis da luva, foram aplicadas médias aritméticas com cinco amostras, o que retorna uma leitura com estabilidade e tempo de processamento satisfatórios.

3.3 SOFTWARE

A versão determinada ao criar este projeto foi a API 19: Android 4.4 (KitKat) que tem o alcance de cerca de 95,3% dos dispositivos, conforme ilustrado na Figura 22 e a linguagem selecionada foi o JAVA, por ser mais difundida.

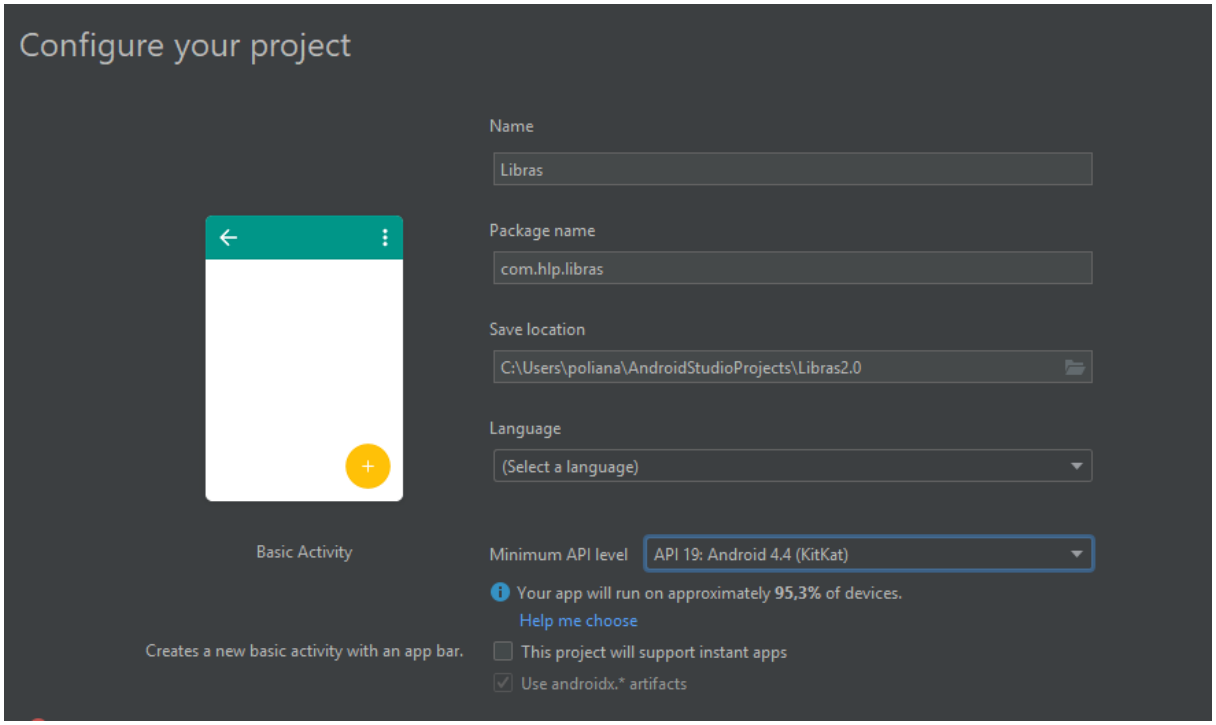


Figura 22 – Versão do Android utilizada

Fonte: dos Autores (2019)

O primeiro item inserido no aplicativo foi o botão “Iniciar” na primeira tela (*activity*) e posteriormente foi adicionada a função ativar o *bluetooth* do aparelho conforme ilustrado na Figura 30. Ao clicar na opção “Permitir”, apresentará a mensagem “O *bluetooth* foi ativado”. Caso opte por recusar, o aplicativo é encerrado e apresentará a mensagem: “O *bluetooth* não foi ativado, o *app* será encerrado”. Em caso de ativação do *bluetooth*, o aplicativo entra na tela secundária onde são mostradas as letras ou números classificados

3.3.1 Diagrama Casos de Uso

A seguir, é apresentado o diagrama de casos de uso referente ao aplicativo desenvolvido na IDE Android Studio, que serve de interface com o usuário para retornar a letra reconhecida.

Figura 23 – Diagrama de Casos de Uso

Fonte: dos Autores (2019)

3.3.2 Requisitos Funcionais e Não Funcionais

Nesta seção são apresentados os quadros referentes aos requisitos funcionais e não funcionais do aplicativo desenvolvido:

Tipo de Requisito: Comunicação			
ID	**Descrição do requisito**	**Importância**	**Fonte**
RF 1.0.0	lista de dispositivos pareados do *mobile*.	Essencial	Usuário
RF 2.0.0	ativar o *bluetooth* do aparelho móvel ao iniciar o aplicativo	Essencial	Usuário
RF 3.0.0	conectar o aplicativo ao hardware	Essencial	Usuário

Quadro 3 – Requisitos Funcionais - Comunicação

Fonte: dos Autores (2019)

Tipo de Requisito: Mostrar resultado			
ID	**Descrição do requisito**	**Importância**	**Fonte**
RF 4.0.0	Receber *strings* enviadas dos sensores da luva tradutora	Essencial	Aplicativo
RF 5.0.0	Comparar os dados recebidos com os dados Treinados pelo KNN	Essencial	Aplicativo
RF 6.0.0	Mostrar o resultado na tela do aplicativo.	Essencial	Aplicativo

Quadro 4 – Requisitos Funcionais – Mostrar Resultados

Fonte: dos Autores (2019)

Tipo de Requisito: Instalações do ***back-end*.**			
ID	**Descrição do requisito**	**Importância**	**Fonte**
RNF 1.0.0	Instalação do JDK	Essencial	Tecnologia
RNF 1.0.1	Preferência pela versão 8 (Java *Development* Kit 8)	Essencial	Tecnologia

Quadro 5 – Requisitos Não Funcionais – Instalação do Back-End

Fonte: dos Autores (2019)

Tipo de Requisito: Instalações do ***front-end*.**			
ID	**Descrição do requisito**	**Importância**	**Fonte**
RNF 2.0.0	Instalação do Android Studio	Essencial	Tecnologia
RNF 2.0.1	Preferência pela versão 3.5.1	Essencial	Tecnologia

Quadro 6 – Requisitos Não Funcionais – Instalação do Front-End

Fonte: dos Autores (2019)

Tipo de Requisito: Hardware			
ID	**Descrição do requisito**	**Importância**	**Fonte**
RNF 3.0.0	O aplicativo é compatível com *smartphones* e *tablets*	Essencial	Tecnologia
RNF 3.0.1	O aplicativo é compatível com a versão do *Android* 4.4 (KitKat) ou superior.	Essencial	Tecnologia

Quadro 7 – Requisitos Não Funcionais – Hardware

Fonte: dos Autores (2019)

3.4 METODOLOGIA

Primeiramente, realizou-se uma pesquisa bibliográfica exploratória para embasamento teórico acerca dos principais pontos de cunho social que envolvem o projeto, bem como para conhecimento acerca dos trabalhos já realizados, componentes e ferramentas necessários para sua execução.

Além disso, foi necessário fazer o levantamento de requisitos funcionais e não funcionais do aplicativo *mobile*, indicando quais as funcionalidades, as finalidades, as premissas e restrições, dentre outros itens pertinentes.

Posteriormente foi realizado um estudo sobre a datilologia (“soletrar palavras” com os sinais manuais usando os caracteres alfanuméricos), para então iniciar a adaptação da primeira versão do protótipo, o qual utilizava a plataforma Arduino® e atualmente aplica o microcontrolador PIC 18F4550 da Microchip®, programado em linguagem C, na plataforma de desenvolvimento integrado MPLAB com compilador XC8, ambos fornecidos pela Microchip®.

A análise qualitativa dos dados é feita através da precisão de leitura dos sensores, média de leitura e acurácia do classificador utilizado com base em cada configuração de letra e número em Libras. Para tanto, foram geradas planilhas de amostras das leituras dos sensores e dispostas em gráficos.

Por fim, as planilhas definitivas geradas foram utilizadas como base de treinamento para o classificador KNN, implementado diretamente no aplicativo desenvolvido, para possibilitar a resposta ao usuário na própria aplicação.

3.5 PROCEDIMENTOS DE TESTE

Os primeiros testes das leituras dos sensores flexíveis foram realizados utilizando o software de simulação Proteus®, com as respostas sendo exibidas via Terminal Serial Virtual, LCD 16x2 e também por LEDs conforme Figura 24:

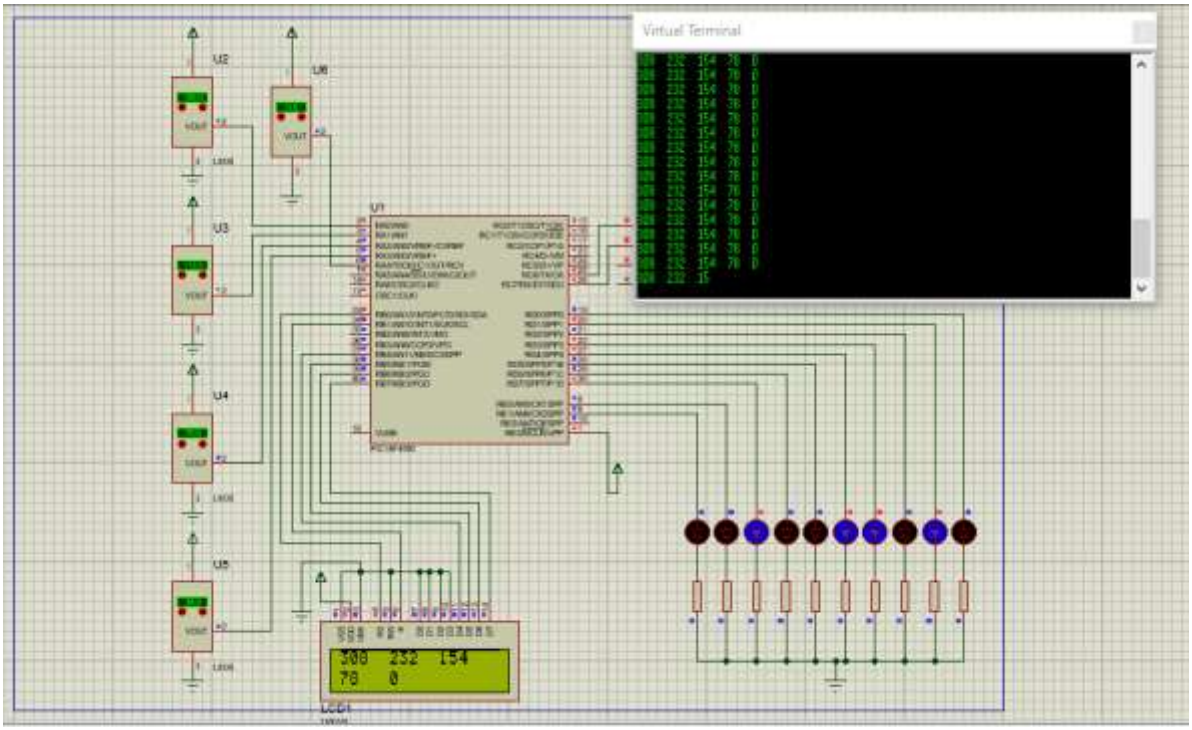


Figura 24 – Teste de leitura dos Sensores Flexíveis
Fonte: dos Autores (2019)

Como no simulador utilizado não existe sensores flexíveis na lista de componentes, foram utilizados resistores variáveis para testes iniciais para ser possível visualizar as conversões no ADC.

Posteriormente, os testes de leitura passaram a ser obtidos diretamente do *hardware* desenvolvido, utilizando o emulador de terminal serial “Putty” para exibir tanto as leituras dos sensores flexíveis, quanto às leituras dos eixos X, Y e Z do acelerômetro do MPU-6050 conforme visto na Figura 25:

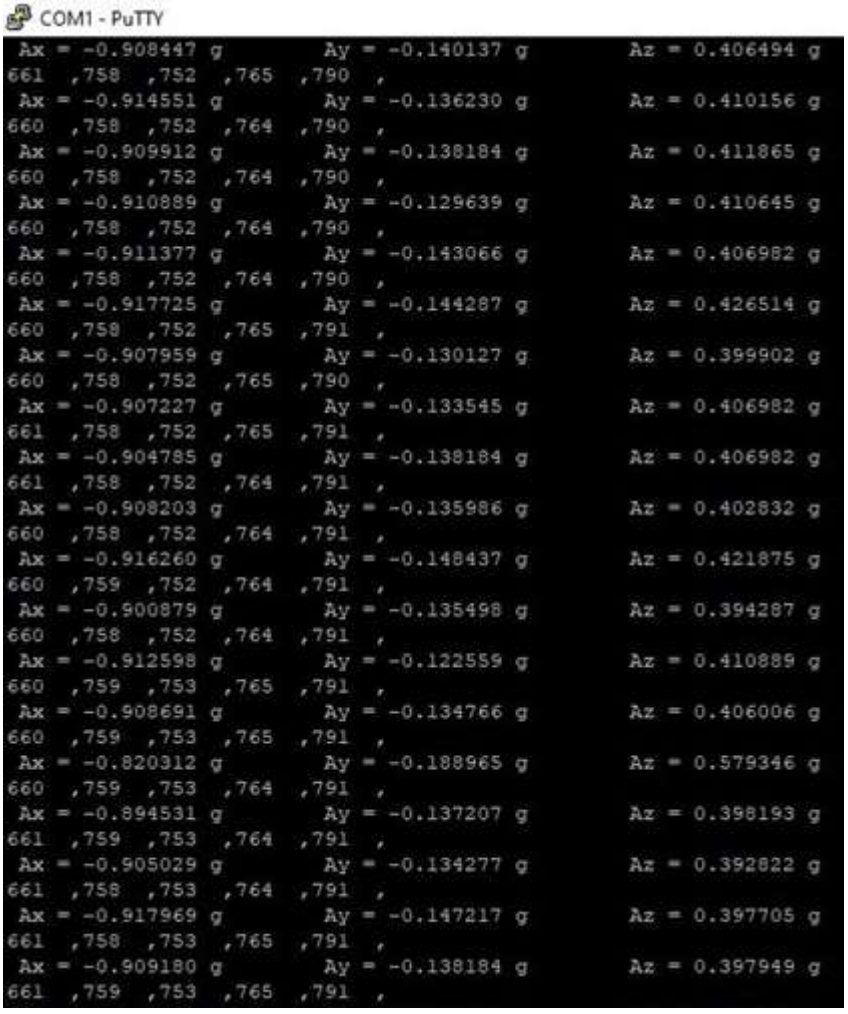


Figura 25 – Leituras dos Sensores Flexíveis e MPU6050 via Putty

Fonte: dos Autores (2019)

Com isto finalizou-se os testes de aquisição de dados e foram geradas as planilhas de leitura das letras e números em Libras para análise da precisão e também para servir de base de treinamento para o classificador utilizado, por meio de arquivos no formato .csv gerados pelo “Putty” na seguinte sequência:

Figura 26 – Fluxograma de Aquisição de dados

Fonte: dos Autores (2019)

.

As Figuras 27 e 28 mostram um exemplo de gráficos gerados a partir das planilhas com cem amostras de leitura da letra A, através da luva de dados:

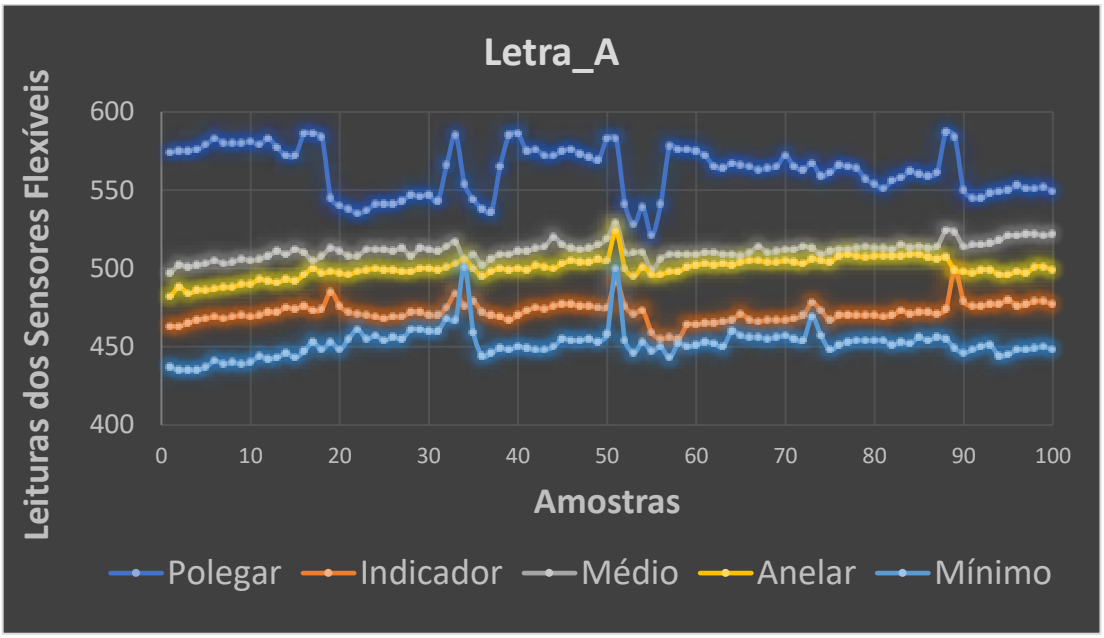


Figura 27 – Amostras de leitura da Letra A: Sensores Flexíveis

Fonte: dos Autores (2019)

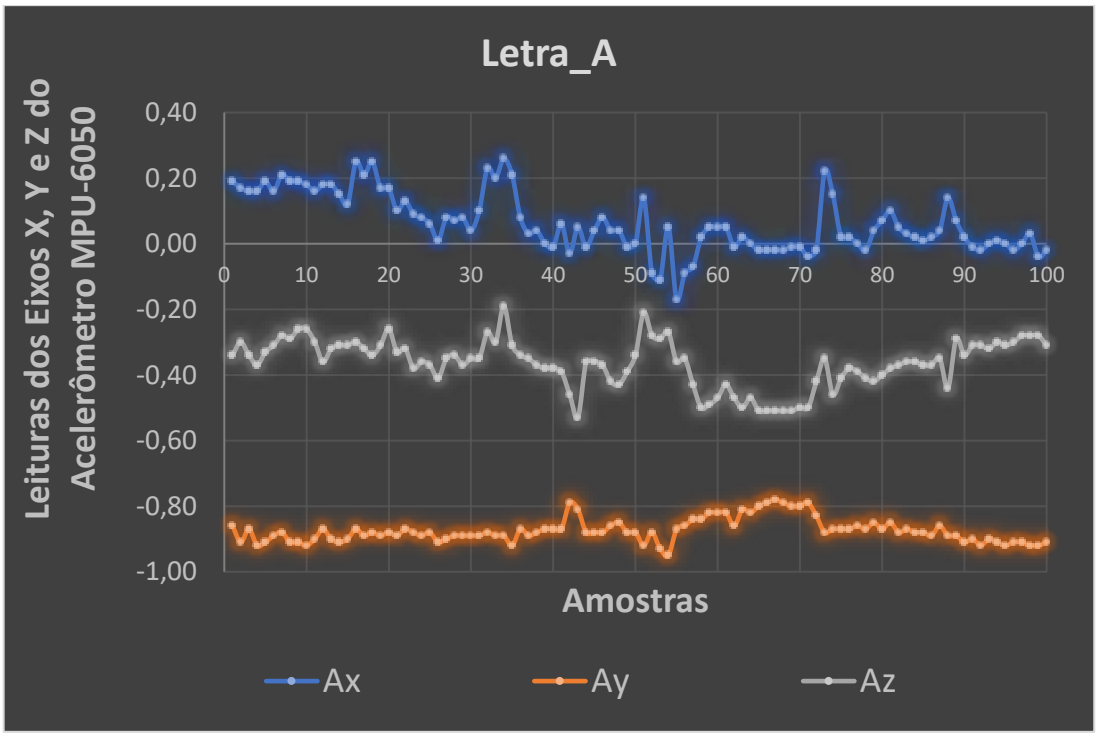


Figura 28 – Amostras de leitura da Letra A: MPU6050

Fonte: dos Autores (2019)

Na etapa seguinte, utilizando as planilhas geradas, foi realizado o treinamento do classificador KNN e de uma rede neural artificial com 8 neurônios de entrada, uma camada oculta de 22 neurônios e na camada de saída 36 neurônios para fins de comparação de acurácia. Para tanto, a configuração de mão e movimento correspondente a cada letra e número em Libras foi feito utilizando a luva na mão direita, obtendo-se o padrão de leitura de cada caractere. Os dados então foram enviados via porta serial do computador, conectado através de um módulo FTDI232 (conversor usb serial) e foram salvas em arquivos do tipo .csv por meio do emulador de Terminal Serial Putty, formando a base de dados das leituras dos sensores conforme ilustrado na Figura 29:

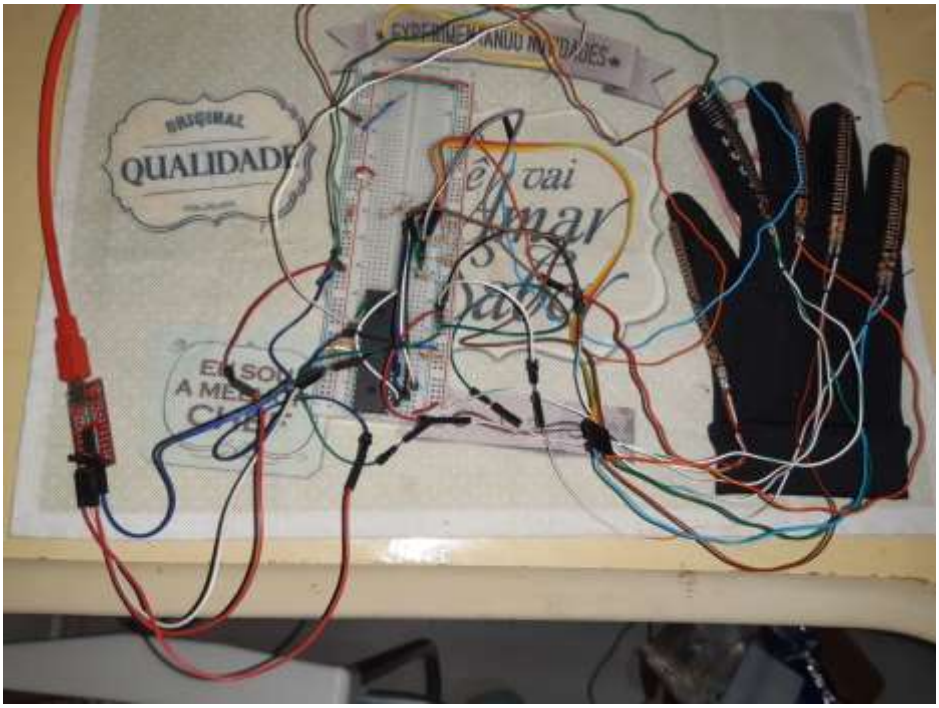


Figura 29 – Envio de dados com uso do FTDI232
Fonte: dos Autores (2019)

Os resultados de acurácia obtidos estão dispostos no item Resultados e Discussões deste projeto.

Em relação ao aplicativo desenvolvido o primeiro passo para iniciar os testes foi efetuar a ativação do *bluetooth* do aparelho, sendo necessária a inclusão de duas linhas de permissões no código fonte. Na figura abaixo é mostrado a tela com a solicitação que será apresentada ao usuário:

Figura 30 – Ativação do *bluetooth*
Fonte: dos Autores (2019)

Com a permissão concedida será apresentada a mensagem “O *bluetooth* foi ativado”, estando pronto para receber as leituras efetuadas pela luva tradutora para então compará-las com o treinamento realizado pelo KNN e, por fim, mostrar na tela do celular. O fluxograma disposto a seguir mostra a sequência desse funcionamento:

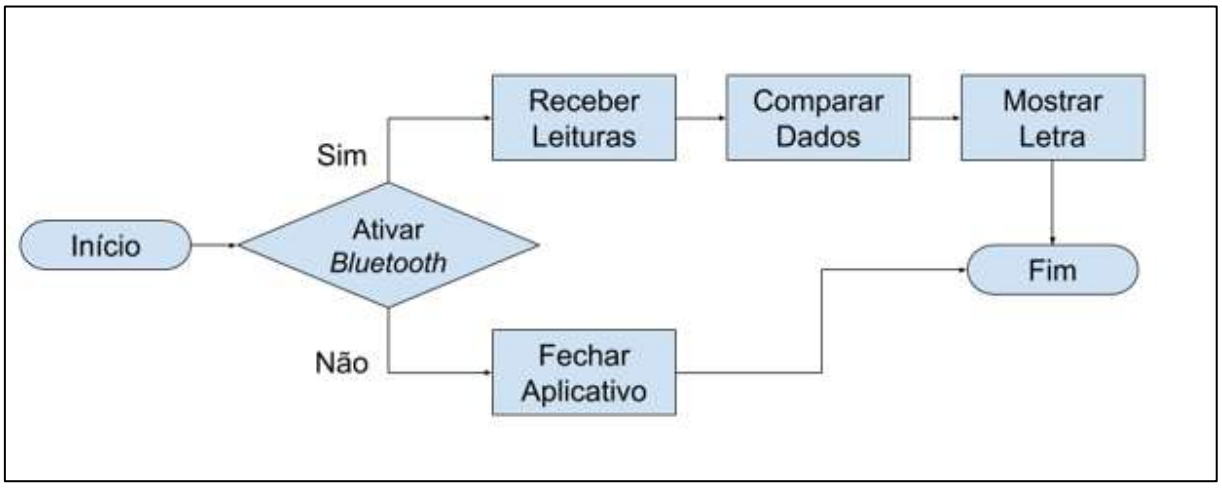


Figura 31 – Processos do aplicativo
Fonte: dos Autores (2019)

4 RESULTADOS E DISCUSSÕES

Atualmente, é possível obter a leitura dos sinais correspondentes à flexão dos cinco dedos e também dos três eixos acelerômetro do MPU-6050 com a utilização do microcontrolador PIC 18F4550. O tratamento de sinal é aplicado via programação, retornando os valores das leituras em formato *string*, as quais são enviadas a cada 0,8 segundos aproximadamente para o aplicativo no celular via *bluetooth*. A partir desses dados, o classificador KNN seleciona as letras ou números correspondentes à configuração e movimento da mão para ser apresentado o resultado no aplicativo, conforme sequência apresentada na Figura 32:

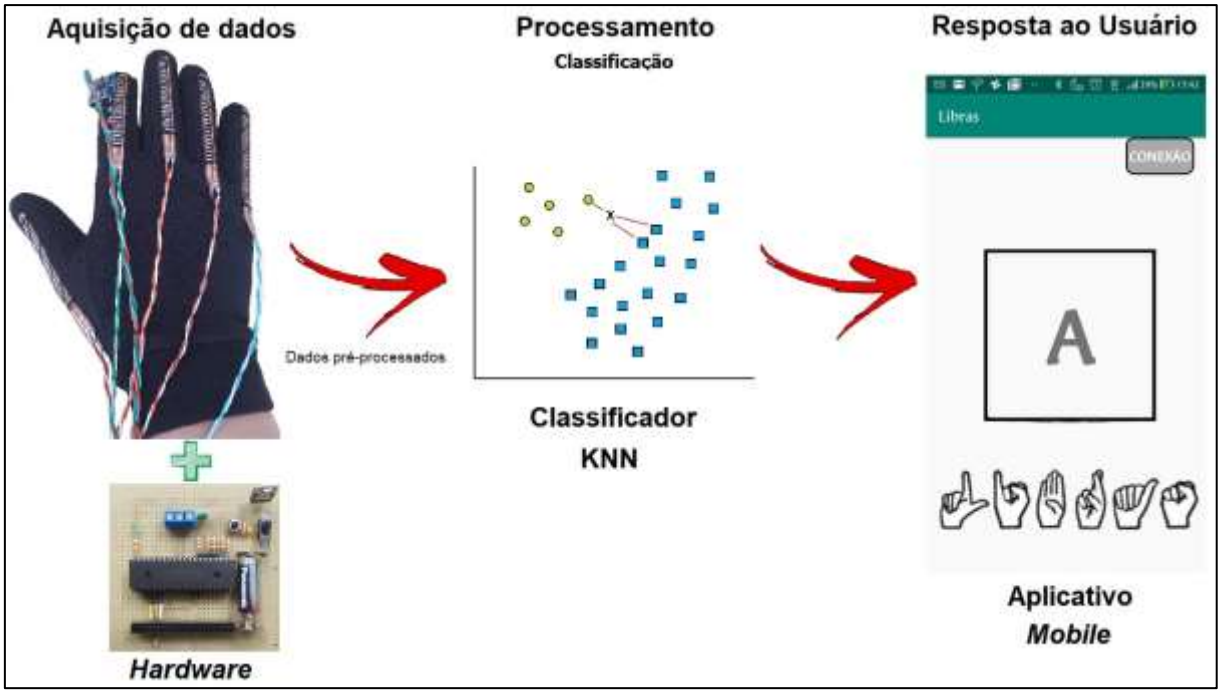


Figura 32 – Módulos do Projeto

Fonte: dos Autores (2019)

Na Figura 33 é destacada a interface principal do aplicativo, onde efetivamente são apresentados os resultados finais de todo o processamento contido no projeto:

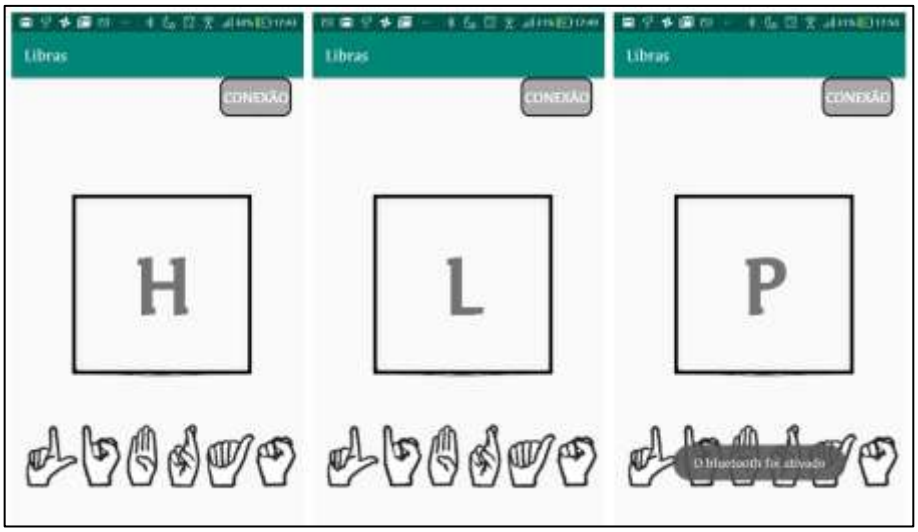


Figura 33 – Exemplos de resposta no aplicativo

Fonte: dos Autores (2019)

4.1 RESULTADOS COM CLASSIFICADOR KNN

Utilizou-se uma base de dados com aproximadamente 100 leituras dos sensores para cada letra do alfabeto em Libras (de A a Z) e números (0 a 9), totalizando em 3.881 leituras. Dessas leituras foram utilizados 70% (2.720) da base para treinamento e 30% (1.161) para o teste conforme Quadro 8 a seguir:

	Amostras	Aplicação (%)
Treinamento	2.720	70%
Teste	1.161	30%
Total	**3.881**	**100%**

Quadro 8 – Coleta de dados para treinamento KNN

Fonte: dos Autores (2019)

Considerando a parte da amostra que foi utilizada para testes (1.161) obteve-se os seguintes resultados:

	Amostras	Porcentagem de acerto e erro
Acertos	1.098	95%
Erros	63	5%
Total	**1.161**	**100%**

Quadro 9 – Acurácia de predição do KNN

Fonte: dos Autores (2019)

Conforme pode ser observado no Quadro 9, a predição obteve uma acurácia de 95%, ou seja, do total de amostras utilizadas para os testes, a predição do KNN classificou corretamente 95% dos dados.

A Figura 34 apresenta o detalhamento da predição do algoritmo através da matriz de confusão, onde os dados na linha diagonal correspondem as amostras classificadas corretamente, enquanto as amostras dispersas referem-se as amostras classificadas incorretamente.

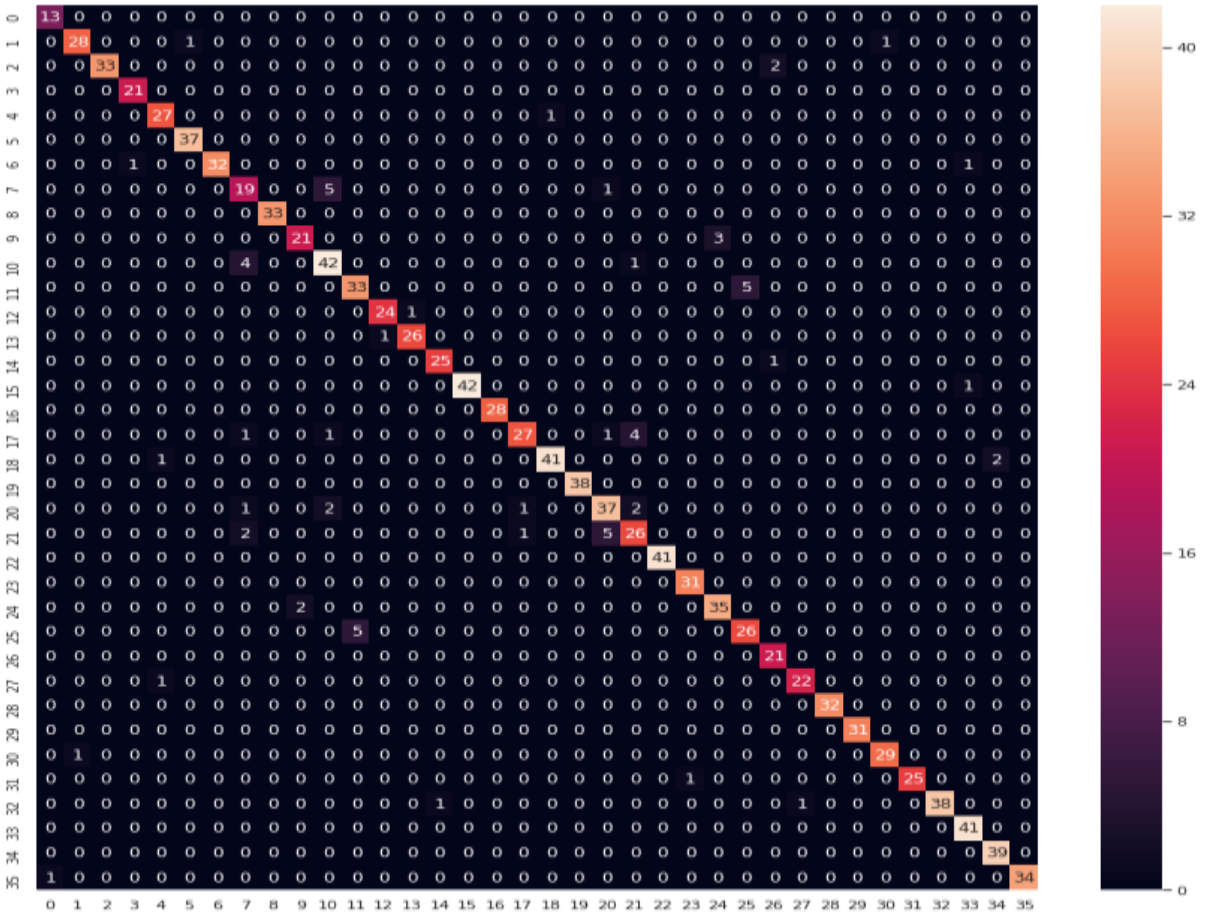


Figura 34 – Matriz de Confusão do KNN

Fonte: dos Autores (2019)

Além da acurácia, utilizamos outras métricas comuns utilizadas em *machine learning* as quais citamos: *precision*, *recall* e *f1-score*. O *precision* analisa a proporção das classificações positivas, se foram realmente corretas, ou seja, baseia-se apenas nos verdadeiros positivos (VP) e falsos positivos (FP) da seguinte forma:

$$precision = (VP)/(VP + FP) \quad (6)$$

Já o *recall* verifica a proporção de verdadeiros positivos que foram identificados corretamente, utilizando os verdadeiros positivos (VP) e falsos negativos (FN) conforme segue:

$$recall = (VP +)/(VP + FN). \quad \text{.....(7)}$$

E por fim, o *f1-score* refere-se ao balanço entre o *precision* e o *recall.* Para facilitar a visualização, abaixo é demonstrado as fórmulas de cada métrica:

$$f1 - score = \ 2 * ((precision * recall)/(precision + recall)) \quad (8)$$

Diante disso, a Figura 35 mostra o resultado geral referente ao treinamento do algoritmo KNN confirmando a acurácia de 95%, demonstrando a eficiência desse modelo simples de classificação.

```
In [1]: runfile('C:/Users/Herdney/Desktop/knn_libras/libras_knn.py', wdir='C:/Users/
Herdney/Desktop/knn_libras')
              precision    recall  f1-score   support

           0       1.00      1.00      1.00        23
           1       0.95      1.00      0.97        36
           2       1.00      0.96      0.98        24
           3       0.96      0.90      0.93        30
           4       0.95      0.95      0.95        38
           5       1.00      0.95      0.97        38
           6       0.95      0.84      0.89        25
           7       0.81      0.86      0.83        29
           8       1.00      1.00      1.00        40
           9       0.90      1.00      0.95        19
          10       0.92      0.88      0.90        51
          11       0.81      0.96      0.88        23
          12       1.00      1.00      1.00        23
          13       1.00      1.00      1.00        25
          14       1.00      1.00      1.00        35
          15       1.00      1.00      1.00        32
          16       1.00      1.00      1.00        33
          17       0.84      0.84      0.84        25
          18       0.95      0.93      0.94        41
          19       0.98      0.98      0.98        43
          20       0.79      0.79      0.79        47
          21       0.74      0.74      0.74        35
          22       1.00      1.00      1.00        49
          23       0.97      1.00      0.98        28
          24       1.00      0.94      0.97        31
          25       0.88      0.88      0.88        26
          26       0.97      0.97      0.97        30
          27       1.00      0.97      0.98        33
          28       0.97      1.00      0.99        37
          29       1.00      1.00      1.00        27
          30       0.96      0.96      0.96        24
          31       1.00      0.97      0.98        33
          32       0.97      1.00      0.99        34
          33       0.97      1.00      0.99        38
          34       0.97      1.00      0.99        34
          35       1.00      1.00      1.00        26

    accuracy                           0.95      1165
   macro avg       0.95      0.95      0.95      1165
weighted avg       0.95      0.95      0.95      1165
```

Figura 35 – Relatório métrica KNN

Fonte: dos Autores (2019)

4.1.1 Resultados KNN por Classes

Além da acurácia total de 95% conforme apresentado na seção anterior, foi calculado também a acurácia individual de cada classe. A base é composta por 26 letras e 10 números que totalizam 36 classes. Essas classes foram divididas em 36 arquivos separados (um para cada classe). Com isso, foi aplicado o algoritmo para esses 36 arquivos para gerar a matriz de confusão de cada classe. Na figura 36 abaixo mostra um exemplo da matriz de confusão gerada para a classe “A”.

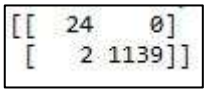


Figura 36 – Matriz de Confusão do KNN

Fonte: dos Autores (2019)

A matriz de confusão refere-se aos resultados de Verdadeiro Positivo, Falso Positivo, Falso Negativo e Verdadeiro Negativo conforme o Quadro 10 abaixo:

	Classe 0	Classe 1
Classe 0	VP	FP
Classe 1	FN	VN

Quadro 10 – Modelo Matriz de Confusão

Fonte: dos Autores (2019)

A acurácia consiste em somar os Verdadeiros Positivos e Verdadeiros Negativos e dividi-los pelo total de amostras conforme demonstrado na fórmula abaixo.

$$Acurácia = (VP + VN) \,/\, (VP + FP + FN + VN) \quad (5)$$

Após aplicar as classes ao algoritmo KNN, obteve-se a matriz de confusão de todas as classes e suas respectivas acurácias conforme apresentado no Quadro 11 abaixo:

Classes	0	1	Erros	Acertos / Acurácia	Total
A	24	0	2	1163	**1165**
	2	1139	0,17%	99,83%	**100,00%**
B	1133	1	1	1164	**1165**

	0	31	0,09%	99,91%	**100,00%**
C	1135	2	4	1161	**1165**
	2	26	0,34%	99,66%	**100,00%**
D	1137	0	1	1164	**1165**
	1	27	0,09%	99,91%	**100,00%**
E	1127	1	4	1161	**1165**
	3	34	0,34%	99,66%	**100,00%**
F	1134	1	1	1164	**1165**
	0	30	0,09%	99,91%	**100,00%**
G	1131	1	6	1159	**1165**
	5	28	0,52%	99,48%	**100,00%**
H	1135	5	7	1158	**1165**
	2	23	0,60%	99,40%	**100,00%**
I	1133	0	0	1165	**1165**
	0	32	0,00%	100,00%	**100,00%**
J	1130	4	8	1157	**1165**
	4	27	0,69%	99,31%	**100,00%**
K	1115	5	6	1159	**1165**
	1	44	0,52%	99,48%	**100,00%**
L	1130	2	6	1159	**1165**
	4	29	0,52%	99,48%	**100,00%**
M	1144	0	1	1164	**1165**
	1	20	0,09%	99,91%	**100,00%**
N	1136	2	3	1162	**1165**
	1	26	0,26%	99,74%	**100,00%**
O	1129	2	2	1163	**1165**
	0	34	0,17%	99,83%	**100,00%**
P	1118	0	1	1164	**1165**
	1	46	0,09%	99,91%	**100,00%**
Q	1138	0	0	1165	**1165**
	0	27	0,00%	100,00%	**100,00%**
R	1136	1	9	1156	**1165**
	8	20	0,77%	99,23%	**100,00%**
S	1131	2	7	1158	**1165**
	5	27	0,60%	99,40%	**100,00%**
T	1122	0	2	1163	**1165**
	2	41	0,17%	99,83%	**100,00%**
U	1115	10	19	1146	**1165**
	9	31	1,63%	98,37%	**100,00%**
V	1127	7	15	1150	**1165**
	8	23	1,29%	98,71%	**100,00%**
W	1131	2	6	1159	**1165**
	4	28	0,52%	99,48%	**100,00%**
X	1131	1	1	1164	**1165**
	0	33	0,09%	99,91%	**100,00%**

Y	1125	1	3	1162	**1165**
	2	37	0,26%	99,74%	**100,00%**
Z	1128	4	5	1160	**1165**
	1	32	0,43%	99,57%	**100,00%**
0	1131	4	5	1160	**1165**
	1	29	0,43%	99,57%	**100,00%**
1	1137	0	1	1164	**1165**
	1	27	0,09%	99,91%	**100,00%**
2	1133	0	0	1165	**1165**
	0	32	0,00%	100,00%	**100,00%**
3	1131	0	0	1165	**1165**
	0	34	0,00%	100,00%	**100,00%**
4	1133	3	3	1162	**1165**
	0	29	0,26%	99,74%	**100,00%**
5	1136	1	1	1164	**1165**
	0	28	0,09%	99,91%	**100,00%**
6	1130	0	0	1165	**1165**
	0	35	0,00%	100,00%	**100,00%**
7	1124	0	0	1165	**1165**
	0	41	0,00%	100,00%	**100,00%**
8	1126	0	2	1163	**1165**
	2	37	0,17%	99,83%	**100,00%**
9	1125	0	0	1165	**1165**
	0	40	0,00%	100,00%	**100,00%**

Quadro 10 – Acurácia por classe

Fonte: dos Autores (2019)

4.2 RESULTADOS COM REDE NEURAL ARTIFICIAL

Para realizar o treinamento da Rede Neural Artificial foi utilizada a mesma base de dados do treinamento para o algoritmo KNN. A Rede Neural é composta por 8 dados de entrada que correspondem as leituras dos sensores dos cinco dedos (Polegar, Indicador, Médio, Anelar e Mínimo) e 3 referentes ao sensor MPU-6050 (ax, ay e az). Além disso, existe uma camada oculta na qual foi atribuído aleatoriamente 22 neurônios e por fim, a camada de saída com 36 neurônios que correspondem as letras do alfabeto (de A a Z) e números (de 0 a 9) conforme Figura 37 abaixo:

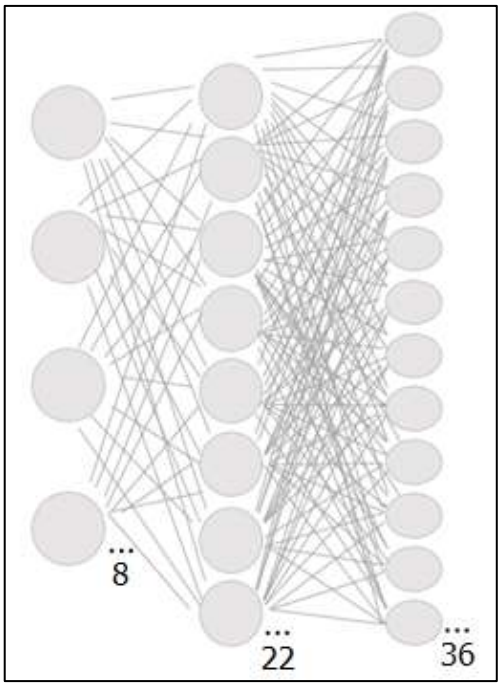


Figura 37 – Rede Neural

Fonte: dos Autores (2019).

Diante disso, os parâmetros utilizados também foram os mesmos, ou seja, 70% (2.716) da base para treinamento e 30% (1.165) para o teste, obtendo os resultados dispostos no Quadro 11 a seguir:

	Amostras	Aplicação (%)
Treinamento	2.716	70%
Teste	1.165	30%
Total	**3.881**	**100%**

Quadro 11 – Coleta de dados para treinamento da RNA

Fonte: dos Autores (2019)

Considerando a porcentagem da amostra que foi utilizada para teste (1.165) obteve-se os seguintes resultados:

	Amostras	%
Acertos	1.091	94%
Erros	74	6%
Total	**1.165**	**100%**

Quadro 12 – Acurácia de predição da RNA

Fonte: dos Autores (2019)

Conforme pode ser observado no Quadro 11, a predição obteve uma acurácia de 94%, ou seja, do total de amostras utilizadas para os testes a predição classificou corretamente 94% dos dados.

A Figura 38 apresenta o detalhamento da predição do algoritmo através da matriz de confusão, onde na linha diagonal representa as amostras classificadas corretamente, enquanto as amostras dispersas referem-se as amostras classificadas incorretamente.

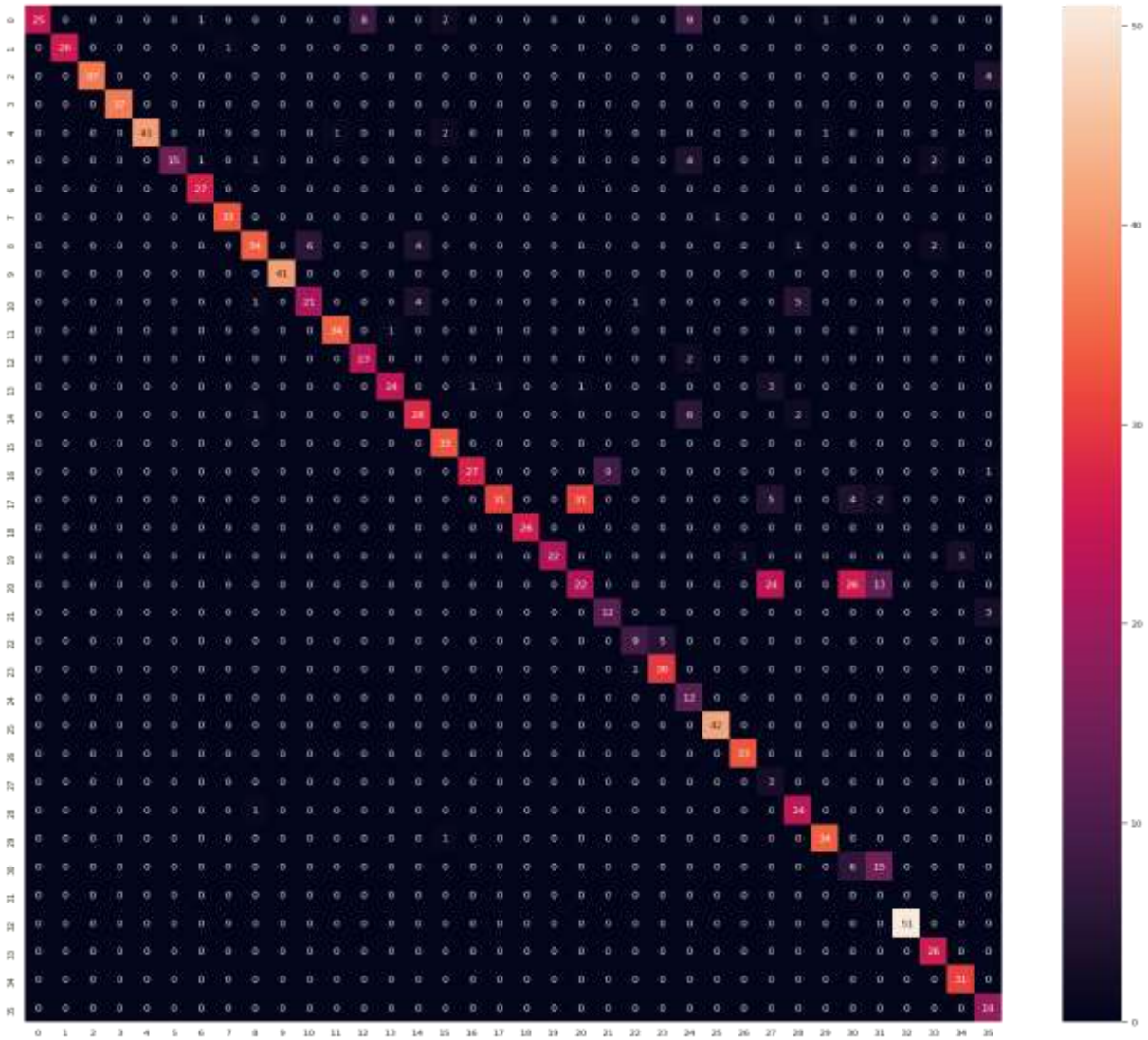


Figura 38 – Matriz de Confusão da RNA

Fonte: dos Autores (2019)

Comparando os resultados entre os algoritmos KNN e Rede Neural Artificial obtivemos os resultados apresentados no quadro abaixo:

	Acertos	Erros	Total Amostra	Acurácia
KNN	1.098	63	1.161	95%
RNA	1.091	74	1.165	94%

Quadro 13 – Comparativo entre KNN e RNA

Fonte: os Autores (2019)

Como pode ser observado, o KNN obteve acurácia 1% melhor que a Rede Neural Artificial, contudo, deve-se levar em conta que houve apenas um treinamento com uma camada oculta com 22 neurônios, ou seja, os resultados podem ser melhores se incluir mais camadas ocultas. Mas como o resultado de ambos os métodos são satisfatórios, outras configurações da Rede Neural Artificial foram desconsideradas e o KNN foi selecionado devido sua facilidade de implementação e bons resultados.

4.3 COMPARATIVO ENTRE TRABALHOS JÁ REALIZADOS

Comparando-se com os trabalhos citados no capítulo Estado da Arte e no Quadro 14, pode-se concluir que a acurácia dos testes com o algoritmo KNN e com a RNA foram positivos, levando-se em consideração que a maior acurácia foi apresentada por Mohandes S.; Aliyu; M. Deriche (2014), os quais obtiveram 98,3% de acurácia nos testes com algoritmo de classificação Naives Bayes e 99,1% como Percéptron Multicamada para tradução da Língua Arábica de Sinais e o menor resultado de acurácia foi de 85% no trabalho dos autores Kim; Jang; Bien (1996) utilizando Rede Neural Fuzzy para a tradução da Língua Coreana de Sinais.

O trabalho mais próximo é o da Malásia, que também utiliza o microcontrolador PIC 18F4550, sensores flexíveis e acelerômetros e obteve acurácia de 91% aplicando modelo oculto de Markov (SWEE et.al., 2007). É possível verificar também que muitos autores utilizaram luvas sensoriais prontas como a *Cyberglove*® por exemplo: nos trabalhos de Wang; Gao; Shan. (2002) e Oz; Leu (2011) que obtiveram resultados próximos ao deste projeto (93% e 90% respectivamente), comprovando ser possível projetar a luva de forma a obter bons resultados.

País de origem da LS	Aquisição de dados	Sensores	Processamento	Método/Classificação	Interface Usuário	AUTOR	Acurácia	Ano
Estados Unidos	VPL *Data-Glove*	Transdutores de fibra óptica e Polhemus 3D	*Silicon Graphics* 4D/240S	*Multilayer Perceptron*	x	S. Sidney Fels and Geoffrey E. Hinton	95%	1992
Coréia	2 VPL *Data-Glove*	Transdutores de fibra óptica e Polhemus 3D	*Sun Sparc-Station* I1	*Fuzzy min-max neural network*	x	Jong-Sung Kim, Won Jang, and Zeungnam Bien	85%	1996
Taiwan	*Data-glove™*	Polhemus 3D	Pentium PC-133	Modelo Oculto de Markov	x	Rung-Huei Liang1, Ming Ouhyoung2	95%	1998
China	2 *Cyberglove™*	Vibro-táteis	Pentium III 700MHz	Modelo Oculto de Markov	x	Chunli Wang, Wen GAO, Shiguang Shan	93%	2002
Malásia	2 *Data-glove*	10 flexíveis + 6 acelerômetros	PIC18F4550	Modelo Oculto de Markov	Interface Gráfica	Tan Tian Swee, A.K. Ariff, Sh-Hussain. Salleh, Siew Kean Seng,and Leong Seng Huat	91%	2007
Estados Unidos	1 *Cyberglove™*	Vibro-táteis	*Flock of Birdss motion tracker*	RNA	x	Cemil Oz a,n, MingC.Leu b	+- 90%	2011
Índia	Captura de Imagem	Câmera MP	Pentium® CPU B950	*Eigen values and Eigen vectors*	MATLAB	Joyeeta Singha e Karen Das	97%	2013
China	*Kinect*	*Hand tracking*	Não especificado	Correspondência de trajetória	Interface Gráfica	Xiujuan Chai, Guang Li, Yushun Lin, Zhihao Xu,Yili Tang, Xilin Chen, Ming Zhou	96%	2013
Arábia	*Leap Motion Controller*	Câmeras Infravermelhas e LEDs	*Leap Motion Controller*	*Naive Bayes Classifier (NBC) e Multilayer Perceptron (MLP)*	MATLAB	M. Mohandes, S. Aliyu and M. Deriche;	NBC-98,3% MLP-99,1%	2014
Brasil	*Data-glove*	Sensores Indutivos	MSP430F5529	*Backpropagation* Levenberg-Marquard	LCD 16x2	Ruani Lazzarotto	97%	2016
Brasil	Luva Sensoriada	Sensores Flexíveis e MPU-6050	Microcontrolador PIC 18F4550	KNN	Aplicativo *Mobile*	Herdney Souza dos Santos, Leila Fabiola Ferreira, Poliana G. L. Alves	95%	2020

Quadro 14 – Quadro Comparativo dos Trabalhos Relacionados

Fonte: dos Autores (2019)

5 CONCLUSÃO

Os resultados iniciais estão satisfatórios, apresentando poucos erros, sendo necessário em uma próxima etapa, analisar um maior número de leituras dos sensores para identificação dos padrões alfanuméricos da língua brasileira de sinais. Ainda assim foi possível construir um banco de dados para mapeamento de letras e treinamento do algoritmo de classificação KNN com bom desempenho e acurácia de 95%.

O projeto como um todo contribuiu muito para o desenvolvimento profissional da equipe, visto que através de conhecimentos teóricos e práticos repassados nas aulas, somando-se aos estudos realizados pela equipe e as dificuldades enfrentadas, foi possível colocar em prática todos esses conhecimentos no projeto.

5.1 PRINCIPAIS DIFICULDADES

Em relação ao *hardware*, as principais dificuldades enfrentadas estão relacionadas com a programação do microcontrolador, principalmente no que tange a aplicação do protocolo I^2C que é bastante complexo. Outra questão é o comprimento da região ativa dos sensores flexíveis que limitou a abrangência de leitura da flexão dos dedos, sendo uma possibilidade de melhoria utilizar sensores com região ativa maior.

Sobre a área de desenvolvimento de aplicativo *mobile* em Java, visto que não havia um conhecimento aprofundado da equipe sobre esse tema, foi necessário dedicar maior tempo em pesquisas e testes, o que tornou este módulo o mais complexo a ser desenvolvido.

5.2 MELHORIAS FUTURAS

Como melhorias futuras, considera-se a construção de uma segunda luva de dados, alteração dos sensores flexíveis por sensores maiores para teste, utilização de microcontrolador mais robusto, aplicação de processamento digital de imagens em

conjunto com as luvas para tradução de sinais em Libras (não somente datilologia) e inclusão de respostas sonoras no aplicativo.

REFERÊNCIAS

Acessos de telefonia móvel no Brasil. Disponível em: <https://www.anatel.gov.br/dados/component/content/article/84-destaque/283-acessos-smp> Acesso em: 25/10/2019.

BARROS, H. (2011) **Linguagem de Programação Java** – Universidade Estadual da Paraíba.

BATES, M. P. (2008). **Programming 8-bit Pic Microcontrollers in C** (1st ed.). Newnes.

BELL, Charles. **Beginning Sensor Networks with Arduino and Raspberry P i**. 1. ed. New York: Apress Media, 2013.

Bluetooth module HC-05 with PIC microcontroller tutorial. Disponível em: <https://deepbluembedded.com/bluetooth-module-hc05-interfacing-pic-microcontroller-tutorial/> Acesso em: 16/10/2019

CARDLE, J P. **Android App Development in Android Studio Java + Android Edition for Beginners**. p. 202, 2016.

CARDLE, J. P., **Android App Development in Android Studio**, Manchester Academic Publishers, 2017.

CASTRO, A. P., CARVALHO, I. S**. Comunicação por Língua Brasileira de Sinais,** 4 ed. Distrito Federal, Senac, 2011.

COPPIN, B. **Inteligência Artificial**. LTC – Livros Técnicos e Científicos Editora Ltda: Rio de Janeiro, 2010.

COSMINA, Iuliana. **Java for Absolute Beginners**. Edinburgh: Apress Media LLC, 2018.

CYBERGLOVE. Disponível em: <http://www.cyberglovesystems.com>. Acesso em: 15/10/2019.

Datasheet módulo HC-05. Disponível em: <https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf> Acesso em: 11/10/2019.

Datasheet MPU-6000 and MPU-6050 Product Specification Revision 3.4 Disponível em: <https://www.olimex.com/Products/Components/RF/BLUETOOTH-SERIAL-HC-06/resources/hc06.pdf> Acesso em: 18/03/2019.

EMPRESA ÁGIL. **COMUNICAÇÃO EMPRESARIAL: Conceito, aplicação e importância**. 1. ed. [S.l: s.n.], 2015.

FADERS. **Mini dicionário**. . POA: Serviço de Ajudas Técnicas. , 2010.

FAHN, C., SUN, H. **Development of a data glove with reducing sensors based on magnetic induction**. Industrial Electronics, IEEE Transactions on, v. 52, n. 2, p. 585–594, 2005.

FELIPE, T. A.. **Pela Regulamentação da Lei 10.436, 24 de Abril de 2002**. Revista da Feneis. Publicação trimestral da Federação Nacional de Educação e Integração dos Surdos, n. 24, p.13-14, jan./mar. 2005. Disponível em <https://issuu.com/feneisbr/docs/revista_feneis_24 > Acesso em 05 nov. 2019.

FREITAS, E. M., Sampei R. **Introdução ao Bluetooth.** 2014.

GÉRON, A. **Hands-On Machine Learning with Scikit-Learn & TensorFlow**. Concepts, tools, and techniques to build inteligente systems. Sebastopol: O'Reilly Media , 2017.

HEDAYATI, R. **A Study of Successive Approximation Registers and Implementation of an Ultra- Low Power 10-bit SAR ADC in 65nm CMOS Technology**. Linköping (MS), n. September, 2011.

IBRAHIM, D. **Advanced Pic Microcontroller Projects in C**. Burlington: [s.n.], 2008.

INSTITUTO BRASILEIRO DE GEOGRAFIA E ESTATISTICA. **Pesquisa Nacional de Saúde 2013**. Rio de Janeiro: [s.n.], 2015.

INSTITUTO NACIONAL DE ESTUDOS E PESQUISAS EDUCACIONAIS ANÍSIO TEIXEIRA , **Sinopse Estatística da Educação Básica,** Brasília**,** 2003.

IOVINE, John. **Pic Microcontroller Project Book**. [S.l.]: McGraw-Hill, 2000.

IRAZABAL, J. M., BLOZIS, S. **DesignCon 2003 TecForum I^2C Bus Overview**, Janeiro, Philips Semiconductors, 2003.

JAVA – Origem e evolução da linguagem Java. Disponível em: <http://www.universidadejava.com.br/materiais/origem-evolucao-java/> Acesso em: 10/10/2019.

LAZZAROTTO, R. **Sistema De Reconhecimento De Padrões Do Alfabeto Da Língua Brasileira De Sinais Utilizando Microcontrolador**. p. 104, 2016.

LECHETA, R. R. **Google Android Aprenda a criar aplicações para dispositivos móveis com Android SDK**. São Paulo: Novatec, 2015

LINS, H. A. M. **Experiências docentes ligadas à educação de surdos: aspectos de formação**. [S.l: s.n.], 2012.

LOPES, M. C. **Cultura surda e Libras**. p. 283, 2012.

MARTINS, S. E. S. O. **Formação de leitores surdos e a educação inclusiva**. [S.l: s.n.], 2011.

MARTINS, V. R. O., NASCIMENTO, L. C. R. **Libras e educação de surdos: experiências docentes na formação de educadores**. p.55-66, Leitura Crítica, 2012.

MULLER, A., GUIDO, S. **Introduction to Machine Learning with Python**. O'Reilly Media, Inc. Sebastopol, CA, 2016.

NAGY, Z. **Artificial Intelligence and Machine Learning Fundamentals**. Birmingham: Packt Publishing, 2018.

NEGNEVITSKY, M. **Artificial Intelligence. A Guide to Intelligent Systems**. England: Pearson, 2005.

OZ, C., LEU, M. C. **American Sign Language word recognition with a sensory glove using artificial neural networks**. Engineering Applications of Artificial Intelligence, v. 24, n. 7, p. 1204–1213, 2011.

PEREIRA, F**. PIC Programação em C.** 7 ed. São Paulo, Érica, 2007.

Projeção da população do Brasil e das Unidades da Federação. Disponível em: <https://www.ibge.gov.br/apps/populacao/projecao/> Acesso em: 25/10/2019.

Programação orientada a objetos: Uma introdução. Disponível em: <https://www.hardware.com.br/artigos/programacao-orientada-objetos/> Acesso em: 23/10/2019.

QUADROS, R. M., STUMPF, M. R., LEITE, T. A.. **Estudos da Língua Brasileira de Sinais I**. 1. ed. Florianópolis: Insular, 2013.

ROTHMAN, D. **Artificial Intelligence by Example**. Packt Publishing: Reino Unido, 2018.

RUSSELL, S. J., NORVIG, P. **Artificial Intelligence A Modern Approach**. 3 ed. New Jersey: Pearson, 2010.

RUSSELL, R. **Machine Learning Step-by-Step Guide To Implement Machine Learning Algorithms with Python**. 2018.

SOUSA, D. P., SOUZA, D. J**. Desbravando o Microcontrolador PIC18.** 1 ed. São Paulo, Érica, 2012.

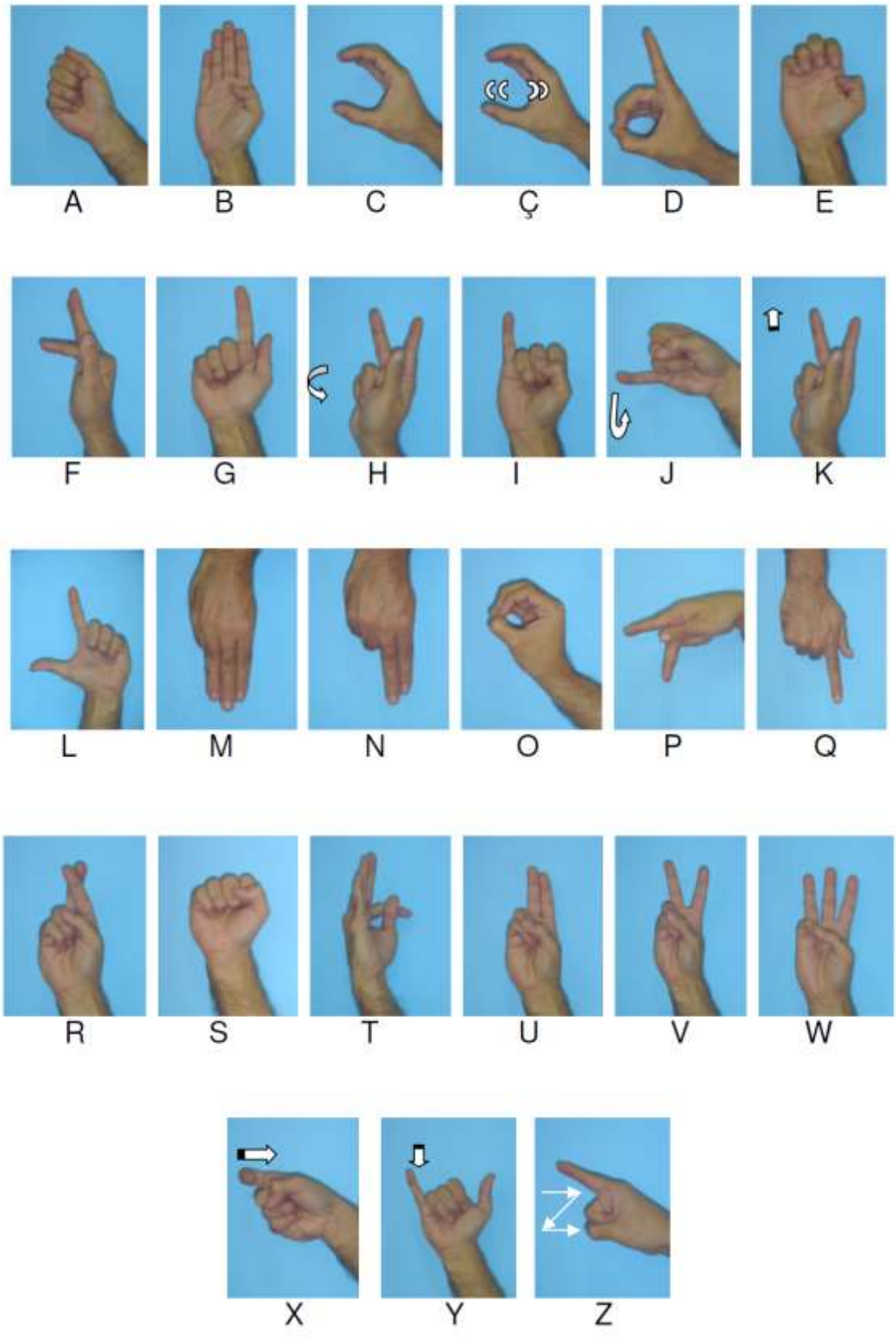
SUTTON, R. S., BARTO, A. G. **Reinforcement Learning: An Introduction**. 2 ed. MIT – Massachusetts: Inglaterra, 2018.

TAN, T. S. e colab. **Wireless data gloves Malay sign language recognition system**. 2007 6th International Conference on Information, Communications and Signal Processing, ICICS, p. 3–6, 2007.

WANG, C., GAO, W., SHAN, S. **An approach based on phonemes to large**

vocabulary Chinese sign language recognition. Proceedings - 5th IEEE International Conference on Automatic Face Gesture Recognition, FGR, p. 411–416, 2002.

ANEXO A - Alfabeto manual



Fonte: Mini dicionário (2010)