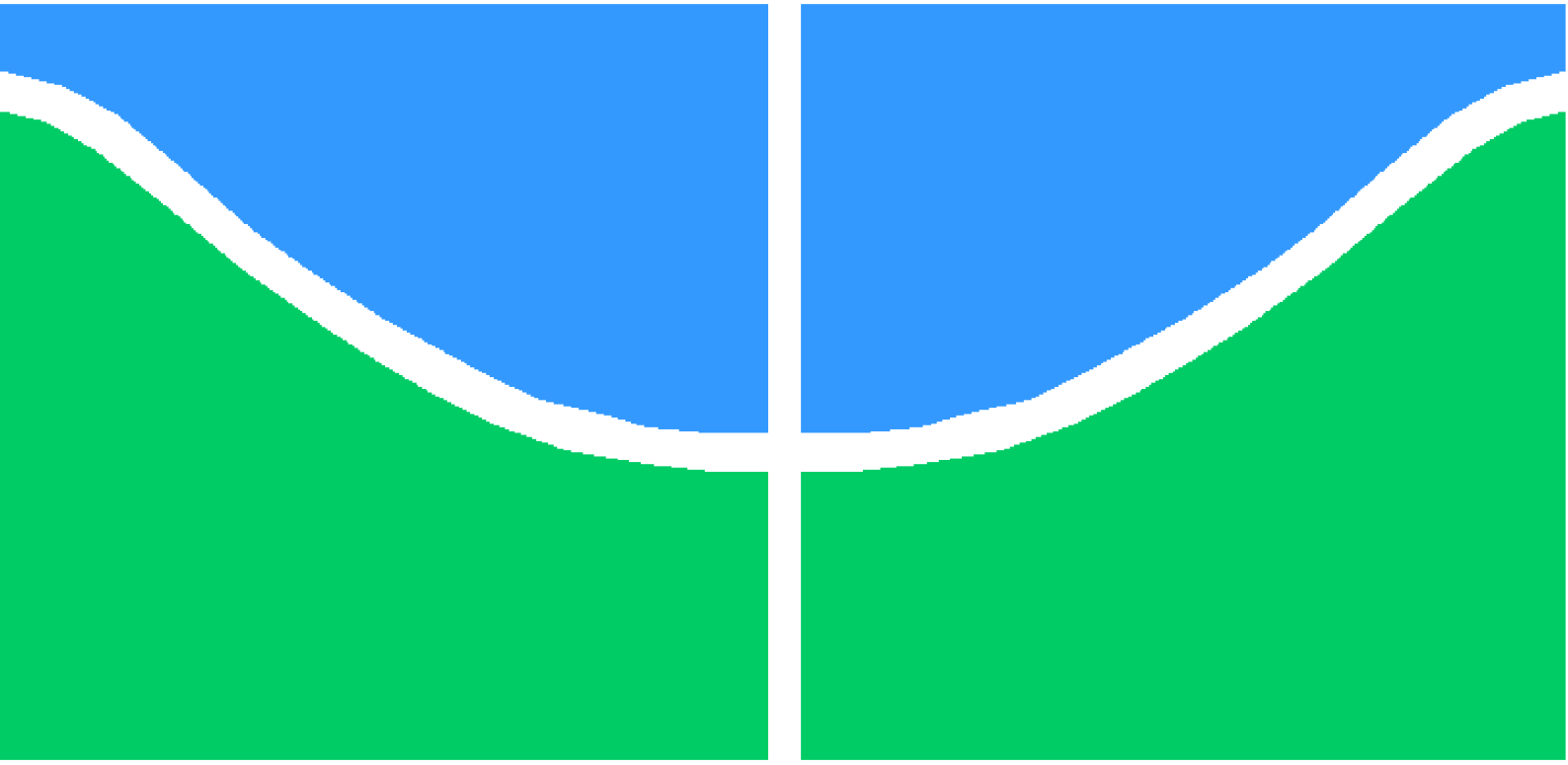


Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Engenharia Eletrônica

Estudo e desenvolvimento de um sistema de reconhecimento de expressões faciais para controle de cadeira de rodas

Autor: Ingrid Miranda de Sousa

Orientador: Dr. Gerardo Antonio Idrobo Pizo

Brasília, DF

2019

Ingrid Miranda de Sousa

Estudo e desenvolvimento de um sistema de reconhecimento de expressões faciais para controle de cadeira de rodas

Monografia submetida ao curso de graduação em Engenharia Eletrônica da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Eletrônica.

Universidade de Brasília - UnB

Faculdade UnB Gama - FGA

Orientador: Dr. Gerardo Antonio Idrobo Pizo

Coorientador: Dr. Daniel Mauricio Muñoz Arboleda

Brasília, DF

2019

Ingrid Miranda de Sousa

Estudo e desenvolvimento de um sistema de reconhecimento de expressões faciais para controle de cadeira de rodas/ Ingrid Miranda de Sousa. – Brasília, DF, 2019-

65 p. : il. (algumas color.) ; 30 cm.

Orientador: Dr. Gerardo Antonio Idrobo Pizo

Trabalho de Conclusão de Curso – Universidade de Brasília - UnB
Faculdade UnB Gama - FGA , 2019.

1. Reconhecimento de expressões faciais. 2. Visão computacional. I. Dr. Gerardo Antonio Idrobo Pizo. II. Universidade de Brasília. III. Faculdade UnB Gama. IV. Estudo e desenvolvimento de um sistema de reconhecimento de expressões faciais para controle de cadeira de rodas

CDU 02:141:005.6

Ingrid Miranda de Sousa

Estudo e desenvolvimento de um sistema de reconhecimento de expressões faciais para controle de cadeira de rodas

Monografia submetida ao curso de graduação em Engenharia Eletrônica da Universidade de Brasília, como requisito parcial para obtenção do Título de Bacharel em Engenharia Eletrônica.

Trabalho aprovado. Brasília, DF, 11 de dezembro de 2019 - Data da aprovação do trabalho:

Dr. Gerardo Antonio Idrobo Pizo
Orientador

Dr. Daniel Mauricio Muñoz Arboleda
Coorientador

Dr. Jones Yudi Mori Alves da Silva
Convidado 1

Dr. Renato Coral Sampaio
Convidado 2

Brasília, DF
2019

.

Dedico aos meus pais e ao meu irmão, que sempre me deram força, acreditaram no meu potencial e incentivaram meus estudos. Obrigada por não medirem esforços para que eu alcançasse essa etapa. Amo vocês.

Agradecimentos

Agradeço primeiramente à Deus, que me deu saúde e força para enfrentar todas as dificuldades.

À Universidade de Brasília pela oportunidade de estudo.

Aos professores pelos ensinamentos no decorrer do curso. Principalmente, ao meu orientador, Gerardo, por todo o conhecimento passado, incentivo, disponibilidade em ajudar, além do tempo e paciência investidos nesse trabalho.

Aos meus pais, Emerson e Soraia, por todo o cuidado, encorajamento, dedicação e carinho. Além de serem uma grande referência tanto como pessoas quanto como profissionais. Obrigada também pelo apoio e sugestões de aperfeiçoamento, vocês foram essenciais para a execução desse trabalho.

Ao meu irmão, João Lucas, por ser um excelente amigo e sempre estar ao meu lado. Bem como por aceitar participar da fase de testes experimentais do programa.

Aos meus avós, em especial à minha avó, Antônia (in memorian), por ter sido um exemplo de pessoa e que sempre torceu para que eu alcançasse essa etapa.

Ao meu amigo, Mateus de Oliveira, pelo companheirismo em todos os momentos e inclusive por se dispor a ler e propor melhorias à esse trabalho.

E por fim, à toda minha família, amigos, colegas de estudo, companheiros de trabalho e todos que contribuíram direta ou indiretamente na minha formação.

"You cannot hope to build a better world without improving the individuals. To that end each of us must work for his own improvement, and at the same time share a general responsibility for all humanity, our particular duty being to aid those to whom we think we can be most useful."

(Marie Curie)

Resumo

A tecnologia tem sido cada vez mais utilizada em diversos âmbitos. E uma das áreas que pode agregar bastante para a sociedade atual consiste na tecnologia assistiva, a qual proporciona recursos que auxiliam pessoas com deficiência física. Considerando isso, esse trabalho tem o intuito de promover uma solução que melhore a locomoção de tetraplégicos, por meio da criação de uma interface de controle, para cadeira de rodas elétrica, utilizando visão computacional. À vista disso, foi elaborado um sistema apto para capturar imagens; realizar a detecção de face do usuário; efetuar a extração de características; captar expressões faciais dos olhos e boca, além de movimentos de inclinação da cabeça; e por fim, associá-los com comandos capazes de controlar a cadeira. Além disso, paralelamente, enviar os dados via serial para um microcontrolador e apresentar tais resultados em uma interface gráfica. Para isso, foram utilizadas as bibliotecas de visão computacional OpenCV e Dlib, a linguagem de programação Python e o framework para interface gráfica Tkinter. De modo que, obteve-se como resultado um sistema de reconhecimento em tempo real, para ambientes adequadamente luminosos, de quatro comandos: frente, trás, esquerda e direita. Diante disso, conclui-se que, apesar de melhorias serem necessárias, tal interface cumpre seu objetivo proposto.

Palavras-chaves: Interface de controle. Cadeira de rodas. Visão computacional. Reconhecimento de expressões faciais. OpenCV. Dlib. Python.

Abstract

The technology has been more used in different scopes. One of the areas that can aggregate a lot for the current society consists in the assistive technology, which provides resources that help people with physical disability. Considering that, this work has the goal to promote a solution capable of improve tetraplegic's locomotion, by creating an eletric wheelchair control interface based on computer vision. Then, it was elaborated a system able to capture images, to make the user's face detection, to efetuate the feature extraction and to capture facial expressions from the eyes and mouth, also movements of head inclination. Finally, associating them with the comands that control the wheelchair. Futhermore, at the same time, sending this data via serial to a microcontroller and presenting this movements in a graphic interface. For this, it was used the computer vision libraries OpenCV and Dlib, with the programming language Python and the GUI framework Tkinter. So that in the end, it was obtained as result a recognize system in real time, that works on properly illuminated environments, with four commands: ahead, backwards, left and right. Consequently, it was concluded that, despite improvements are needed, this interface accomplishs its goal.

Key-words: Control interfaces. Wheelchair. Computer vision. Facial expressions recognition. OpenCV. Dlib. Python.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

API	Application Programming Interface
AU	Action Units
CFS	Correlation Features Selection
CPU	Central Processing Unit
EEG	Eletroencefalografia
EOG	Eletrooculografia
FACS	Facial Action Coding System
FPS	Frames por segundo
GPL	GNU General Public License
GUI	Graphical User Interface
HOG	Histogram of Oriented Gradients
IBGE	Instituto Brasileiro de Geografia e Estatística
ICC	Interface cérebro-computador
LFW	Labeled Faces in the Wild
MP	Megapixel
OpenCV	Open Source Computer Vision Library
PSFL	Python Software Foundation License
RGB	Red, Greend and Blue
SDK	Software Development Kit
SVM	Support Vector Machine
TCC	Trabalho de Conclusão de Curso
USB	Universal Serial Bus

Lista de símbolos

d_{A_B}	Distância euclidiana de abertura da boca
$d_{A_{IC}}$	Distância euclidiana de altura da inclinação da cabeça
$d_{A_{OD}}$	Distância euclidiana de abertura do olho direito
$d_{A_{OE}}$	Distância euclidiana de abertura do olho esquerdo
d_{C_B}	Distância euclidiana de comprimento da boca
$d_{C_{OD}}$	Distância euclidiana de comprimento do olho direito
$d_{C_{OE}}$	Distância euclidiana de comprimento do olho esquerdo
d_{C_R}	Distância euclidiana de comprimento do rosto
p_n	Ponto facial de número n
x_B	Coeficiente de abertura da boca
x_{IC}	Coeficiente de inclinação da cabeça
x_{OD}	Coeficiente de abertura do olho direito
x_{OE}	Coeficiente de abertura do olho esquerdo
R_{LED}	Resistência associada ao LED
$V_{PD_{Arduino}}$	Tensão da porta digital do Arduino
V_{LED}	Tensão de operação do LED
I_{LED}	Corrente de operação do LED
$\sum$	Somatório

Sumário

1 Introdução

No último Censo Demográfico, realizado a cada dez anos, 45,6 milhões de pessoas declararam ter ao menos um tipo de deficiência, incluindo visual, auditiva, motora ou mental/intelectual (IBGE, 2010). Além disso, cerca de 7% da população total brasileira afirmou possuir deficiência motora, ou seja, exibir certo grau de dificuldade no quesito mobilidade. Uma vez que a mobilidade está associada à capacidade de um indivíduo se locomover em um ambiente ou manipular objetos.

À vista disso, pessoas que possuem algum fator correlacionado à redução da eficácia de movimentação requerem maior esforço na realização de certas atividades do cotidiano. Contudo, a tecnologia assistiva pode auxiliar nesse aspecto, seja qual for a parte do corpo afetada. Segundo Bersch e Tonolli (2006), o conceito de tecnologia assistiva refere-se a um termo utilizado para identificar toda a gama de serviços e recursos que colaboram com a ampliação de habilidades funcionais de pessoas que possuem deficiência, e por consequência, promovem mais inclusão e independência. Alguns exemplos de recursos auxiliares a esse intuito são andadores, cadeiras de rodas e próteses.

Ademais, esses dados do IBGE (2010) também mostram que dentre os indivíduos que declararam ter alguma deficiência motora, 33,42% apresenta a classificação mais grave, a qual indica incapacidade de andar sem auxílio de outra pessoa, isto é, os cadeirantes.

Atualmente, existem diversos modelos de cadeira de rodas disponíveis no mercado, devido às particularidades nas necessidades de cada usuário. Os quais, dependendo do nível de lesão medular que sofreram, podem ser agrupados em duas categorias principais: paraplégicos ou tetraplégicos. De acordo com Bromley (2006), a paraplegia relaciona-se à perda ou diminuição da função motora dos membros inferiores e órgãos pélvicos. Enquanto, a tetraplegia está ligada à redução dos movimentos tanto do membros inferiores e órgãos pélvicos, quanto do tronco e dos membros superiores.

Para Bertoncello e Gomes (2002), pode-se classificar as cadeiras de rodas existentes com base no grau de tecnologia presente em cada uma delas. Onde, as de baixa complexidade tecnológica denominam-se Mecanomanuais, as de média são as Eletromecânicas e as de alto grau de complexidade Eletroeletrônicas.

As cadeiras de rodas mecanomanuais não possuem mecanismos complexos. De modo que são conduzidas pelo trabalho muscular do próprio usuário ou por um terceiro. Dentro dessa categoria, pode-se subdividir ainda em relação ao nível tecnológico envolvido, principalmente em relação aos materiais utilizados. A Figura 1 mostra um exemplo desse tipo de cadeira.

Figura 1 – Cadeira mecanomanual (HOSPINET, 2016)

A cadeira de rodas eletromecânica, vista na Figura 2, também é conhecida como motorizada. Ela proporciona mais autonomia, pois não exige tanto esforço muscular do usuário, que pode controlá-la apenas com um joystick usando a força dos dedos, dos pés ou do queixo. São indicadas para pessoas que possuem mobilidade reduzida, mas que ainda tenham certa habilidade motora nos membros superiores.

Figura 2 – Cadeira eletromecânica (ORTOPONTO, 2017)

Por fim, as eletroeletrônicas consistem nas cadeiras que contam com dispositivos elétricos ou eletrônicos capazes de usar princípios computacionais para controlá-las. Dentre elas, pode-se citar, por exemplo, a que possui câmera para captar o movimento dos olhos. Tal como mostrado na Figura 3.

1.1 Justificativa

Indivíduos que sofreram maiores danos na medula não possuem total autonomia para movimentar-se na cadeira de rodas. Então, para eles pode ser ideal utilizar uma ver-

Figura 3 – Cadeira eletroeletrônica (HYPENESS, 2018)

são motorizada. Todavia, existem alguns usuários, com lesões ainda mais graves, que não detêm da capacidade cognitiva e neuromuscular necessária para navegar em um ambiente dinâmico com o joystick (COWAN et al., 2012).

Segundo Fehr, Langbein e Skaar (2000), clínicos indicam que cerca de metade desses pacientes que não são capazes de operar uma cadeira de rodas motorizada, utilizando métodos convencionais, seriam beneficiados por um sistema de navegação controlado por computador. Nesse caso, a solução mais apropriada seria utilizar uma cadeira eletroeletrônica.

Entretanto, ao analisar o mercado brasileiro observa-se a dificuldade de adquirir uma dessas cadeiras, pois é de alto custo financeiro e ainda não é comercializada no país. Isto posto, é notável a necessidade de se desenvolver uma solução de baixo custo que possa agregar mais mobilidade à esse grupo de pessoas, em específico. E assim, buscar trazer mais independência, conforto, segurança e qualidade de vida.

Devido à complexidade de desenvolver uma cadeira parcialmente ou totalmente autônoma, é fundamental criar divisões de trabalho. Nesse cenário, uma equipe de alunos e professores da Universidade de Brasília se dispôs a desenvolver uma cadeira de rodas autônoma, a partir de uma cadeira de rodas comum. Sendo que, já foi desenvolvido e patenteado um kit para motorização de cadeira de rodas manuais (FILHO, 2010) e (QUINTERO R. H. VAN ELS, 2017). Ademais, foi implementado em software um sistema de controle (IVO, 2017). E após isso, foi realizado a instrumentação da cadeira (BRANCO, 2018).

Esse trabalho visa contribuir com os outros por meio da criação de um sistema de reconhecimento de comandos para controle da cadeira, voltado ao público tetraplégico. Permitindo que o próprio usuário determine o caminho que pretende seguir, utilizando suas expressões faciais.

1.2 Objetivos

1.2.1 Objetivo geral

Desenvolver um sistema capaz de reconhecer expressões faciais do usuário e convertê-las em comandos capazes de controlar o movimento de uma cadeira de rodas elétrica.

1.2.2 Objetivos específicos

- Especificar um sistema de aquisição de imagens;
- Definir o conjunto de expressões faciais que originarão os comandos;
- Desenvolver um algoritmo de processamento das imagens obtidas e de reconhecimento das expressões delimitadas;
- Realizar o envio de dados de controle para um microcontrolador;
- Criar uma interface gráfica que represente os movimentos reconhecidos;

1.3 Metodologia

Para o desenvolvimento deste trabalho foi fundamental realizar uma revisão bibliográfica, a fim de compreender as tecnologias existentes e apresentar melhorias. Sendo que o método proposto foi implementado na linguagem Python e usou as bibliotecas OpenCV, Dlib e Tkinter. Logo, primeiramente foi essencial instalar tais ferramentas computacionais.

Na etapa de criação do algoritmo foram implementadas funções de reconhecimento facial, aplicação de landmarks do rosto e extração de características pelo método de Distâncias Euclidianas. Com base em tais características, foram reconhecidas as expressões faciais, as quais associaram-se com os comandos de controle.

Posteriormente, os dados de comandos foram enviados para um microcontrolador e criou-se uma interface gráfica capaz de apresentar os comandos reconhecidos. Então, por fim, efetuou-se testes de desempenho do sistema e aplicação de possíveis aperfeiçoamentos.

1.4 Organização do trabalho

Este trabalho está estruturado em quatro capítulos. O capítulo 1 tem caráter introdutório, logo, busca contextualizar a natureza do problema, defini-lo e apontar todos os objetivos pretendidos. O capítulo 2 apresenta toda a revisão bibliográfica sobre o tema, então, contém a fundamentação teórica necessária para justificar as escolhas de projeto. O capítulo 3 apresenta todas as ferramentas computacionais utilizadas. O capítulo 4

retrata a solução proposta, descrevendo toda a metodologia adotada durante a execução do sistema. O capítulo 5 mostra os resultados alcançados com a realização do projeto, por meio de testes. Por fim, o capítulo 6 contém as considerações finais do trabalho, sugerindo futuras implementações para a continuação do mesmo.

2 Fundamentação teórica

Esse capítulo apresenta o estudo realizado a respeito do estado da arte. Então, realizou-se uma pesquisa em relação às soluções tecnológicas existentes que auxiliem portadores de tetraplegia. E com isso, observou-se diversas interfaces de controle de cadeira de rodas, mostradas na seção 2.1, que podem ser comandadas com o mínimo de esforço físico e apenas pelas regiões acima do pescoço.

Além disso, considerando que a solução pretendida envolve o reconhecimento de expressões faciais, averiguou-se também o nível de desenvolvimento dessa área. Como pode ser analisado na seção 2.2.

Posteriormente, apresentou-se conceitos importantes para o desenvolvimento do trabalho tais como: expressões faciais, sistema de codificação facial, imagem digital, processamento de imagens e visão computacional.

2.1 Interfaces de controle de cadeira de rodas

A evolução de áreas tais como a robótica e a visão computacional tem acarretado no desenvolvimento de equipamentos e outros aparatos habilitados no auxílio da locomoção de cadeirantes. Em especial, de tetraplégicos que necessitam de alguém que os conduza. Desse modo, diversas pesquisas estão sendo feitas no intuito de construir formas de permitir que uma cadeira de rodas motorizada seja controlada por uma interface de comunicação homem-máquina. Sendo que, algumas dessas interfaces podem ser vistas a seguir.

2.1.1 Movimento dos olhos

Um dos modos possíveis de realizar o controle de uma cadeira de rodas é feito por meio do mapeamento do movimento dos olhos. Araújo et al. (2010) afirma que é viável utilizar interfaces oculares invasivas ou não invasivas, tanto para captar os sinais elétricos provenientes dessa movimentação quanto para identificar a posição onde encontra-se os olhos. Dentre as técnicas existentes para capturar e diagnosticar tais sinais oculares, pode-se citar:

- Eletrorretinografia
- Potencial Evocado Visual
- Videoculografia

- Oculografia Infravermelho
- Eletrooculografia

Um dos protótipos existentes que utiliza essa metodologia consiste no Projeto SIAMO, o qual, usa a eletro-oculografia (EOG) como interface de controle. E para isso, posicionam-se cinco eletrodos ao redor dos olhos, tal como mostrado na Figura 4. Onde, dois deles são encarregados de detectar o movimento horizontal e ficam dispostos ao lado dos olhos; o outro par identifica o movimento vertical, posiciona-se um acima e um abaixo do olho direito; e o quinto eletrodo, utilizado como referência, está situado na testa. Com isso, afere-se a tensão induzida nesse sistema de eletrodos, possibilitando a estimação do potencial elétrico existente entre a retina e a córnea (BAREA et al., 2002).

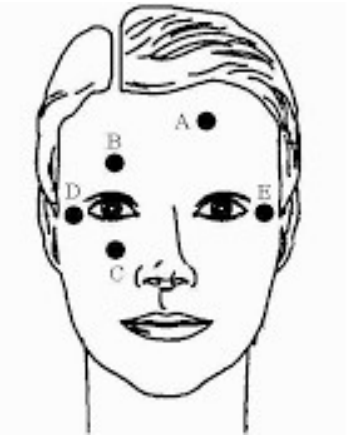


Figura 4 – Posicionamento dos eletrodos para interface baseada em EOG - Projeto SIAMO (BAREA et al., 2002)

No geral, esse tipo de controle requer atenção, pois existe o risco de acontecerem erros causados pela mudança da posição dos olhos antes que um comando seja finalizado. E ainda, no caso do uso de eletrodos, é possível que o usuário se sinta incomodado.

2.1.2 Comando de voz

A técnica de controle de voz pode ser utilizada por qualquer indivíduo capaz de falar de maneira consistente, ou seja, requer menos habilidades físicas (SIMPSON; LEVINE, 2002).

Seu desenvolvimento é dado por meio de um microfone que captura a voz do usuário. Em seguida, é feito o processamento dos dados recebidos, a identificação de palavras e a validação. Por fim, se é comprovado que a palavra dita corresponde a um dos comandos predeterminados, o mesmo é acionado.

Entretanto, um empecilho encontrado na implementação desse modelo consiste na possibilidade de haver falhas na identificação do comando, principalmente em ambientes ruidosos.

2.1.3 Cérebro-computador

A interface cérebro-computador (ICC), do inglês *brain-computer interface* é um sistema que usa sinais elétricos advindos de certas áreas do cérebro para ativar dispositivos externos, tais como computadores ou próteses (MACHADO et al., 2009).

Pode-se citar como um exemplo de método não invasivo, referente a essa interface, a eletroencefalografia (EEG). Tanaka, Matsunaga e Wang (2005) desenvolveram um algoritmo de treino, que funciona de maneira recursiva, a fim de reconhecer padrões provenientes desses sinais cerebrais, e posteriormente, controlar as direções seguidas pela cadeira de rodas.

2.1.4 Sopro e sucção

Essa interface é constituída por um tubo de ar e um dispositivo de controle que converte o fluxo de ar em um sinal capaz de controlar a cadeira, através de circuitos eletrônicos micro-controlados, tal como no trabalho de Ferreira (2008). Portanto, o sistema consiste de um duto rígido, onde uma das extremidades localiza-se em frente à boca do condutor da cadeira, enquanto a outra está voltada ao transdutor de fluxo de ar, conforme mostrado na Figura 5.

Figura 5 – Interface de controle por sopro e sucção (FERREIRA, 2008)

Nesse tipo de interface é fundamental fornecer ferramentas que mostrem ao usuário o comando que foi selecionado. Indicando, por exemplo, o sentido e a velocidade de movimentação.

2.2 Sistemas de reconhecimento de expressões faciais

Atualmente, informações provindas de imagens da face possuem uma vasta gama de aplicações, por exemplo: em sistemas de vigilância, monitorando indivíduos com expressões controversas; no entretenimento computacional; no aperfeiçoamento de ferramentas de comunicação digital; na educação, para análise de interesse do aluno durante a aula; e na saúde, acompanhando os pacientes em hospitais.

Desse modo, é essencial realizar a localização da face e extração das suas características. Posto isso, diversos trabalhos de visão computacional estão sendo desenvolvidos com intuito também de reconhecer expressões faciais. O funcionamento de alguns deles pode ser visto a seguir.

O trabalho de Miranda et al. (2009) apresenta procedimentos para a realização do reconhecimento facial mediante aplicação dos conceitos de: máscara de utilizador, que tem função de focar o esforço computacional para áreas específicas da imagem captada; filtro de macro-visão, o qual diminui a sensibilidade do sistema ao ruído excessivo; estimativa de posicionamento tridimensional por técnicas de regressão; e de *reality checks*, por estimativa da continuidade do movimento.

Leão et al. (2012) desenvolveu um sistema em tempo real, usando *Kinect*, para identificar a emoção do usuário dentre uma das seis opções: raiva, felicidade, medo, tristeza, surpresa e aversão. Para tanto, utilizou como base o algoritmo *Haartraining*. Além disso, fez a colocação automática dos pontos encarregados de realizar a leitura dos movimentos da face, usou a técnica de análise do fluxo óptico nesses pontos e interpretou tais movimentos como *action units* (AU).

Outro trabalho nessa área, foi feito por Sousa et al. (2016), o qual realizou uma avaliação dos aspectos afetivos em aplicações computacionais com base no reconhecimento de expressões faciais e emoções. Para tal fim, desenvolveu uma ferramenta computacional que captura as imagens por meio do *Kinect*, posteriormente realiza o rastreamento de características faciais com o algoritmo de Viola-Jones e a técnica CANDIDE-3, ambos recursos provenientes da biblioteca do *Kinect* SDK. Após isso, é feito a classificação de acordo com a unidade de animação (UA) correspondente, que pode ser comparada com a AU do FACS. Por fim, é feito o registro de emoções na base de dados e avalia-se a usabilidade por computação afetiva.

Junior et al. (2016) desenvolveu uma metodologia para reconhecimento de expressões faciais utilizando técnicas baseadas na geometria facial humana, propondo dois métodos: o das Distâncias Empíricas, que obteve 77,66% de acurácia, e o de Distâncias CFS, que alcançou 91,33%. Sendo que após as fases de extração e seleção de características, utilizou como classificador de padrões a máquina de vetores de suporte, por adequar-se a menos amostras de treinamento.

SANTIAGO (2017) elaborou um extrator de características que aplica estimação de movimento para o reconhecimento de expressões faciais. Essa estimação é feita com base nos deslocamentos de regiões entre duas imagens. Onde, a partir dos vetores de movimento, são obtidos os vetores de características. Os quais, tornam-se os dados de entrada para uma SVM (*Support Vector Machine*), que consequentemente irá classificar a expressão facial.

2.3 Expressões faciais

As expressões faciais consistem em mudanças proporcionadas pelos músculos da face, vistos na Figura 6, os quais movimentam-se voluntaria ou involuntariamente. Elas são decorrentes do estado emocional ou das intenções do indivíduo. Assim, tais expressões além de poder ser naturais e espontâneas, podem também ser controladas. Desse modo, é um dos elementos mais importantes para a comunicação humana.

À vista disso, é possível predeterminar algumas expressões de fácil execução, caracterizá-las e construir uma tabela que as relacione com comandos. E posteriormente, utilizar esses comandos para o controle da cadeira de rodas.

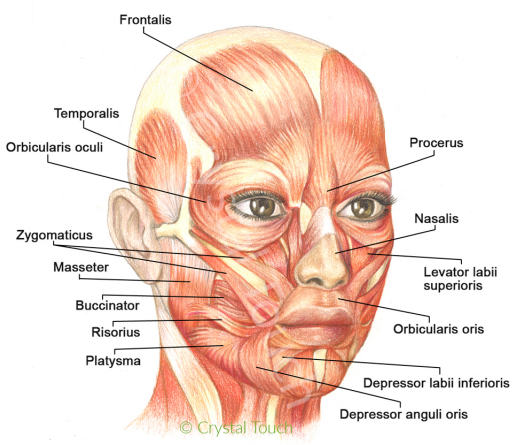


Figura 6 – Musculatura facial (CLINIC, 2019)

2.4 Sistema de codificação facial

O *Facial Action Coding System* (FACS) é um sistema criado por Ekman (1997) capaz de descrever todos os movimentos faciais visíveis. É utilizado de diferentes formas, com variados intuitos e por diversos públicos, que vão desde animadores até cientistas da computação com interesse em reconhecimento facial. É composto por unidades de ação, do inglês *Action Units* (AUs), as quais consistem em componentes de representação do movimento muscular individual, além de *Action Descriptors (ADs)*, que são movimentos

unitários que envolvem a ação de um grupo de músculos e *Movements (Ms)*. No entanto, costuma-se denominar todas essas classes como AUs.

Então, cada AU possui um código numérico e tem uma função associada, como é possível observar nos exemplos contidos na Tabela 1. Essas unidades de ação podem ser utilizadas de maneira singular ou em conjuntos com outras AUs, assim como nota-se na Figura 7. Dessa forma torna-se viável codificar as expressões faciais com maior precisão.

Tabela 1 – Número e características de algumas AUs existentes

Número da AU	Ação principal
1	Levantar parte externa da sobrancelha
2	Levantar parte interna da sobrancelha
4	Abaixar a sobrancelha
5	Levantar a pálpebra superior
6	Levantar a bochecha
7	Apertar a pálpebra
9	Enrugar o nariz
10	Levantar o lábio
12	Levantar canto da boca
15	Retrair canto da boca para baixo
16	Direcionar lábio inferior para baixo
17	Elevar o queixo
19	Colocar a língua para fora
26	Descer o queixo
27	Abrir amplamente a boca
28	Sugar os lábios
33	Encher as bochechas
43	Fechar os olhos
45	Piscar
55	Cabeça inclinada para esquerda
56	Cabeça inclinada para direita

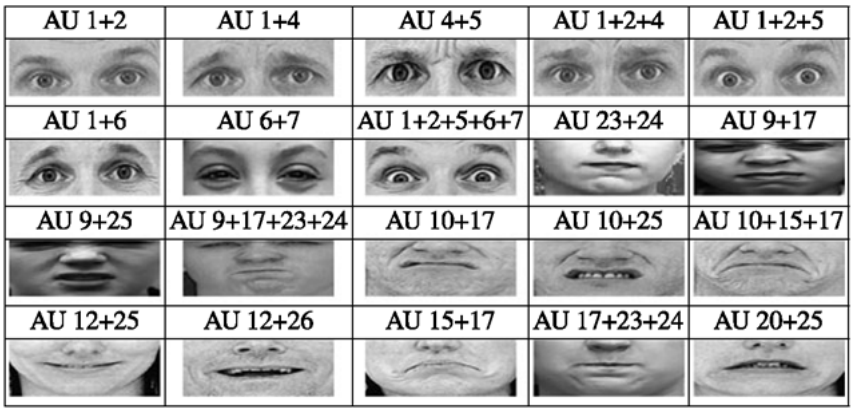


Figura 7 – Exemplos de combinações de AUs (WHAT-WHEN-HOW, 2012)

2.5 Imagem digital

Uma imagem pode ser definida como uma função de duas dimensões, f(x,y), tal que x e y são coordenadas espaciais do plano e f é a intensidade ou nível de cinza da imagem naquela localização.

O conceito de imagem digital é estabelecido como sendo a descrição da imagem, mas como uma sequência finita de valores discretos. Posto que, cada um desses valores corresponde a um pixel. E um pixel consiste na menor unidade de uma imagem digital.

Dentre os formatos de imagem existentes, pode-se citar:

- Imagens binárias: também conhecidas como imagens lógicas, são matrizes de duas dimensões, onde pode ser atribuído a cada pixel os valores 0 ou 1. Sendo que, o bit 0 indica a cor preta e o bit 1 a cor branca.

- Imagens de intensidade ou escala de cinza: são matrizes bidimensionais que designam a intensidade em um ponto, por meio da atribuição de um valor a cada pixel. O intervalo possível é dependente da resolução de bits da imagem. Por exemplo, em uma imagem de 8 bits, a variação provável será entre 0 e 255.

- Imagens RGB ou coloridas: são matrizes tridimensionais que alocam em cada pixel três valores numéricos. Um para cada uma das componentes: vermelho, verde e azul, do inglês *red, green and blue* (RGB). É plausível considerar que esse tipo de imagem possui três planos 2D distintos, então sua dimensão é dada por C x R x 3. No qual, C representa o número de colunas e R o número de linhas totais.

- Imagens em ponto flutuante: divergem dos outros tipos, pois ao invés de armazenar valores do tipo inteiro, utilizam ponto flutuante para representar a intensidade. Assim como no caso anterior, seu intervalo relaciona-se com a resolução de bit (SOLOMON; BRECKON, 2013).

2.6 Processamento de imagens

Há um interesse significativo da comunidade científica pela área de processamento de imagens, pois ela é responsável por viabilizar variadas aplicações em duas categorias distintas: o aperfeiçoamento de dados pictóricos para interpretação humana e a análise automática do computador de informações retiradas de uma cena (FILHO; NETO, 1999).

O processamento de imagens é composto de alguns processos fundamentais, os quais são descritos no diagrama, que pode ser observado na Figura 8. Sendo que, cada um deles possui uma função associada, e essa relação é mostrada na Tabela 2.

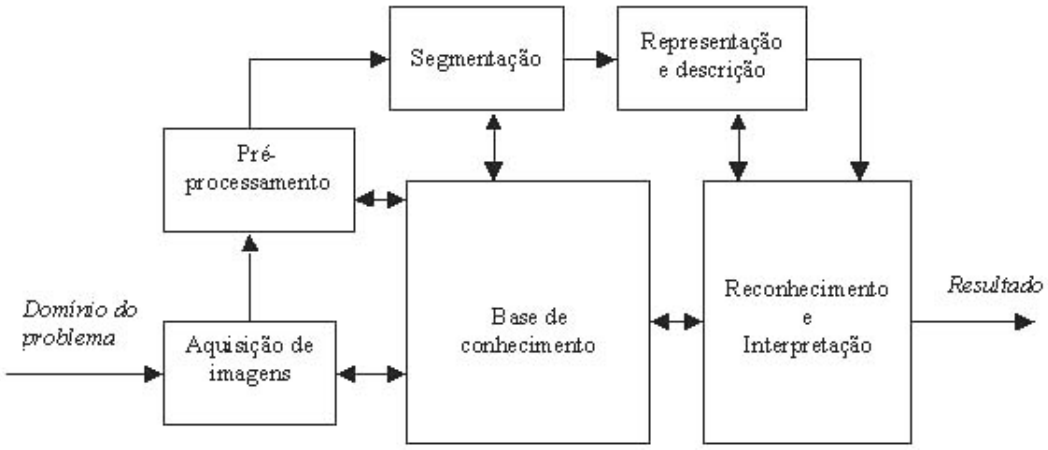


Figura 8 – Passos fundamentais do processamento de imagens (GONZALEZ; WOODS, 2014)

Tabela 2 – Características dos principais processos do processamento de imagens

Processo	**Função**
Aquisição de imagens	Obter imagens, a partir de dados captados por sensores
Pré-processamento	Melhorar a imagem, com realce e diminuição de ruído
Segmentação	Dividir uma imagem em objetos
Representação e descrição	Extrair os atributos que diferenciam classes de objetos
Reconhecimento	Atribuir um rótulo ao objeto, com base em seus descritores

Gonzalez e Woods (2014) classificam tais processos computacionais em três níveis: baixo, médio e alto. Os de baixo nível abragem operações primitivas, como pré-processamento para diminuição do ruído, realce de contraste e aguçamento das imagens. Destacando que, nesse tipo de processo, tanto a entrada quanto a saída são imagens.

Já os processos de nível médio envolvem a segmentação da imagem em regiões ou objetos; a descrição desses objetos, com intuito de reduzi-los em uma forma apropriada para o posterior processamento computacional; e finalmente, a classificação de objetos individuais. Nesse caso, as entradas são imagens, no entanto, as saídas são os atributos extraídos delas.

E o processamento de nível alto consiste em atribuir significado ao conjunto de objetos reconhecidos anteriormente, analisando a imagem e realizando funções cognitivas associadas à visão. Todavia, esse último processo já pode ser considerado pertencente a visão computacional.

2.7 Visão computacional

Backes e Junior (2019) define a visão computacional como sendo a área de estudo que busca transmitir a capacidade de visão para máquinas. Salientando que ao referir-se à visão, não trata-se apenas da captação de imagens, mas também de extrair informações e associá-las com outras imagens vistas anteriormente.

Portanto, a visão computacional é o ramo da ciência e tecnologia responsável por desenvolver aplicações de processamento de imagens. É composta de métodos e técnicas capazes de retirar e interpretar dados provenientes de imagens. Tem diversas finalidades e pode ser aplicada no reconhecimento de padrões, na visão de robôs, na segurança de ambientes e em variadas tarefas que são dependentes da visão biológica.

Atualmente, a pesquisa nessa área direciona-se à diversos algoritmos de alto desempenho designados para soluções específicas. No entanto, ainda não há muito desenvolvimento em algoritmos de âmbito genérico capazes de emular a eficiência da visão biológica. Ressaltando que grande parte dos avanços observados são propiciados pela utilização de redes neurais (BACKES; JUNIOR, 2019).

Isto posto, todos os conceitos apresentados anteriormente serão fundamentais para a execução desse trabalho. O qual, usará processamento de imagens e visão computacional para o reconhecimento de algumas expressões faciais e movimentos de cabeça, codificados pelo FACS na forma de unidades de ação, com o objetivo de interpretar comandos e exercê-los em uma cadeira de rodas elétrica.

2.8 Detecção de face

Primeiramente, é essencial realizar a detecção da face do usuário. Com esse propósito, será aplicado um método amplamente utilizado, baseado em um descritor HOG (*Histogram of Oriented Gradients*) e um classificador SVM. O qual, foi desenvolvido a partir de cinco filtros HOG: vista frontal, lateral esquerda, lateral direita, frontal rotacionada a esquerda e frontal rotacionada a direita. Seu treinamento foi realizado utilizando um conjunto de 2825 imagens provindas do *dataset* LFW (*Labeled Faces in the Wild*) e documentado por Davis King, autor da biblioteca Dlib.

E foi escolhido por ser o método mais rápido de detecção na CPU, possibilitando mais eficiência no reconhecimento em tempo real. Além disso, é capaz de reconhecer faces em diferentes ângulos. Apesar de não ser recomendado na identificação de imagens pequenas, pode ser utilizado para essa aplicação, pois o rosto a ser identificado localiza-se relativamente próximo à câmera (GUPTA, 2018).

Alguns dos principais detectores do Dlib e OpenCV podem ser vistos na Figura 9. Realizando uma comparação entre eles pode-se afirmar que o Haar possui arquitetura

simples, trabalha quase em tempo real usando CPU e é apto a detectar faces em diferentes escalas. No entanto, pode fornecer vários falsos positivos e não funciona em rostos não frontais. Já o método DNN é o mais preciso entre eles, funciona em tempo real e tem habilidade para identificar em diferentes ângulos e escalas. Todavia, é mais lento que o HOG. Logo, o HOG é o mais rápido e leve entre eles. Por fim, o MMOD atua bem para diferentes orientações faciais e seu treinamento é fácil, mas é o mais lento.

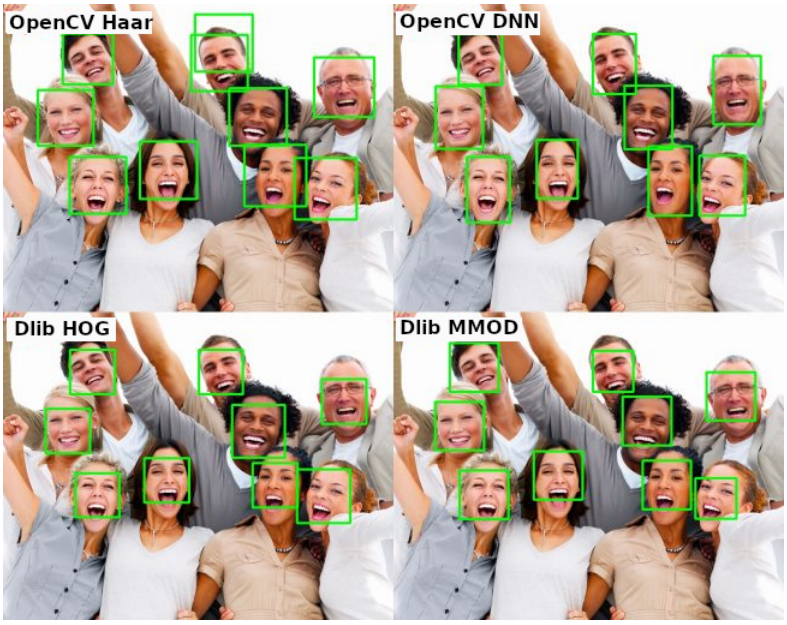


Figura 9 – Comparação entre detectores de face. Adaptado de (GUPTA, 2018)

2.9 Aplicação de landmarks

Após apontar a localização da face, é essencial obter mais informações sobre o rosto do usuário. Então, para atender à esse objetivo, será aplicada a técnica de extração de landmarks. As *landmarks* faciais são usadas para representar regiões de interesse da face, tais como olhos, sobrancelhas, nariz, boca e linha do queixo. Costumam ser utilizadas com diversas finalidades, por exemplo para realizar o alinhamento da face, estimar o posicionamento da cabeça, detectar piscada, dentre outros. E têm como base um preditor de forma.

Existem algumas bibliotecas com capacidade de fazer essa identificação. Dentre elas, optou-se também pela Dlib. A qual, é responsável por distinguir 68 pontos de referência faciais. Os quais são mostrados na Figura 10.

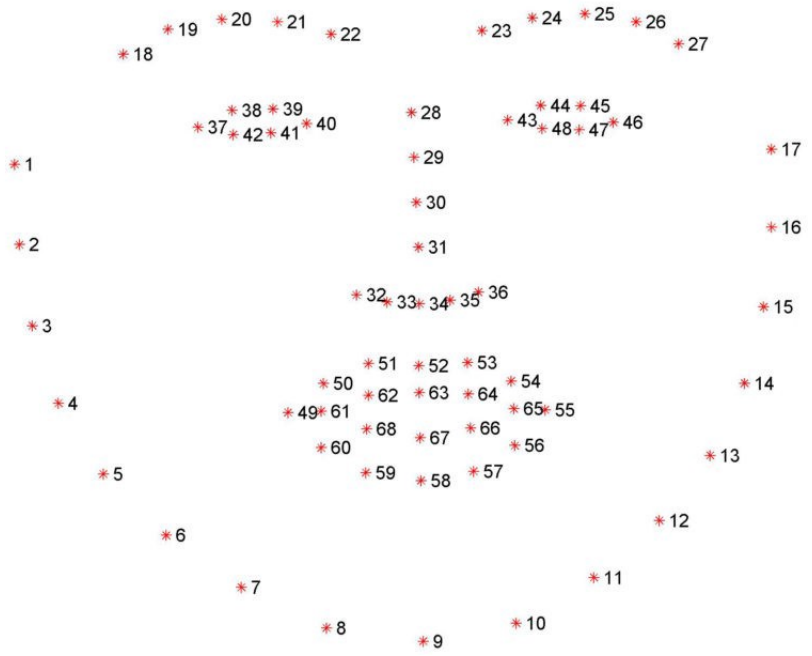


Figura 10 – Landmarks possíveis de ser obtidos com biblioteca Dlib (JOSÉ, 2018)

3 Ferramentas de desenvolvimento

Esse capítulo é dedicado à introduzir e descrever as principais ferramentas computacionais utilizadas no desenvolvimento deste projeto. Assim como esclarecer algumas de suas vantagens e desvantagens de uso. Sendo que, dentre elas estão: Python, OpenCV, Dlib e Tkinter.

3.1 Python

Python é uma linguagem de alto nível orientada a objetos. Borges (2014) a caracteriza como possuidora de sintaxe clara e concisa, o qual possibilita uma melhor legibilidade do código-fonte. Além disso, inclui estruturas de alto nível, tais como listas, data/hora e dicionários; alguns módulos prontos para uso; e a viabilidade de se importar *frameworks* desenvolvidos por terceiros. Essa foi a linguagem escolhida para o desenvolvimento do projeto devido às seguintes vantagens:

- É uma das linguagens mais comuns em aplicações de ciência dos dados;
- Possui documentação bastante completa;
- É multiplataforma, ou seja, é suportado por sistemas Linux, Mac OS ou Windows;
- Tem recursos de extensibilidade, os quais, permitem integrar componentes Java e bibliotecas de C e C++;
- É amplamente utilizada em empresas reconhecidas;
- Possui suporte a bibliotecas de inteligencia artificial, visão computacional e reconhecimento de imagens;

3.2 OpenCV

O OpenCV é uma biblioteca *open source* desenvolvida para visão computacional, processamento de imagem e aprendizagem de máquina. Foi escrito nas linguagens C e C++, funciona no Linux, Windows e MAC OS X e possui interface para Java, Python, C e C++. Foi elaborado com intuito de ser acessível a programadores, ao prover eficiência computacional, sendo focado em aplicações de tempo real. Teve origem em um projeto de pesquisa da Intel no ano de 1998 (OPENCV, 2019).

Essa biblioteca possui mais de 2500 algoritmos otimizados, que inclui desde metodologias mais clássicas até as mais atuais. Como resultado, é muito utilizada por empresas,

grupos de pesquisa e órgãos governamentais. Estima-se que a comunidade de usuários atinja mais de 47 mil pessoas. Sendo assim, é reconhecida como uma ferramenta que auxilia os desenvolvedores a criarem projetos mais completos e robustos.

Sua estrutura é modular, o que significa que os pacotes incluem tanto bibliotecas compartilhadas quanto estáticas. Algumas informações referentes à esses módulos, podem ser analisados no Quadro 3.2 .

Quadro 3.2 - Módulos e funções presentes no OpenCV

Nome do módulo	**Função**
core	define estruturas básicas de dados e funções básicas usadas em todos os outros módulos
imgproc	possui várias técnicas de processamento de imagem
video	é um módulo de análise de vídeo
calib3d	possui funções de calibração da câmera e reconstrução 3D
features2d	tem detectores e descritores de pontos de saliência
objdetect	detecta objetos e instâncias de classes predefinidas
highgui	interface de fácil uso para desenvolver recursos simples de UI
videoio	tem uma interface fácil de usar para captura e codecs de vídeo

3.3 Dlib

O Dlib é uma biblioteca desenvolvida em C++, que posteriormente teve sua API criada para Python, o qual possui licença *open source*. Ele contém algoritmos referentes à aprendizagem de máquina, processamento de imagens e visão computacional. Desse modo, vem sendo utilizado tanto na indústria quanto em ambiente acadêmico, devido ao seu alcance em diferentes domínios, tais como: robótica, dispositivos embarcados, celulares e ambientes de alta performance computacional (KING, 2019).

Dentre suas vantagens, pode-se citar a documentação que é completa, a diversidade de programas de exemplo e o suporte à sistemas Windows, Linux e Mac OS. Entretanto, a razão principal que determinou sua escolha foi a capacidade de identificar *landmarks* da face.

3.4 Tkinter

Tkinter consiste em uma biblioteca referente à linguagem Python, a qual funciona como uma ferramenta para desenvolvimento de interfaces gráficas ou GUI, do inglês *Graphical User Interface*. Está disponível para grande parte da plataforma Unix, assim como, para sistemas Windows e Mac (LUNDH, 1999). E utiliza como licença o PSFL, *Python Software Foundation License*, que é compatível com a licença para software livre GPL (*GNU General Public License*).

Deve-se salientar que há outras ferramentas que desempenham a mesma função. Dentre elas estão: PyGtk, PyQt e wxPython. No entanto, optou-se por essa opção devido à sua simplicidade de uso, nível de intuitividade e o número de recursos disponíveis (POLO, 2017).

4 Implementação

Para solucionar o problema apontado, criou-se um algoritmo de reconhecimento de expressões faciais. Para isso, foram utilizadas algumas funções do OpenCV e do Dlib, devido à necessidade de se implementar um sistema em tempo real. Além disso, optou-se por usar a linguagem Python. O fluxograma de ordem de execução do algoritmo é mostrada na Figura 11. Nela é possível observar os subsistemas que serão utilizados para a criação do algoritmo. E a descrição de cada um deles, pode ser vista nas seções posteriores.

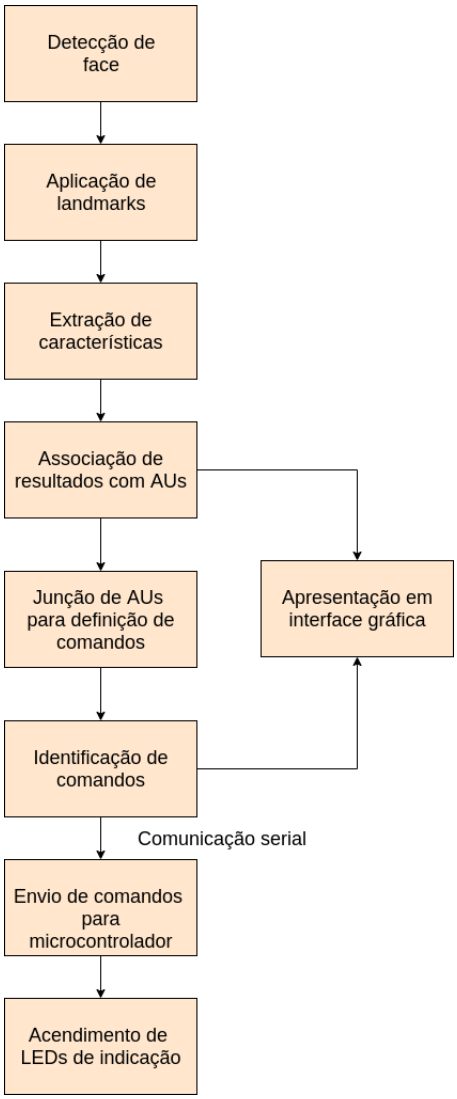


Figura 11 – Fluxograma de ordem de execução do algoritmo

Sendo que primeiramente, para a detecção de face importou-se no Dlib a função *get_frontal_face_detector*. E em seguida para realizar a aplicação de *landmarks* chamou-se a seguinte função: *dlib.shape_predictor("shape_predictor_68_face_landmarks.dat")*.

4.1 Extração de características e classificação

Depois de obter alguns pontos de interesse da face é necessário representá-los de maneira que sirva para o processamento computacional. No geral, pode-se afirmar que há duas formas de realizar essa representação: em termos de suas características externas, ou seja, da fronteira da imagem; ou em relação às suas características internas, portanto, os pixels constituintes da região. Sendo que, a externa costuma ser aplicada quando o foco está nas características da forma e a interna quando está nas propriedades regionais, como a cor e a textura.

Junior et al. (2016) mostra em seu trabalho que métodos que utilizam características geométricas da face possuem desempenho semelhante ou ainda melhor que técnicas que fazem uso de textura. Além disso, possuem como vantagem: a simplicidade e facilidade no processo de reconhecimento; a diminuição do impacto sofrido em condições de mudança de iluminação; e também a menor necessidade de espaço de memória, pois expressam a imagem em um conjunto de vetores de características, consequentemente, diminuindo o número de cálculos para a etapa de classificação. No entanto, tem como desvantagem o fator de possuir menos informações da face completa, e assim, pode trazer mais dificuldades caso seja necessário detectar mudanças sutis.

Posto isso, para a aplicação desse trabalho pretende-se utilizar a descrição com base nas características geométricas. E para isso, será implementado os cálculos de Distância Euclidiana entre os principais pontos do rosto. Sua implementação é dada com base na Equação 4.1.

$$d(p_a, p_b) = ||p_a - p_b|| = \sqrt{\sum (p_a - p_b)^2} = [(p_{a_x} - p_{b_x}) + (p_{a_y} - p_{b_y})]^2 \tag{4.1}$$

Destacando que os pontos escolhidos para esse objetivo devem ser os que são mais capazes de demonstrar a movimentação da face. Desse modo, a relação entre alguns dos landmarks escolhidos e a parte do rosto a ser movimentada, pode ser visto no Quadro 4.1 e a sua representação na face foi exemplificada na Figura 12.

A partir dessas distâncias obtidas é possível criar associações matemáticas para identificar movimentos mais intensos. Como por exemplo, a abertura total da boca, pois as distâncias calculadas serão muito maiores do que com a boca fechada ou em movimento de fala. Outro caso, é a diferenciação do olho aberto para o olho fechado, tornando viável a identificação de piscadas mais demoradas. E assim, classificar movimentos individuais faciais.

Um outro movimento utilizado para esse intuito foi o de inclinar a cabeça para esquerda ou direita. O qual, diferentemente dos outros, ao invés de utilizar a distância

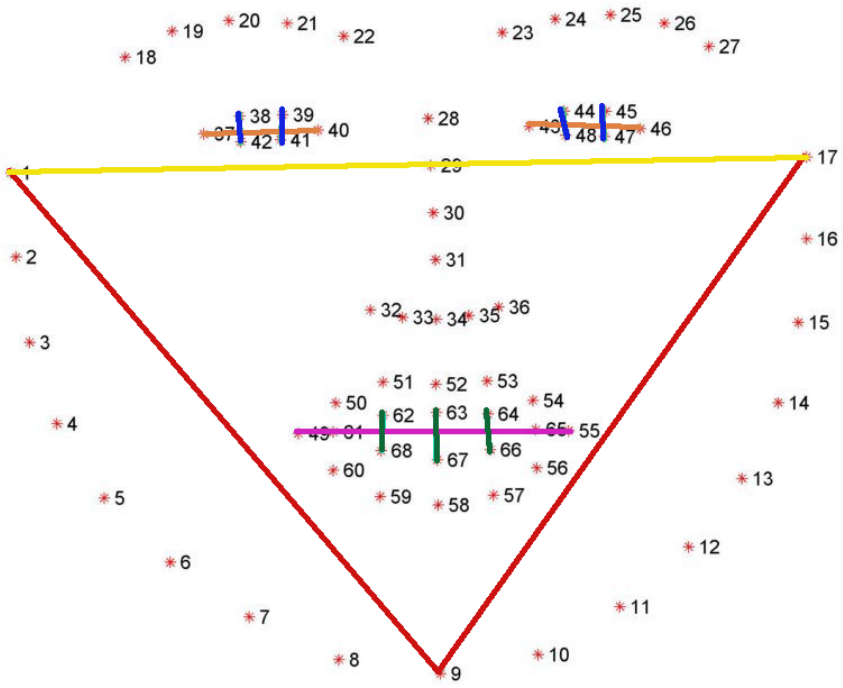


Figura 12 – Exemplificação das distâncias consideradas

Quadro 4.1 - Relação entre a parte do rosto e os pontos a aplicar a Distância Euclidiana

Parte do rosto	**Pontos escolhidos**	**Nomenclatura**
Abertura do olho esquerdo	$d(p_{42}, p_{38}) + d(p_{42}, p_{39}))$	$d_{A_{OE}}$
Comprimento do olho esquerdo	$d(p_{40}, p_{37})$	$d_{C_{OE}}$
Abertura do olho direito	$d(p_{48}, p_{44}) + d(p_{47}, p_{45})$	$d_{A_{OD}}$
Comprimento do olho direito	$d(p_{46}, p_{43})$	$d_{C_{OD}}$
Abertura da boca	$d(p_{68}, p_{62}) + d(p_{67}, p_{63}) + d(p_{66}, p_{64})$	d_{A_B}
Comprimento da boca	$d(p_{55}, p_{49})$	d_{C_B}
Altura da inclinação da cabeça	$d(p_1, p_{17})$ (em função de y)	$d_{A_{IC}}$
Comprimento do rosto	$d(p_9, p_1) + d(p_9, p_{17})$ (em função de y)	d_{C_R}

euclidiana em relação a x e y, usou apenas em relação a y. Essa alteração foi feita, pois a rotação é fundamental nesse caso, não podendo ser invariante como nos outros.

Com intuito de tornar a detecção invariável à escala, relacionou-se as aberturas e alturas encontradas com os comprimentos referentes à parte do corpo analisada. Tal como é mostrado nas equações abaixo. E comparando o resultado obtido com alguns valores fixos, é possível estimar a unidade de ação realizada.

- Olho esquerdo

$$x_{OE} = \frac{d_{A_{OE}}}{2 * d_{C_{OE}}} \tag{4.2}$$

- Olho direito

$$x_{OD} = \frac{d_{C_B}}{2 * d_{A_B}} \tag{4.3}$$

- Boca

$$x_B = \frac{d_{C_B}}{d_{A_B}} \tag{4.4}$$

- Inclinação da cabeça

$$x_{IC} = \frac{100 * d_{A_{IC}}}{d_{C_R}} \tag{4.5}$$

Por fim, após obter a identificação de tais movimentos individuais, por exemplo: piscar do olho esquerdo ou abertura da boca. Será verificado se mais de uma AU foi realizada, e em caso positivo, será checado se a associação entre elas relaciona-se com o conjunto de movimentos que deve movimentar a cadeira ou não. Frisando que se não houver a movimentação de todas os membros necessários para gerar o comando, a cadeira não deve se mover.

4.2 Definição de comandos

É fundamental que ao escolher os movimentos para controlar a cadeira, os mesmos sejam apropriados. Atentando para o fato de que como a câmera estará voltada para o rosto do usuário durante o decorrer do dia, ele realizará expressões faciais comuns. As quais, não podem ser confundidas com as expressões de controle, a fim de evitar movimentações indevidas e até mesmo possíveis acidentes.

Desse modo, exemplificou-se na Tabela 3 associações de comandos com as expressões faciais. De forma que, cada um dos movimentos e a combinação deles foi feita com base no conceito de AUs do sistema FACS, os quais foram explicados na seção 2.4.

Tabela 3 – Definição de comandos para expressões faciais

Comando	**AUs**	**Expressão**
Virar para esquerda	55 + 27	Inclinar a cabeça para esquerda e abrir a boca
Virar para direita	56 + 27	Inclinar a cabeça para direita e abrir a boca
Andar para trás	45 + 27	Abrir a boca e piscar
Andar para frente	28 + 45	Pressionar os lábios e piscar

4.3 Interface Gráfica

Após a identificação das unidades de ação, e posteriormente dos comandos, deve-se apresentar tais resultados para o usuário. E para isso foi desenvolvida uma interface gráfica utilizando o *framework* Tkinter, a qual desenvolve as funções mostradas no diagrama da Figura 13.

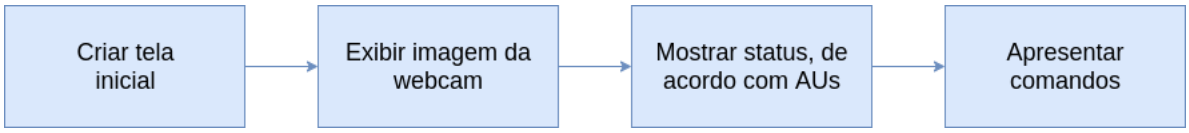


Figura 13 – Diagrama representativo das funções da interface gráfica

No trecho de código visto na Figura 14 pode-se notar a simplicidade de implementação dessa ferramenta. Explanando o funcionamento do código, primeiramente importa-se a biblioteca a ser utilizada. Em seguida, cria-se um objeto chamado *root*, por meio da função Tk(). Após isso, é possível fazer a chamada das funções, passando tal objeto como argumento. Desse modo, utilizou-se a função *label* para inserir textos e imagens na posição desejada.

```
if __name__ == '__main__':
    root=tk.Tk() #assigning root variable for Tkinter as tk
    root.config(background = "#d7f1ef")
    lmain = tk.Label(master=root, bg="white")
    lmain.grid(row=8, column=1, rowspan=1, columnspan=2, sticky=W+E+N+S)
    root.title("Sistema de reconhecimento de expressões faciais")

    ### Image Declarations ###
    photoGreen = PhotoImage(file="green.png")
    photoGreen = photoGreen.subsample(6,6)
    photoRed = PhotoImage(file="red.png")
    photoRed = photoRed.subsample(6,6)
    photoCadeiraDireita = PhotoImage(file="CadeiraDireita.png")
    photoCadeiraEsquerda = PhotoImage(file="CadeiraEsquerda.png")
    photoCadeiraFrente = PhotoImage(file="CadeiraFrente.png")
    photoCadeiraStop = PhotoImage(file="CadeiraStop.png")
    photoCadeiraTras = PhotoImage(file="CadeiraRe.png")
    photoCapa = PhotoImage(file="capaIG2.png")
    photoCapa = photoCapa.zoom(1,1)

    Label (root, image=photoCapa, bg="#d7f1ef").grid(row=0, column=0, columnspan = 3, sticky=W+E)

    Label (root, text="DESCRIÇÃO", bg="#d7f1ef", fg="black", font="helvetica 16 bold") .grid(row=2, column=0, columnspan=1, sticky=W)
    Label (root, text="Abrir amplamente a boca", bg="#d7f1ef", fg="black", font="helvetica 16 ") .grid(row=3, column=0, columnspan=1, sticky=W)
    Label (root, text="Sugar os lábios", bg="#d7f1ef", fg="black", font="helvetica 16 ") .grid(row=4, column=0, columnspan=1, sticky=W)
    Label (root, text="Piscar", bg="#d7f1ef", fg="black", font="helvetica 16 ") .grid(row=7, column=0, columnspan=1, sticky=W)
    Label (root, text="Inclinar cabeça para esquerda", bg="#d7f1ef", fg="black", font="helvetica 16 ") .grid(row=5, column=0, columnspan=1, sticky=W)
    Label (root, text="Inclinar cabeça para direita", bg="#d7f1ef", fg="black", font="helvetica 16 ") .grid(row=6, column=0, columnspan=1, sticky=W)

    Label (root, text="AU", bg="#d7f1ef", fg="black", font="helvetica 16 bold") .grid(row=2, column=1, columnspan=1, sticky=W)
    Label (root, text="27", bg="#d7f1ef", fg="black", font="helvetica 16 ") .grid(row=3, column=1, columnspan=1, sticky=W)
    Label (root, text="28", bg="#d7f1ef", fg="black", font="helvetica 16 ") .grid(row=4, column=1, columnspan=1, sticky=W)
```

Figura 14 – Trecho de código utilizando funções do Tkinter

Após finalizada a interface gráfica é necessário integrá-la com o código de identificação dos comandos, para o funcionamento completo do programa. Sendo assim, optou-se por colocar toda a parte de reconhecimento em uma função e criar um *loop* que faça tanto uma chamada à essa função quanto atualize a interface com os resultados obtidos.

Para realizar a atualização da interface com diferentes imagens, referentes a cada um dos comandos e status, criou-se funções individuais para cada uma, também por meio

da função *label*. Sendo que ao haver a detecção de alguma das AUs predeterminadas e/ou dos comandos, é feita a chamada da função associada.

4.4 Protocolo de comunicação

Após o reconhecimento dos comandos é fundamental executá-los, e para isso deve-se enviar os sinais obtidos para o microcontrolador ou sistema embarcado que realizará o controle da cadeira de rodas. A fim de exemplificar o processo, optou-se por utilizar um Arduino Uno, mostrado na Figura 15 e um conjunto de quatro LEDs para sinalização de cada um dos comandos.

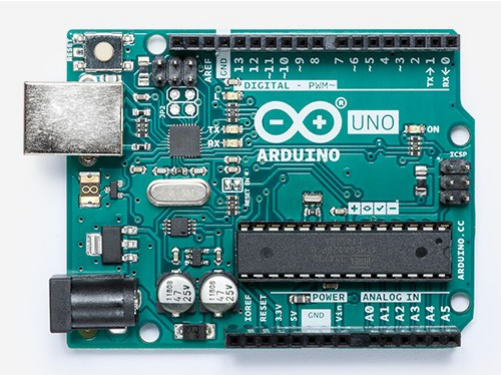


Figura 15 – Arduino Uno (ARDUINO, 2019)

Então, para o envio de dados utilizou-se um protocolo de comunicação serial. Onde a troca de informações ocorre em sequência, entre o computador e o Arduino, via USB. Isso é possível devido à existência de um canal de comunicação por hardware na placa do Arduino Uno, o qual está interligado aos pinos digitais RX, TX e ao microcontrolador ATMEGA16U2, que é responsável pela conversão do sinal para comunicação USB. As configurações dos parâmetros são mostradas na Tabela 4.

Tabela 4 – Configuração de parâmetros do protocolo de comunicação

Parâmetro	**Configuração**
Velocidade de transmissão	9600 baud rate
Quantidade de bits	8
Bit de paridade	Não
Stop bits	1

4.5 Algoritmo executado

Por fim, pode-se exemplificar a lógica do algoritmo por meio do fluxograma presente na Figura 16. Sendo que, a íntegra do código, assim como as instruções de instalação

das ferramentas computacionais, compilação e a solução de alguns possíveis erros estão disponíveis em: <https://github.com/ingridmiranda/TCC>.

De modo que, para toda a implementação e teste do projeto foi utilizado um notebook com processador Intel Core I7, 64-bits, no sistema operacional Ubuntu 16.04 LTS. E a webcam do mesmo para a obtenção das imagens, a qual possui resolução máxima de 1280x720 (0,922MP) e taxa de quadros de 30FPS.

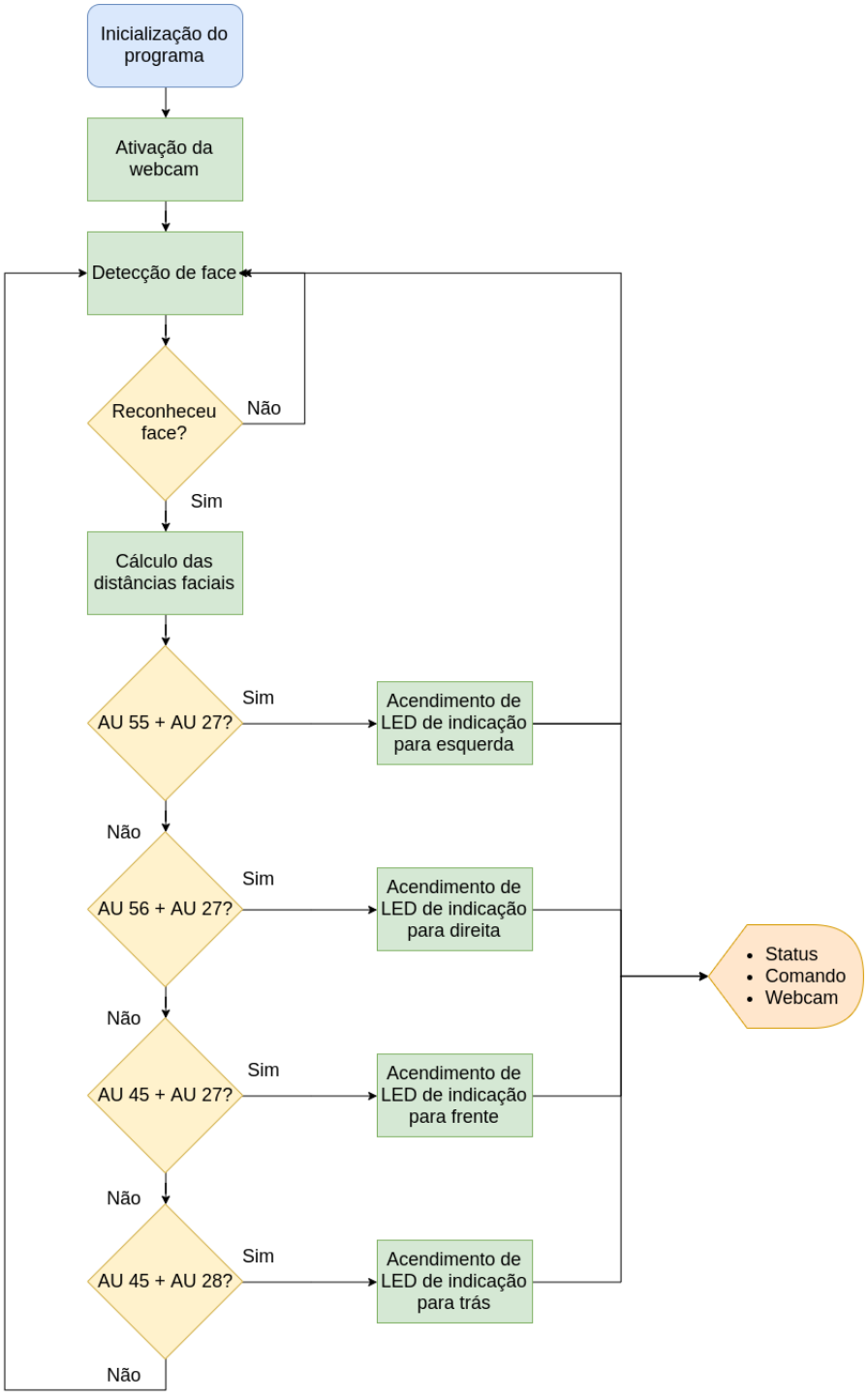
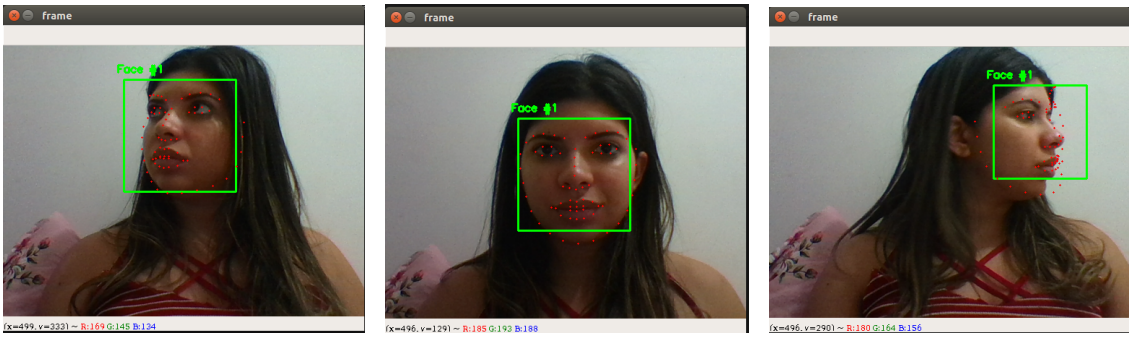


Figura 16 – Fluxograma do algoritmo

5 Resultados

Primeiramente, analisou-se a capacidade de detecção facial HOG do Dlib. E notou-se sua eficiência na detecção de rostos, inclusive quando não estão totalmente voltados para frente. Além disso, o reconhecimento ocorre em aproximadamente 400ms. Portanto, funcionou da maneira almejada para esse projeto.

Após isso, foi feita uma avaliação da capacidade de posicionar corretamente as *landmarks*. Desse modo, observou-se que quando o rosto está situado frontalmente os pontos são captados com exatidão. Entretanto, ao rotacionar a face para os lados há uma dificuldade maior no posicionamento. E assim, os pontos localizam-se erroneamente. Consequentemente, as expressões escolhidas para representar os comandos devem ser todas com o rosto voltado para frente, a fim de evitar movimentos indesejados da cadeira. Salientando que o tempo de execução dessa etapa é de cerca de 800ms. O resultado desses dois primeiros testes, podem ser vistos na Figura 17. Ademais, constatou-se que a presença de óculos também pode distorcer a qualidade da distribuição das *landmarks*.



(a) Para esquerda (b) Para frente (c) Para direita

Figura 17 – Aplicação de landmarks em diferentes ângulos

Nestas duas funções, citadas anteriormente, encontrou-se um ponto fraco relacionado à luminosidade do ambiente. Sendo que, quando o rosto está nítido, a detecção e o posicionamento são feitos nos tempos citados anteriormente. Porém, quando não está incindindo luz suficiente ou todo o sistema está contraluz, há incapacidade no reconhecimento. Logo, nesse caso, não é possível efetuar o processamento, impossibilitando a execução de comandos.

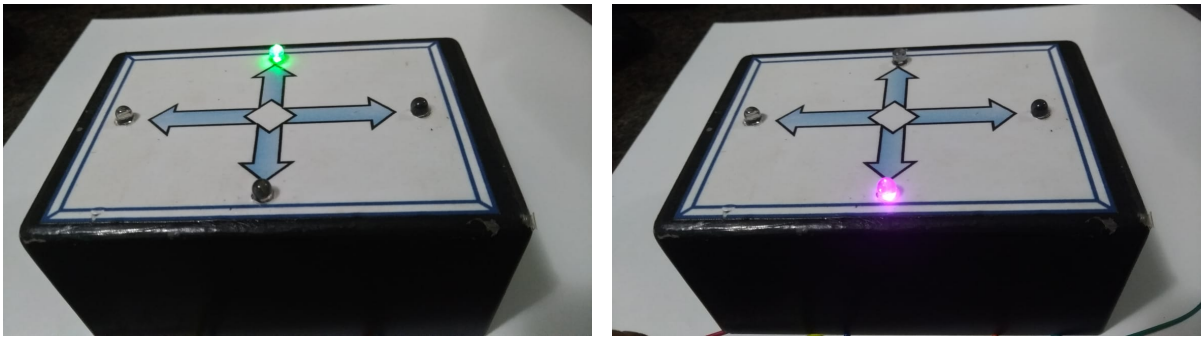
A próxima etapa consistiu em aplicar expressões matemáticas entre as *landmarks*, com intuito de distinguir as unidades de ação. Para isso, foram utilizadas as distâncias calculadas tal como na Tabela 4.1 e realizados os cálculos vistos nas equações 4.2 a 4.5. A partir disso, constatou-se experimentalmente o intervalo de valores para cada um dos movimentos individuais, onde, os mesmos estão dispostos na Tabela 5.

Tabela 5 – Valores de identificação das AUs

Variável	**Intervalo de valores**	**Movimento associado**
x_{OE}	< 0.20	Olho esquerdo piscando
x_{OD}	< 0.20	Olho direito piscando
x_B	< 6	Boca aberta
x_B	> 40	Boca comprimida
x_{IC}	$>= 40$	Cabeça inclinada para esquerda
x_{IC}	$<= -40$	Cabeça inclinada para direita

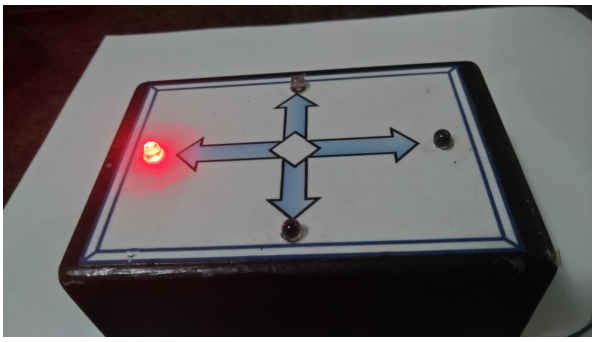
Então, depois de efetivada e testada a identificação das unidades de ação, pôde-se criar condições de associação das AUs e comparação com os comandos. E mais uma vez, foram feitos testes replicando os movimentos de frente para a *webcam* e atentando-se aos *logs* do código, com o propósito de ver a eficácia da detecção e fazer ajustes, caso necessário.

À medida que os resultados tornaram-se consistentes, pôde-se enviar sinais via serial para o Arduino. Os quais, nesse primeiro momento, serviram para acenderem LEDs de indicação do movimento, bem como exibido na Figura 18. Mas, que posteriormente, podem ser replicados para realizar o controle dos motores da cadeira. Evidenciando que todo esse processo após a aplicação de *landmarks* é executado em 200ms.

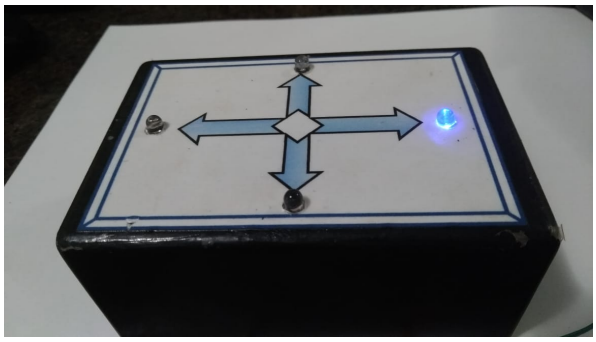


(a) Frente

(b) Trás



(c) Esquerda



(d) Direita

Figura 18 – LEDs de indicação

Para projetar o circuito é fundamental ter conhecimentos das tensões e correntes associadas aos LEDs e às portas digitais do Arduino, para isso pôde-se consultar as Tabelas 6 e 7.

Tabela 6 – Informações de tensão e corrente para LEDs

Coloração do LED	**Tensão de operação (V)**	**Corrente de operação (mA)**
Azul	2,5 - 3,0	20
Rosa	2,5 - 3,0	20
Verde	2,0 - 2,5	20
Vermelho	1,8 - 2,0	20

Tabela 7 – Especificações das portas digitais do Arduino

Parâmetro	**Configuração**
Número de portas	14
Tensão de operação	5V
Corrente máxima de operação	40mA

E a partir disso, utilizar a Lei de Ohm, mostrada na Equação 5.1. E assim, encontrar o valor adequado do resistor que será associado ao LED.

$$V = R \cdot I \tag{5.1}$$

Sendo que para essa aplicação pode-se manipulá-la, chegando à Equação 5.2.

$$R_{LED} = \frac{V_{PD_{Arduino}} - V_{LED}}{I_{LED}} \tag{5.2}$$

Com isso, obteve-se os valores exibidos na Tabela 8. No entanto, é importante evidenciar que se houver dificuldades em encontrar tais valores no mercado, pode-se utilizar algum outro que seja aproximado. Nesse caso, optou-se por usar dois com 100Ω e dois com 150Ω.

Tabela 8 – Intervalo de resistores indicados para cada LED

Coloração do LED	**Resistor associado (Ω)**
Azul	100 - 125
Rosa	100 - 125
Verde	125 - 150
Vermelho	150 - 160

Logo, o esquemático do circuito responsável pelo acendimento dos LEDs pode ser visto na Figura 19.

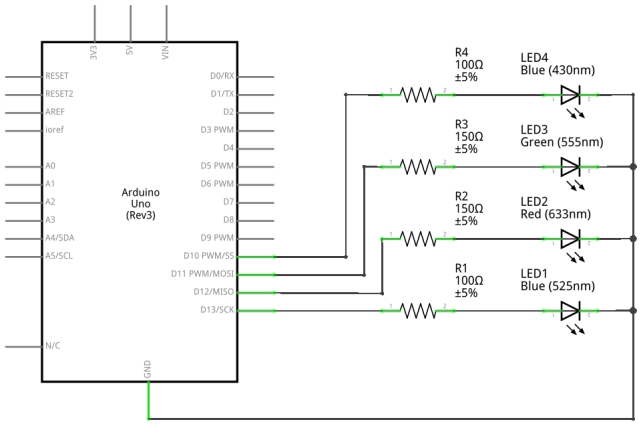


Figura 19 – Esquemático do circuito utilizando Software Fritzing

Para a transmissão dos comandos do computador para o microcontrolador utilizou-se o envio de caracteres, com os valores de 1 a 4, para exemplificar cada um dos comandos. Todavia, para o recebimento e comparação no Arduino, pode-se utilizar a codificação ASCII. Tal relação é exibida na Tabela 9.

Tabela 9 – Relação entre transmissão de comandos

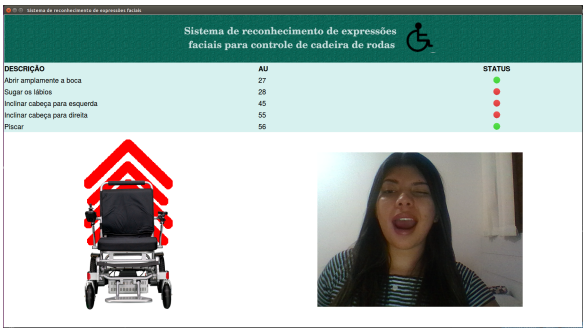
Comando	**Caractere**	**Codificação ASCII**
Virar para direita	1	49
Virar para esquerda	2	50
Seguir em frente	3	51
Ir para trás	4	52

Por fim, integrou-se todo o sistema com a interface gráfica. E notou-se que a mesma é capaz de atualizar e mostrar, tanto os status de cada uma das cinco unidades de ação quanto uma imagem representativa do movimento que a cadeira irá realizar, caso algum comando seja detectado. Sendo que o programa é iniciado com a cadeira parada, tal como mostrado na Figura 20.

Figura 20 – Interace gráfica sem comando reconhecido

E todas as possíveis movimentações a serem detectadas, são exibidas na Figura 21. Ou seja, as expressões necessárias para enviar comandos para frente, trás, esquerda e direita.

(a) Frente



(b) Trás

(c) Esquerda

(d) Direita

Figura 21 – Demonstração dos comandos

No entanto, deve-se ressaltar que apesar da eficácia do sistema em ambientes adequadamente luminosos. Ao inserir a interface gráfica, foi gerado um atraso de aproximadamente três segundos entre o tempo de execução do movimento e de identificação e exibição. Porém, observou-se que o tempo de processamento foi o mesmo (200ms) e o de atualização foi de 20ms. Logo, o atraso é proveniente da própria interface.

6 Considerações finais

Os resultados obtidos, por meio dos testes experimentais, mostram que o sistema é capaz de detectar as cinco unidades de ação escolhidas. E consequentemente, a associação delas, ou seja, os quatro comandos completos: frente, trás, esquerda e direita. Entretanto, devido à baixa qualidade da câmera encontrou-se alguns problemas relacionados principalmente à luminosidade. Portanto, sugere-se que em trabalhos futuros invista-se em uma câmera melhor ou em um sistema de iluminação em frente ao usuário, realizando previamente um estudo a fim de não causar desconforto ao mesmo.

Com intuito de tornar o sistema mais compacto, pode-se substituir o computador por um sistema embarcado que realize todo o processamento. Sendo que o mesmo pode continuar enviando os dados à um microcontrolador que faça o controle da cadeira, ou ainda, migre tais funções para que ele próprio a controle.

Com a presença da interface gráfica obteve-se um atraso na resposta. No entanto, ao tornar o projeto mais compacto não haverá tela de exibição. Logo, é plausível retirá-la do sistema e manter, por exemplo, os LEDs de indicação para apresentação ao usuário. Assim, diminuindo o tempo de execução do algoritmo.

Além disso, para diminuir o número de falsos positivos e negativos recomenda-se aumentar a complexidade do sistema. Aumentando o número de unidades de ação associadas e até mesmo incluindo conceitos de redes neurais. E por fim, deve-se integrar e implementar testes na própria cadeira de rodas.

Conclui-se, portanto, que este trabalho alcançou todos os objetivos propostos inicialmente. Todavia, posteriormente, convém realizar aperfeiçoamentos, a fim de minimizar as falhas existentes e tornar essa interface de controle mais completa e segura para o usuário.

Referências

ARAÚJO, M. V. de et al. Interface homem-máquina baseada em eog para seleção de movimentos de uma órtese ativa. Congresso Brasileiro de Automática. Belo Horizonte, 2010. Citado na página 31.

ARDUINO. *ARDUINO UNO REV3*. 2019. Disponível em: <https://store.arduino.cc/usa/arduino-uno-rev3>. Citado 2 vezes nas páginas 15 e 52.

BACKES, A. R.; JUNIOR, J. J. d. M. S. *Introdução à visão computacional usando Matlab*. [S.l.]: Alta Books Editora, 2019. Citado na página 39.

BAREA, R. et al. System for assisted mobility using eye movements based on electrooculography. *IEEE transactions on neural systems and rehabilitation engineering*, IEEE, v. 10, n. 4, p. 209–218, 2002. Citado 2 vezes nas páginas 15 e 32.

BERSCH, R.; TONOLLI, J. Introdução ao conceito de tecnologia assistiva e modelos de abordagem da deficiência. *Bengala Legal*, 2006. Citado na página 25.

BERTONCELLO, I.; GOMES, L. V. N. Análise diacrônica e sincrônica da cadeira de rodas mecanomanual. *Revista Produção*, SciELO Brasil, v. 12, n. 1, p. 72–82, 2002. Citado na página 25.

BORGES, L. E. *Python para desenvolvedores: aborda Python 3.3*. [S.l.]: Novatec Editora, 2014. Citado na página 43.

BRANCO, M. D. C. C. Instrumentação de cadeira de rodas motorizada para usuários com tetraplegia. Brasília, 2018. Citado na página 27.

BROMLEY, I. *Tetraplegia and paraplegia: a guide for physiotherapists*. [S.l.]: Elsevier Health Sciences, 2006. Citado na página 25.

CLINIC, B. P. *Muscles of facial expressions and how they work*. 2019. Disponível em: <https://crystal-touch.nl/muslces-of-facial-expressions-and-how-they-work/>. Acesso em: 02 jun. 2019. Citado 2 vezes nas páginas 15 e 35.

COWAN, R. E. et al. Recent trends in assistive technology for mobility. n. Figure 1, p. 1–8, 2012. Citado na página 27.

EKMAN, R. *What the face reveals: Basic and applied studies of spontaneous expression using the Facial Action Coding System (FACS)*. [S.l.]: Oxford University Press, USA, 1997. Citado na página 35.

FEHR, L.; LANGBEIN, W. E.; SKAAR, S. B. Adequacy of power wheelchair control interfaces for persons with severe disabilities: A clinical survey. *Journal of rehabilitation research and development*, REHIBILITATION RESEARCH & DEVELOPMENT SERVICE, v. 37, n. 3, p. 353–360, 2000. Citado na página 27.

FERREIRA, C. L. L. Interface de sopro e sucção para controle de cadeira de rodas. *Paraná, Brazil*, p. 64, 2008. Citado 2 vezes nas páginas 15 e 33.

FILHO, O. M.; NETO, H. V. *Processamento digital de imagens*. [S.l.]: Brasport, 1999. Citado na página 37.

FILHO, W. d. B. V. e. a. Desenvolvimento de kit para automação de cadeira de rodas convencional. VI Congresso Nacional de Engenharia Mecânica, Campinha Grande-PB, 2010. Citado na página 27.

GONZALEZ, R. C.; WOODS, R. C. *Processamento digital de imagens* . [S.l.]: Pearson Educación, 2014. Citado 2 vezes nas páginas 15 e 38.

GUPTA, V. Face detection - opencv, dlib and deep learning (c++/python). *Learnopencv.com*, 2018. Disponível em: <https://www.learnopencv.com/face-detection-opencv-dlib-and-deep-learning-c-python/>. Citado 3 vezes nas páginas 15, 39 e 40.

HOSPINET. *CADEIRA DE RODAS SIMPLES PROLIFE PL001*. 2016. Disponível em: <https://www.hospinet.com.br/cadeira-de-rodas-simples-prolife-pl001/p>. Acesso em: 19 mai. 2019. Citado 2 vezes nas páginas 15 e 26.

HYPENESS. *Microsoft Brasil cria cadeira de rodas controlada pelo movimento dos olhos*. 2018. Disponível em: <https://www.hypeness.com.br/2018/03/microsoft-brasil-cria-cadeira-de-rodas-controlada-pelo-movimento-dos-olhos/>. Acesso em: 19 mai. 2019. Citado 2 vezes nas páginas 15 e 27.

IBGE, I. B. de Geografia e E. *Censo Demográfico 2010 - Características gerais da população, religião e pessoas com deficiência*. 2010. Disponível em: <https://biblioteca.ibge.gov.br/visualizacao/periodicos/94/cd_2010_religiao_deficiencia.pdf>. Acesso em: 16 mai. 2019. Citado na página 25.

IVO, R. M. Sistema de controle de cadeira de rodas motorizada para usários portadores de tetraplegia. Brasília, 2017. Citado na página 27.

JOSÉ, I. *Mapeamento facial (landmarks) com Dlib + python*. 2018. Disponível em: <https://medium.com/brasil-ai/mapeamento-facial-landmarks-com-dlib-python-3a200bb35b87>. Acesso em: 17 jun. 2019. Citado 2 vezes nas páginas 15 e 41.

JUNIOR, J. d. A. F. et al. Reconhecimento automático de expressões faciais baseado em características geométricas. Universidade Federal de Sergipe, 2016. Citado 2 vezes nas páginas 34 e 48.

KING, D. Dlib c++ library. 2019. Disponível em: <https://dlib.net>. Acesso em: 20 ago. 2019. Citado na página 44.

LEÃO, L. P. et al. Detecção de expressões faciais: uma abordagem baseada em análise do fluxo óptico. *Revista GEINTEC-Gestão, Inovação e Tecnologias*, v. 2, n. 5, p. 472–489, 2012. Citado na página 34.

LUNDH, F. An introduction to tkinter. 1999. Disponível em: <www.pythonware.com/library/tkinter/introduction/index.htm>. Acesso em: 18 nov. 2019. Citado na página 45.

MACHADO, S. et al. Interface cérebro-computador. *Revista Neurociências*, v. 17, n. 4, p. 329–235, 2009. Citado na página 33.

MIRANDA, J. C. P. et al. Sistema de visão para controlo de cadeira de rodas inteligente: um sistema de visão adaptado para reconhecimento de expressões faciais com algoritmos inteligentes. 2009. Citado na página 34.

OPENCV. *About OpenCV*. 2019. Disponível em: <https://opencv.org/about/>. Acesso em: 08 jun. 2019. Citado na página 43.

ORTOPONTO. *Cadeira de Rodas Motorizada Elétrica E4 ULX Ortobras com Encosto Rígido Hummel*. 2017. Disponível em: <https://www.ortoponto.com.br/produto/cadeira-de-rodas-motorizada-eletrica-e4-ulx-ortobras-com-encosto-rigido-hummel-666>. Acesso em: 19 mai. 2019. Citado 2 vezes nas páginas 15 e 26.

POLO, G. *PyGTK, PyQT, Tkinter and wxPython comparison*. 2017. Citado na página 45.

ASSOCIAÇÃO DE APOIO AOS PORTADORES DE NECESSIDADES ESPECIAIS E DA COMUNIDADE DO DISTRITO FEDERAL; TIPO D DESIGN INDUSTRIAL LTDA; FUNDAÇÃO UNIVERSIDADE DE BRASÍLIA. M. L. Siqueira W. de B. V. Filho D. M. M. Arboleda R. A. Pereira G. de S. L. Queiroga M. L. de L. Torres; J. C. M. Franco C. H. L. Quintero R. H. Van Els. *Kit para motorização de cadeira de rodas manuais*. 2017. MU 8903008-7. Citado na página 27.

SANTIAGO, H. d. C. Reconhecimento de expressões faciais utilizando estimação de movimento. Universidade Federal de Pernambuco, 2017. Citado na página 35.

SIMPSON, R. C.; LEVINE, S. P. Voice control of a powered wheelchair. *IEEE Transactions on neural systems and rehabilitation engineering*, IEEE, v. 10, n. 2, p. 122–125, 2002. Citado na página 32.

SOLOMON, C.; BRECKON, T. *Fundamentos de processamento digital de imagens: uma abordagem prática com exemplos em Matlab*. [S.l.]: Grupo Gen-LTC, 2013. Citado na página 37.

SOUSA, A. L. d. et al. Reconhecimento de expressões faciais e emocionais como método avaliativo de aplicações computacionais. *Encontro Regional de Computação e Sistemas de Informação-ENCOSIS*, Universidade do Estado do Pará, 2016. Citado na página 34.

TANAKA, K.; MATSUNAGA, K.; WANG, H. O. Electroencephalogram-based control of an electric wheelchair. *IEEE transactions on robotics*, IEEE, v. 21, n. 4, p. 762–766, 2005. Citado na página 33.

WHAT-WHEN-HOW. *Facial Expression Recognition (Face Recognition Techniques) Part 1*. 2012. Disponível em: <http://what-when-how.com/face-recognition/facial-expression-recognition-face-recognition-techniques-part-1/>. Acesso em: 13 jun. 2019. Citado 2 vezes nas páginas 15 e 36.