



UNIP - UNIVERSIDADE PAULISTA
CIÊNCIAS DA COMPUTAÇÃO

SARINA ELLEN RODRIGUES

**DESENVOLVIMENTO DE TECLADO VIRTUAL PARA AUXÍLIO NA
COMUNICAÇÃO DE PESSOAS COM TETRAPLEGIA DO TIPO
COMPLETA.**

LIMEIRA-SP

2023

SARINA ELLEN RODRIGUES

**DESENVOLVIMENTO DE TECLADO VIRTUAL PARA AUXÍLIO NA
COMUNICAÇÃO DE PESSOAS COM TETRAPLEGIA DO TIPO
COMPLETA.**

Trabalho de conclusão de curso para
obtenção do título de graduação em
Ciências da Computação apresentado à
Universidade Paulista – UNIP.

Orientador(a): Prof. Dr. Danilo Rodrigues
Pereira.

LIMEIRA-SP

2023

CIP - Catalogação na Publicação

Rodrigues, Sarina Ellen
DESENVOLVIMENTO DE TECLADO VIRTUAL PARA AUXÍLIO NA
COMUNICAÇÃO DE PESSOAS COM TETRAPLEGIA DO TIPO COMPLETA. /
Sarina Elle Rodrigues. - 2023.
00 f. : il.

Trabalho de Conclusão de Curso (Graduação) apresentado ao Instituto
de Ciência Exatas e Tecnologia da Universidade Paulista, Limeira, 2023.

Área de Concentração: Protótipo.

Orientador: Prof. Dr. Danilo Rodrigues Pereira.

1. Teclado Virtual. 2. Tetraplegia. 3. Comunicação. I. Pereira,
Danilo Rodrigues (orientador). II., André, Amaury (coorientador). III. Título.

SARINA ELLEN RODRIGUES

**DESENVOLVIMENTO DE TECLADO VIRTUAL PARA AUXÍLIO NA
COMUNICAÇÃO DE PESSOAS COM TETRAPLEGIA DO TIPO COMPLETA.**

Trabalho de conclusão de curso para
obtenção do título de graduação em
Ciências da Computação apresentado à
Universidade Paulista – UNIP.

Aprovado(a) em: ____/____/____

BANCA EXAMINADORA

Prof. ou Profa. Dr(a)/Me(a). _____

Universidade Paulista - UNIP

Prof. ou Profa. Dr(a)/ Me(a). _____

Universidade Paulista - UNIP

Prof. ou Profa. Dr(a)/ Me(a). _____

Universidade Paulista - UNIP

DEDICATÓRIA

Dedico este trabalho aos meus amigos que me deram forças e me apoiaram, em especial ao meu orientador pelo apoio contínuo durante todo período de estudo e me indicando que direção tomar. E nesta finalização com sua valiosa contribuição que fez toda diferença.

*“Temos de fazer o melhor que podemos, esta é a nossa sagrada
responsabilidade humana”.*

Albert Einstein

RESUMO

A tetraplegia completa é uma das lesões medulares mais devastadoras que podem ocorrer, a pessoa perde autonomia física e fica dependente de terceiros para as suas atividades diárias que antes era simples. A transição de uma vida independente para um estado de dependência exige uma redefinição do seu estilo de vida, viver com uma tetraplegia é um processo complexo que necessita de adaptação do indivíduo e sua família. E apesar de manter as capacidades mentais e intelectuais, a pessoa com tetraplegia completa também apresentam comprometimento da fala, dificultando sua comunicação.

Frente às limitações enfrentadas, a utilização de computadores e smartphones é basilar para que essas pessoas tenham acesso à informação. O uso de computadores também pode possibilitar que pessoas com ausência de movimentos de membros superiores e inferiores, desfrutem de momentos particulares, pois , na grande maioria, estão acompanhados de um cuidador.

Assim, a produção de texto em um computador, uma tarefa comum, pode ser afetada para estas pessoas acometidas por distúrbios neuro motores severos, por vezes piorado à síndrome do encarceramento¹, pois dispõem exclusivamente dos movimentos dos olhos como forma de comunicação com o mundo exterior. Deste modo, este trabalho propõe a elaboração de um teclado virtual ocular que usa a detecção do piscar do olho como uma ação para a entrada de texto, reproduzindo o desenvolvimento da interface do teclado virtual, com varredura das letras no algoritmo e a fala automatizada ao final de cada palavra digitadas.

A metodologia deste trabalho, contém pesquisa bibliográfica sobre a entrada de informação através da detecção do piscar e desenvolvimento de teclados virtuais.

Temos por resultado alcançado uma precisão boa na captação das letras, e ate uma velocidade de digitação consideravelmente boa, pois o teclado foi confeccionado com uma ordem de letras estudada e impressas de acordo com o que se mais utiliza para a escrita portugues hoje em dia.

Palavras-chave: Tetraplegia Completa. Comunicação. Teclado Virtual

¹ A síndrome do encarceramento costuma ser causada por acidente vascular cerebral, mas pode ser causada por síndrome de Guillain-Barré ou por um câncer que afete uma certa parte do cérebro.

ABSTRACT

Complete Tetraplegia is one of the most devastating spinal cord injuries that can occur, the person loses physical autonomy and becomes dependent on others for their daily activities, which were previously simple. The transition from an independent life to a state of dependence requires a redefinition of your lifestyle. Living with quadriplegia is a complex process that requires adaptation by the individual and their family. And despite maintaining mental and intellectual capabilities, people with complete tetraplegia also have speech impairment, making communication difficult.

Given the limitations faced, the use of computers and smartphones is essential for these people to have access to information. The use of computers can also enable people with lack of movement in their upper and lower limbs to enjoy private moments, as, in the vast majority, they are accompanied by a caregiver.

Thus, text production on a computer, a common task, can be affected for these people affected by severe neuromotor disorders, sometimes worsening to locked-in syndrome, as they exclusively use eye movements as a form of communication with the outside world. . Therefore, this work proposes the development of a virtual ocular keyboard that uses eye blink detection as an action for text entry, reproducing the development of the virtual keyboard interface, with letter scanning in the algorithm and automated speech at the same time. end of each word typed.

The methodology of this work contains bibliographical research on information entry through blink detection and development of virtual keyboards.

As a result, we have achieved good precision in capturing letters, and even a considerably good typing speed, as the keyboard was made with a studied order of letters and printed in accordance with what is most used for writing Portuguese today.

Keywords: Complete Tetraplegia. Communication. Virtual Keyboard.

¹ Locked-in syndrome is usually caused by a stroke, but it can be caused by guillain-barré syndrome or cancer that affects a certain part of the brain.

LISTA DE ILUSTRAÇÕES

Figura 1 – Níveis Medulares.....	9
Figura 2 – Funcionamento Mouse Ocular 1.....	15
Figura 3 - Tobii PCEye Plus 1.....	16
Figura 4 – Site Teclado Virtual Telepati 1.....	16
Figura 5 - Fluxograma Arquitetura do Sis 1.....	18
Figura 6 - Modelo de shape predictor 1.....	22
Figura 7 – Pontos Faciais do Modelo 1.....	22
Figura 8 - Linhas para cálculo piscar 1.....	23
Figura 9 – Layout Teclado Virtual Padrão 1.....	24
Figura 10 – Tabela de Resultados 1.....	24
Figura 11 – Sequência Letras Teclado Vir 1.....	25
Figura 12 – Teste Captura Video em Tempo 1.....	31
Figura 13 – Desenho Linhas na Soma Pisca 1.....	32
Figura 14 – Detectando Piscar 1.....	33
Figura 15 – Teste Varredura 1.....	34
Figura 16 – Integração piscar 1.....	35
Figura 17 – Teste Final 1.....	36

SUMÁRIO

1. INTRODUÇÃO	9
2. OBJETIVO	11
2.1 Objetivos específicos	11
2.2 Justificativa	11
3. MATERIAL E METODOLOGIA	13
3.1 Cronograma de Atividades	13
4. FUNDAMENTAÇÃO TEÓRICA	14
4.1 Tetraplegia e suas limitações	14
4.2 Tecnologia assistiva	14
5. DESENVOLVIMENTO	16
5.1 Arquitetura do sistema	17
5.2 Técnicas	18
5.2.1 Python	18
5.2.1.1 Bibliotecas do Python	19
5.3 Abertura da câmera e a transmissão da imagem em tempo real	20
5.3.1 Conversão das imagens para tons de cinza	20
5.4 Detecção dos olhos e o piscar	21
5.4.1 Uso do Arquivo " <i>shape_predictor</i> " no Projeto	21
5.4.2 O cálculo do piscar	22
5.5 Posicionamento das teclas e lógica na seleção letras para construção do teclado virtual	23
5.6 Unindo o piscar dos olhos com o teclado	25
5.7 A caixa de texto e demais funções	27
5.8 Botões especiais	27
5.8.1 Botão backspace (representado por <-)	27
5.8.2 Botão voltar (representado pelo símbolo <)	28
5.8.3 Botão espaço (representado pelo _)	28
5.8.4 Botão del (localizado na última linha)	28
5.8.5 Intenção por trás dos botões especiais	28
6. TESTES	30
6.1 Conclusão dos testes	36
7. RESULTADOS	37
8. LIMITAÇÕES E TRABALHOS FUTUROS	38
9. CONCLUSÃO	39
10. REFERÊNCIAS BIBLIOGRÁFICAS	40

11. ANEXO44

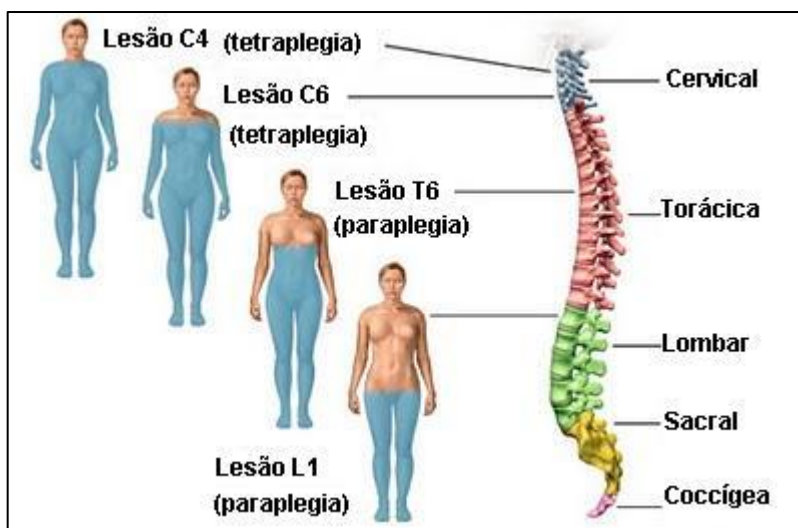
1. INTRODUÇÃO

De acordo com a Organização Mundial da Saúde (OMS) , “Qualidade de vida é a percepção do indivíduo de sua inserção na vida, no contexto da cultura e sistemas de valores nos quais ele vive e em relação aos seus objetivos, expectativas, padrões e preocupações” (OMS, 2013). Trata-se de uma definição que envolve o bem estar espiritual, físico, mental, psicológico e emocional, além de relacionamentos sociais, como família e amigos e, também, saúde, educação, habitação saneamento básico e outras circunstâncias da vida. Esta qualidade é totalmente comprometida nas pessoas com Tetraplegia.

De acordo com Dr. Henrique Noronha (2023), em seu site, “A Tetraplegia acontece quando ocorre uma lesão na medula espinhal a nível de coluna cervical resultando normalmente em uma paralisia que afeta o movimento dos braços, tronco e pernas (Figura 1). Esta perda dos movimentos pode ter intensidade diferentes, porém na Tetraplegia Completa ocorre a perda total da capacidade de movimentar um membro.

A extensão desta lesão pode ser classificada como, completa, que é quando não se tem atividade motora voluntária e nem sensibilidade, e em incompleta que pode existir atividade motora voluntária e sensibilidade parcial. Já pelos níveis da lesão quanto mais alta ela for, maior o comprometimento corporal como por exemplo afetando a fala e movimento do pescoço e cabeça”.

Figura 1– Níveis Medulares



Pessoas com Tetraplegia dependem de outras para poderem realizar suas atividades rotineiras, suas vontades, e isto acaba resultando em uma vulnerabilidade e privação social.

Atualmente existem projetos diversos desenvolvidos para melhorar esta qualidade de vida, os produtos vão desde softwares até artigos de robótica. O desenvolvimento tecnológico não tem revolucionado apenas o mercado de eletrônicos mas também vários equipamentos utilizados por pessoas com deficiência física e motora (Hirata, 2018). E indo nesta direção o presente projeto se propõe a partir de levantamentos com análise de informações, e através de recursos tecnológicos, fornecer uma possibilidade de acesso à uma vida social com melhor qualidade. Com o simples fato de conseguir se comunicar com pessoas ao seu redor, conseguimos favorecer a execução de atividades básicas rotineiras e a sua qualidade de vida melhora muito.

2. OBJETIVO

O objetivo deste trabalho é desenvolver um teclado virtual para comunicação por meio do piscar do olho para pessoas com Tetraplegia Completa, ou seja, limitações, impossibilidades na fala e com movimentos limitados, obtidos através de sequelas decorridas de Traumas de Acidentes, Esclerose Lateral Amiotrófica, Paralisia Cerebral, Acidente Vascular Cerebral / Encefálico dentre outros.

O sistema captará e converterá o piscar dos olhos em letras, e a partir disto a frase formada será “falada” de maneira robótica tornando a comunicação possível e prática. O que será de muita utilidade também para as pessoas ao seu redor que poderão ajudá-los de forma mais eficiente e também terão a ciência de que, apesar de suas impossibilidades estas pessoas conseguem sim raciocinar.

Proporcionando esta facilidade, de terem simples conversas, tudo se tornará mais acessível, e isto trará a liberdade para realizarem ações simples como transmitir suas emoções, sentimentos e aquilo que desejam.

2.1 Objetivos específicos

- Estudar e coletar dados sobre tetraplegia e suas escalas de níveis de deficiência motoras;
- Analisar aplicativos já existentes e sua utilidade neste projeto;
- Desenvolver um teclado virtual com uma interface amigável e de uso fácil.

2.2 Justificativa

Desde o princípio dos tempos a comunicação é a base da socialização e é de vital importância em nossas vidas. Portanto, não ter ou perder esta capacidade é com certeza uma das maiores dores causadas por traumas e também por algumas doenças.

Traumas e doenças estes que dependendo da gravidade resultam em tetraplegia, uma lesão na medula espinhal a nível cervical que causa a perda dos

movimentos do tronco, pernas e braços, permitindo assim apenas o movimento dos olhos, o que passa a ser a única forma de comunicação.

Tal comunicação, que já existe no mercado hoje em dia em forma de ferramentas tecnológicas, nas quais podemos citar: *Telepatix*, *WeCanSpeak* e *TalkActive*, alguns em formato web ou versões somente para tecnologias iOS e alguns até com custos para aquisição.

Nesta direção, esta tecnologia oferecerá, a possibilidade de comunicação acessível e vale destacar a “fala” pronunciada a cada palavra formada que será de extrema utilidade pois nem sempre o usuário estará acompanhado, e assim, o mesmo poderá chamar pessoas ao seu redor.

Com este estudo, espera-se que traga ao usuário a capacidade de socializar, pois sem comunicação, as necessidades diárias e até mesmo a tomada de decisão é totalmente afetada e isto pode levá-lo a não se desenvolver no mundo social e a não ter uma boa saúde mental.

Apresentação do TCC									x
Finalização do projeto e ajustes finais									x

4. FUNDAMENTAÇÃO TEÓRICA

4.1 Tetraplegia e suas limitações

Uma das limitações físicas mais severas, a Tetraplegia Completa, tem um número de pessoas atingidas muito maior do que se imagina, pois normalmente esses indivíduos são frequentemente mantidos em suas residências, absolutamente isolados do mundo.

O diagnóstico da Tetraplegia Completa pode ocorrer de origem traumática pois seus maiores causadores são os acidentes de automóvel, os mergulhos de cabeça e doenças que resultam nela, como Esclerose Lateral Amiotrófica, Acidente Vascular Cerebral / Encefálico dentre outros. Sendo assim, a vida de uma pessoa com Tetraplegia Completa é muito difícil, desde seu início pois uma pessoa que até então fazia tudo sozinha, agora dependera de ajuda constante de outras pessoas, e infelizmente quanto mais pobre, menos acesso tem a tecnologias assistivas, e respectivamente mais isolado estará.

Apesar dessa perspectiva tão negativa, hoje em dia é possível o acesso a algumas ferramentas tecnológicas que tem tornado viável uma vida muito menos difícil, tanto para a pessoa portadora de Tetraplegia Completa, quanto pela sua família e a sociedade na qual ele está inserido. Nestes casos, já estão disponíveis ferramentas pouco convencionais como o acionamento de dispositivos pelo reconhecimento eletrônico do movimento do olho, o que no caso do portador deste tipo de Tetraplegia é sua única forma de comunicação.

4.2 Tecnologia assistiva

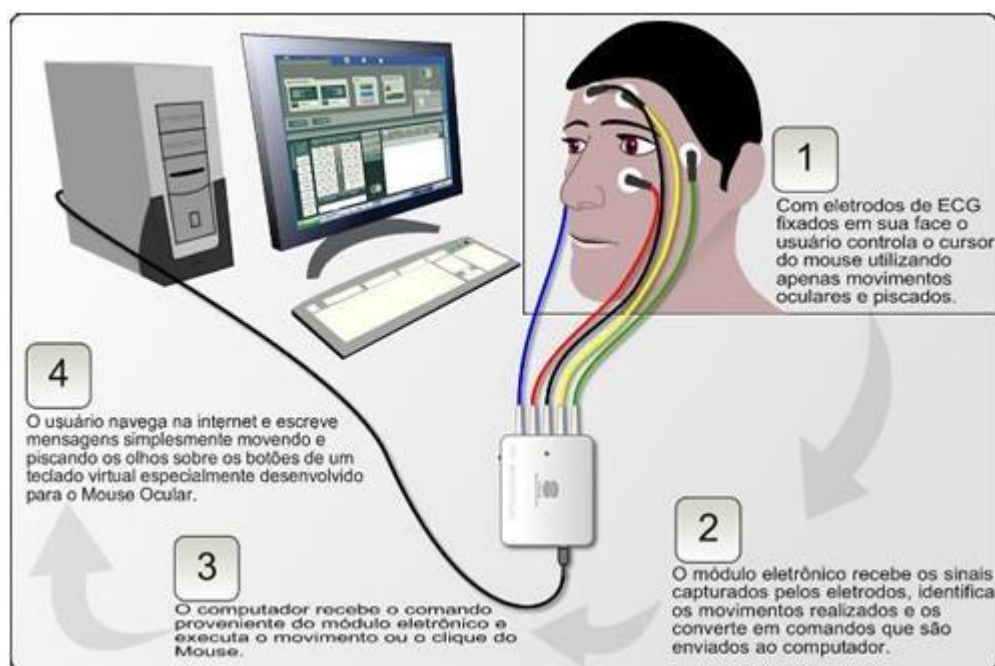
Tecnologia Assistiva é um termo utilizado para identificar recursos voltados às pessoas com deficiência, que visam proporcionar autonomia e em alguns casos independência.

“Para as pessoas sem deficiência a tecnologia torna as coisas mais fáceis. Para as pessoas com deficiência, a tecnologia torna as coisas possíveis”.
(RADABAUGH, 1993)

A evolução tecnológica caminha para tornar a vida de todos mais fáceis, em nosso dia a dia utilizamos uma interminável lista de recursos que facilitam nosso desempenho em funções pretendidas.

Para as pessoas com Tetraplegia Completa temos alguns exemplos de tecnologia assistiva, e talvez, o que mais tem se destacado, é o Mouse Ocular (Figura 2), que utiliza a movimentação dos músculos do globo ocular para controlar o cursor do mouse.

Figura 2 – Funcionamento Mouse Ocular 1



Fonte: <https://cwi.com.br/blog/tecnologias-assistivas-e-sua-importancia/> Acesso em 29/11/2023

Existe também o Tobii PCEye Plus (Figura 3), um sistema composto por um monitor com sensores, caixa de som, câmera, dentre outros que permite controlar qualquer computador através do movimento dos olhos, e também controlar um ambiente para automação residencial pois permite controlar equipamentos da casa que possuem controle remoto.

Figura 3 - Tobii PCEye Plus 1

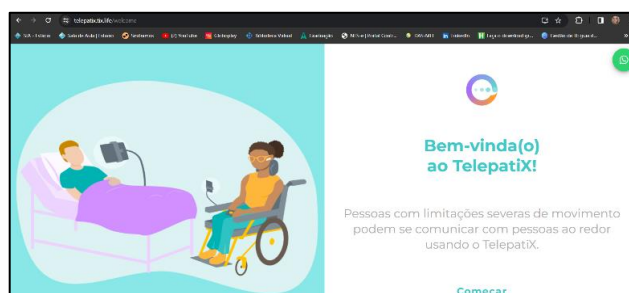


Fonte: <https://www.tobiibrasil.com/produto/tobiipceyepluseyer/> Acesso em 29/11/2023

No contexto de comunicação por meio da fala, há disponibilidade de teclados virtuais, no entanto, alguns destes podem ser inacessíveis devido a sua exclusividade para alguns sistemas, ou até mesmo de compreensão um pouco complexa.

Um exemplo de teclado virtual acessível para pessoas com Tetraplegia Completa, desenvolvido em web que é gratuita é o Telepatix (Figura 4), além de ocular, ele também pode ser usado com leves toques na tela, ou até mesmo com o clique do botão esquerdo de qualquer mouse comum.

Figura 4 – Site Teclado Virtual Telepati 1



Fonte: <https://telepatix.tix.life/welcome> Acesso em 05/10/2023

5. DESENVOLVIMENTO

O entendimento dos teclados virtuais e sua evolução desempenham um papel crucial como base teórica antes de avançarmos para as próximas fases de desenvolvimento. O programa proposto neste trabalho acadêmico, tem diversos

módulos que necessitam ser abrangidos para que se possa determinar às condições de software e desdobramentos necessários. Além disso, é importante compreender não apenas as funções incorporadas no teclado virtual, mas também a integração das informações provenientes do reconhecimento facial, do movimento dos olhos e do piscar, resultando em uma forma de comunicação eficaz.

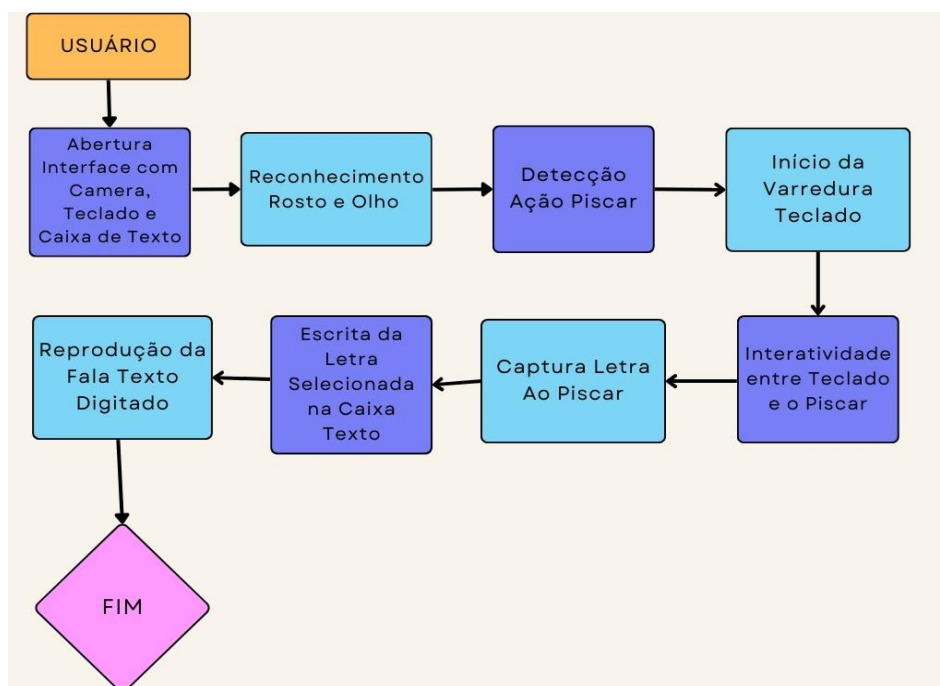
5.1 Arquitetura do sistema

O sistema desenvolvido requer a fixação dos olhos do usuário, e através do piscar será selecionado uma letra por vez, enquanto, o teclado vai passando letra a letra e escrevendo o texto conforme o piscar, segue no Fluxograma da Figura 5.

Requisitos do sistema proposto:

- Deverá reconhecer todas as letras do alfabeto através do piscar dos olhos;
- Deverá funcionar em tempo real. Ou seja, a detecção do piscar dos olhos, ocorrerá em uma fração de tempo gasto para digitar uma letra: imediatamente;
- Deverá ter a opção de apagar o que foi escrito em caso de erro ou desistência;
- Deverá ter a opção de salvar o texto que foi escrito, em um documento da Microsoft Word que ficará salvo na área de trabalho do computador em caso do usuário estar sozinho e precisar escrever algo para seu cuidador / acompanhante quando estiver disponível para ler;

Figura 5 - Fluxograma Arquitetura do Sis 1



Fonte: Elaborado pela Autora

5.2 Tecnologias

5.2.1 Python

Escolhemos Python pois se trata de uma linguagem de programação (linguagem de programação é basicamente um software, que o programador utiliza para escrever outros programas de computador, seja ele um aplicativo ou um programa básico) multiplataforma, o que significa que você pode executar seu código em diferentes sistemas operacionais sem grandes modificações.

Python também tem bibliotecas bem completas e amplas no assunto de reconhecimento facial e de objetos. Existem várias razões pelas quais Python é frequentemente escolhido para esse tipo de projeto:

A. Bibliotecas de Visão Computacional:

Python possui uma excelente biblioteca chamada OpenCV (*Open Source Computer Vision*), que fornece uma ampla gama de ferramentas e algoritmos para visão computacional. Isso inclui detecção facial, rastreamento de objetos, reconhecimento de padrões. Além do OpenCV, há outras bibliotecas úteis, como Dlib para detecção e reconhecimento facial.

B. Facilidade de Desenvolvimento:

Python é conhecido por sua sintaxe clara e legibilidade, o que torna o

desenvolvimento de código mais fácil e rápido. Isso é especialmente útil em projetos onde a prototipagem rápida é essencial.

C. Machine Learning e Aprendizado Profundo:

Para expandir o projeto para incluir recursos mais avançados, como reconhecimento de expressões faciais, *machine learning* ou aprendizado profundo, Python oferece uma ampla variedade de bibliotecas e frameworks para essas tarefas, como *TensorFlow*, *Keras* e *PyTorch*.

Em resumo, Python é uma escolha sólida para projetos de visão computacional, especialmente quando há uma necessidade de prototipagem rápida, uma variedade de bibliotecas disponíveis e uma comunidade ativa que pode oferecer suporte e recursos adicionais. O Python é uma linguagem com alta legibilidade, facilidade de escrita.

5.2.1.1 Bibliotecas do Python

As bibliotecas utilizadas para o desenvolvimento do software foram:

- A. **OS:** Essa biblioteca foi para interagir com o sistema operacional, ela manipula caminhos de arquivos, cria diretórios e garante que os documentos sejam salvos em um local específico.
- B. **Docx:** Foi usada para criar e salvar o texto inserido pelo usuário em um documento Microsoft Word.
- C. **Open CV2:** Essa biblioteca é amplamente utilizada no código para processamento de imagem e visão computacional. Está empregada para capturar frames da câmera, redimensionar a imagem, converter para tons de cinza e desenhar linhas durante a detecção facial.
- D. **Numpy:** O NumPy foi uma biblioteca fundamental na parte de computação numérica, foi utilizada para operações matriciais, essenciais em processamento de imagem, como na criação da matriz representando o teclado.
- E. **From math / hypot:** Essa importação específica da função “*hypot*” do módulo “*math*” foi usada para calcular a hipotenusa, que é aplicada no cálculo do comprimento de linhas no processo de detecção do piscar dos olhos.
- F. **Pyglet:** Esta biblioteca foi usada para reprodução de som no projeto. No código, ela é responsável por carregar e tocar o som associado ao pressionamento das teclas.

G. Pyttsx3: esta foi usada para converter texto em fala. utilizada para sintetizar e reproduzir em áudio o texto digitado pelo usuário.

H. Dlib: Essa biblioteca é conhecida por suas ferramentas de aprendizado de máquina aplicadas à visão computacional. Neste código, foi usada para detecção facial e localização de pontos específicos no rosto, como os olhos.

Essas bibliotecas, em conjunto, oferecem um conjunto diversificado de funcionalidades, permitindo a criação de um teclado virtual que incorpora detecção facial, reconhecimento de olhos, reconhecimento de piscar de olhos, interação de teclas, conversão de texto em fala e manipulação de documentos do Microsoft Word.

5.3 Abertura da câmera e a transmissão da imagem em tempo real

A inicialização da câmera é o primeiro passo e para que consigamos um bom resultado na transmissão ao vivo das imagens, o faremos por captura de frames. Um "frame" em processamento de imagem refere-se a uma única imagem estática que compõe uma sequência de imagens em movimento. No contexto de captura de vídeo, como a transmissão da webcam em tempo real, cada frame representa um instantâneo da cena em um determinado momento.

Após a leitura de frames, à cada interação do *loop* principal, um novo frame é lido da câmera e o frame capturado é redimensionado. Isso é feito para reduzir o tamanho da imagem, o que pode ser útil para melhorar o desempenho.

5.3.1 Conversão das imagens para tons de cinza

A imagem colorida é convertida para tons de cinza, isso é muito benéfico para processamento de imagem, especialmente esta na detecção facial pois ela torna o formato da imagem mais certa na detecção. Operações em imagens em tons de cinza são computacionalmente mais eficientes em comparação com imagens coloridas. A manipulação de uma única intensidade de pixel (escala de cinza) é menos intensiva em recursos computacionais do que lidar com três valores de canal de cor.

Ao converter para tons de cinza, a informação de intensidade luminosa torna-se mais proeminente. Essa intensidade representa a quantidade de luz em cada pixel, permitindo um foco mais direto em características-chave, como bordas e contrastes

na imagem.

5.4 Detecção dos olhos e o piscar

Pessoas com Tetraplegia Completa podem utilizar sistemas que seguem o movimento ocular, permitindo que controlem computadores, dispositivos móveis e até mesmo interfaces de comunicação especializadas apenas com o movimento dos olhos. O rastreamento ocular é frequentemente alcançado por meio de tecnologias como rastreadores oculares, que utilizam câmeras para monitorar a direção do olhar e traduzi-la em comandos compreensíveis.

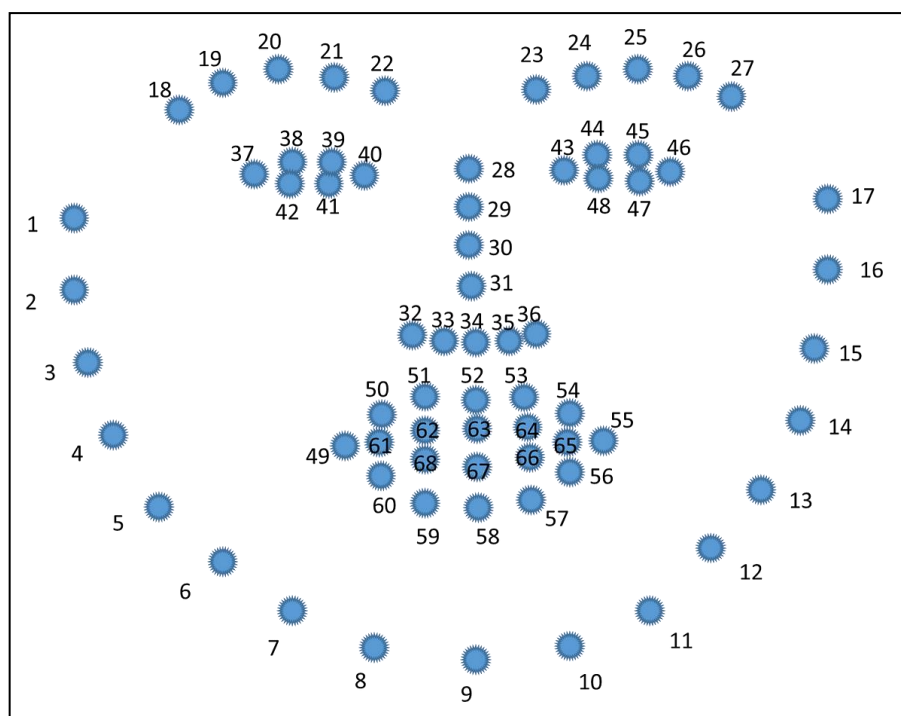
A capacidade de detectar rostos, olhos e até mesmo captar movimentos oculares, incluindo o piscar, é crucial para o desenvolvimento de tecnologias que visam melhorar a vida de pessoas com deficiências físicas, como a tetraplegia.

5.4.1 USO DO ARQUIVO "*SHAPE_PREDICTOR*" NO PROJETO

No desenvolvimento deste projeto as informações contidas no arquivo "*shape_predictor_68_face_landmarks_GTX.dat*" (Figura 6) desempenha um papel vital. Esse arquivo contém pontos de referência específicos no rosto humano, essenciais para o rastreamento facial e a detecção de movimentos, incluindo os movimentos oculares.

Ao utilizar este modelo pré-treinado, nos beneficiamos do conhecimento prévio sobre a estrutura facial humana. Isso simplifica significativamente o processo de identificação de regiões importantes, como os olhos, e a coleta de informações necessárias para interpretar o movimento ocular. Em suma, o uso desse arquivo no projeto permitiu uma detecção mais eficiente de pontos-chave no rosto e contribuiu para uma experiência mais precisa e responsiva no rastreamento do movimento ocular, o que é crucial para a interação do usuário no seu teclado virtual.

Figura 6 - Modelo de shape predictor 1

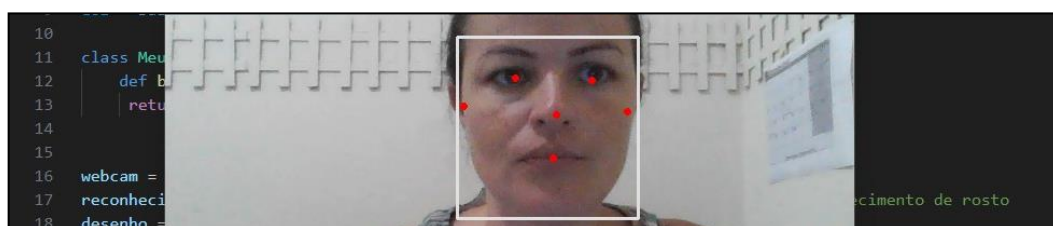


Fonte: <https://vigneshs4499.medium.com/face-detection-and-landmarks-using-dlib-and-opencv-8c824f50cc78> Acesso em 05/09/2023

5.4.2 O cálculo do piscar

O arquivo *shape predictor* contém informações sobre os pontos faciais (Figura 7) em uma imagem, para realizarmos a soma do piscar utilizaremos os pontos de referência dos olhos somente.

Figura 7 – Pontos Faciais do Modelo 1



Fonte: Elaborado pela Autora

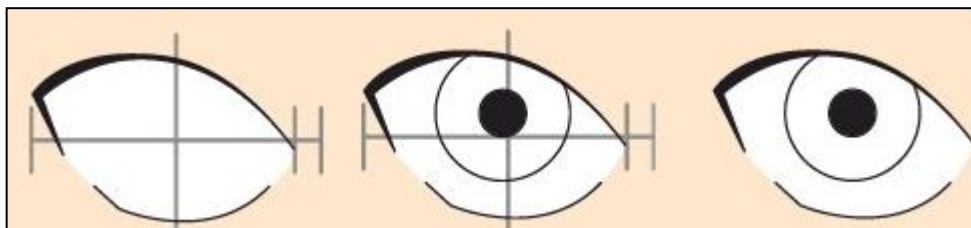
Cada ponto facial tem uma coordenada (x, y) que representa uma característica específica, como o canto do olho, a ponta do nariz, etc.

Os pontos faciais relacionados aos olhos são usados para determinar o piscar. Esses pontos geralmente incluem cantos internos e externos dos olhos, bem como

pontos ao longo da parte superior e inferior das pálpebras.

A métrica de piscada (Figura 8) é calculada medindo as distâncias entre pontos específicos dos olhos. Por exemplo, a distância entre os cantos esquerdo e direito dos olhos pode ser usada como a largura dos olhos, e a distância entre os pontos médios superior e inferior pode ser usada como a altura dos olhos.

Figura 8 - Linhas para cálculo piscar 1



Fonte: <https://www.pressreader.com/brazil/guia-curso-de-desenho/20210531/283635352926947>

Acesso em 10/09/2023

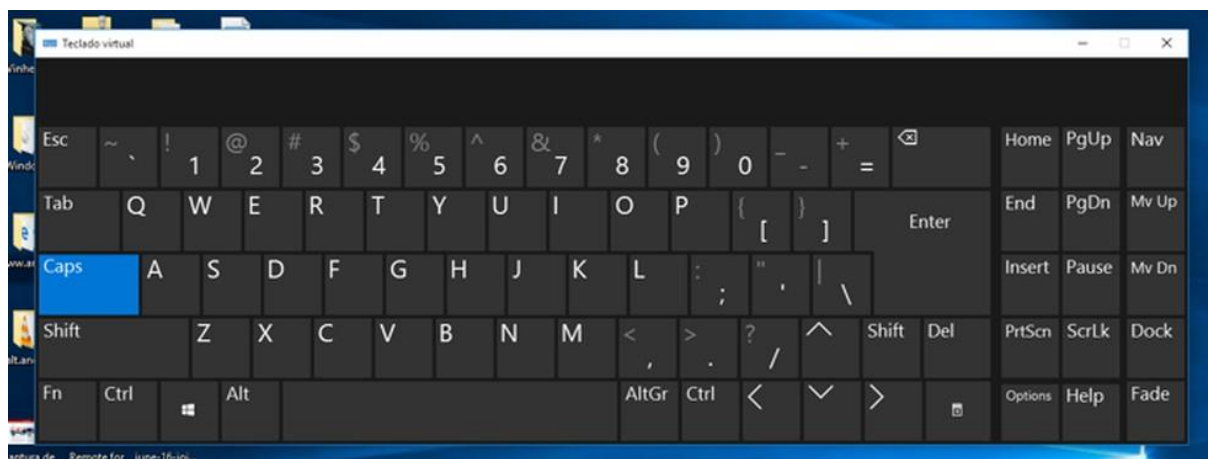
A razão entre a largura e a altura dos olhos é então calculada. Em condições normais (olhos abertos), essa razão é relativamente constante.

Quando ocorre um piscar, a razão largura/altura se altera significativamente. Isso é detectado ao comparar a razão calculada com um limiar. Se a razão ultrapassar esse limiar, é considerado um piscar.

5.5 Posicionamento das teclas e lógica na seleção letras para construção do teclado virtual

A maioria dos teclados virtuais, usam um layout padrão (Figura 9), e ao analisa-lo, observa-se a presença de um grande número de teclas, que, embora projetada para acomodar uma variedade de caracteres e funções, pode se revelar potencialmente exaustiva durante a escrita e isto foi o que motivou a exploração de alternativas mais ergonômicas e intuitivas para otimizar o processo de escrita no contexto deste projeto.

Figura 9 – Layout Teclado Virtual Padrão 1



Fonte: <https://www.techtudo.com.br/noticias/2016/06/como-ativar-o-teclado-virtual-no-windows-10.ghml> Acesso em 10/09/2023

Considerando esse aspecto, e após uma análise mais aprofundada sobre layouts simplificados, deparou-se com uma investigação conduzida pela UFRJ – Universidade Federal do Rio de Janeiro sobre a frequência de uso das letras em palavras faladas na língua portuguesa, observou-se uma ordem específica em que determinadas letras são mais comumente empregadas em contextos cotidianos. Os resultados dessa pesquisa foram compilados em uma tabela (Figura 10) que detalha, em porcentagem, a ordem de frequência das letras no contexto em questão.

Figura 10 - Tabela de Resultados 1

Letra	Frequência	Letra	Frequência
A	14.63%	N	5.05%
B	1.04%	O	10.73%
C	3.88%	P	2.52%
D	4.99%	Q	1.20%
E	12.57%	R	6.53%
F	1.02%	S	7.81%
G	1.30%	T	4.34%
H	1.28%	U	4.63%
I	6.18%	V	1.67%
J	0.40%	W	0.01%
K	0.02%	X	0.21%

L	2.78%	Y	0.01%
M	4.74%	Z	0.47%

Fonte: https://www.gta.ufrj.br/grad/06_2/alexandre/criptoanalise.html#:~:text=As%20vogais%20A%2C%20E%2C%20I,10%20letras%20%C3%A9%20de%204.88 Acesso em 10/09/2023

Os resultados desta pesquisa são as seguintes ordens de letras mais utilizadas:

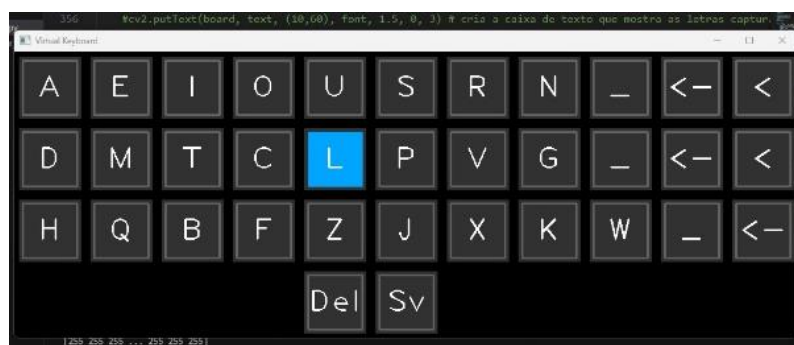
- 6 vogais: A, E, I, O, U, (Y) - 48.75 %
- 5 consoantes de frequência alta: S, R, N, D, M - 29.12 %
- 10 consoantes de frequência média: T, C, L, P, V, G, H, Q, B, F - 21.03 %
- 6 consoantes de frequência baixa: Z, J, X, K, W - 1.10%

Após análise destes resultados, foi utilizado o posicionamento das teclas no teclado conforme ordem acima citada no estudo, para que isso facilitasse e tornasse mais rápido o processo de escrita.

Sequência por linha utilizada no teclado virtual: *a, e, i, o, u, s, r, n, _ (espaço), <- (backspace), < (voltar), d, m, t, c, l, p, v, g, _ (espaço), <- (backspace), < (voltar), h, q, b, f, z, j, x, k, w, _ (espaço), <- (backspace), “del” (deletar) e “sv” (salvar).*

Como na figura abaixo:

Figura 11 – Sequência Letras Teclado Vir 1



Fonte: Elaborado pela Autora

5.6 UNINDO O PISCAR DOS OLHOS COM O TECLADO

Para interagir com o teclado virtual, é implementado um sistema de varredura que refere-se ao processo de percorrer e destacar uma a uma as letras disponíveis. Esse método de varredura é frequentemente empregado para facilitar a seleção de letras, possibilitando que o usuário, escolha letras específicas. A seleção de letras

ocorre sequencialmente durante a varredura, e cada piscada é associada à seleção da letra em foco naquele momento.

A varredura é uma abordagem eficiente, pois permite a seleção de letras de maneira sequencial, evitando a necessidade de rastreamento constante do olhar. A varredura é ativada por um determinado período, durante o qual a piscada é detectada para a seleção da letra em destaque.

Cada vez que uma piscada é captada durante a varredura, a letra em destaque é considerada selecionada. Isso é possível porque a varredura sequencial está vinculada à piscada detectada, indicando a letra naquele momento específico.

A seleção da letra envolve a adição da mesma ao texto em construção. Além disso, são tratadas ações especiais, como *backspace*, delete e salvar. A lógica de controle garante que essas ações sejam realizadas conforme a interação do usuário.

Para melhorar a experiência do usuário, foram implementados feedbacks visuais e sonoros no teclado virtual. Durante o processo de varredura, a letra em destaque indica qual será selecionada na próxima etapa, proporcionando uma orientação visual clara. Simultaneamente, um feedback sonoro é acionado para confirmar cada seleção. Vale ressaltar que, além do sinal sonoro que indica a letra, foi incorporada uma voz mecânica que narra todo o texto digitado. Essa funcionalidade foi concebida considerando situações em que o usuário, por exemplo, um tetraplégico, esteja em um ambiente diferente do seu auxiliar, possibilitando que este último ouça o conteúdo sendo digitado e ofereça assistência à distância, como chamar a pessoa.

Um controle preciso do fluxo de interação é mantido para garantir que as ações sejam tomadas apenas quando uma piscada é detectada durante a varredura. Mecanismos como temporizadores são empregados para sincronizar a varredura com a detecção de piscadas.

Essa abordagem é especialmente benéfica para pessoas com restrições de movimento, oferecendo uma forma alternativa e acessível de comunicação. O uso de um teclado virtual por meio de piscadas amplia as possibilidades de interação para aqueles com limitações motoras.

5.7 A caixa de texto e demais funções

Uma caixa de texto é incorporada à interface para exibir as letras selecionadas pelo piscar do olho. Essa caixa atua como um espaço onde as letras são organizadas para formar palavras e, subsequentemente, frases. A atualização contínua da caixa de texto reflete as escolhas do usuário à medida que ele pisca os olhos para selecionar letras.

Conforme o usuário interage com o teclado virtual, as letras selecionadas são adicionadas à caixa de texto. A organização das letras na caixa ocorre de acordo com a sequência em que foram selecionadas. Isso permite que o usuário veja as palavras e frases que está construindo ao longo do tempo, facilitando a comunicação.

A inclusão do botão "Salvar" (SV) é uma adição valiosa à interface. Esse botão permite que o usuário salve o texto na caixa de texto como um documento Microsoft Word (.docx). Essa funcionalidade é especialmente útil quando o usuário precisa armazenar as mensagens para referência futura.

A capacidade de salvar as mensagens escritas oferece uma solução prática para situações em que a pessoa deseja revisitar ou compartilhar suas comunicações. Ao transformar o texto em um documento Word, é criada uma forma duradoura e acessível de armazenar as interações do usuário.

A funcionalidade de salvar não apenas oferece praticidade, mas também promove a inclusão digital, permitindo que a pessoa acesse suas mensagens de forma independente. Isso torna o sistema mais flexível, adaptando-se às necessidades e circunstâncias específicas do usuário.

A combinação da caixa de texto visível, formação de palavras e frases, juntamente com a capacidade de salvar, contribui significativamente para facilitar a comunicação do usuário por meio do teclado virtual. O sistema é projetado para oferecer uma experiência completa e eficaz de expressão por meio do movimento ocular.

5.8 Botões especiais

5.8.1 Botão backspace (representado por <-)

O botão Backspace é uma adição fundamental para corrigir erros durante a escrita. Representado pelo símbolo <-, esse botão apaga a última letra inserida na

caixa de texto. Essa funcionalidade é útil quando o usuário seleciona uma letra por engano ou deseja corrigir parte do texto. A inclusão do botão Backspace oferece ao usuário um meio eficiente de edição e correção, contribuindo para uma experiência de escrita mais precisa.

5.8.2 Botão voltar (representado pelo símbolo <)

O botão Voltar, simbolizado pelo caractere <, é projetado para oferecer conveniência ao usuário. Ao ser posicionado ao final de cada linha, permite que o usuário retorne rapidamente ao início da primeira linha, evitando a espera pela varredura até a última linha. Essa facilidade é especialmente útil para agilizar a escrita, pois o usuário pode começar a formar palavras e frases sem a necessidade de esperar pela próxima varredura.

5.8.3 Botão espaço (representado pelo _)

O botão Espaço, representado pelo caractere _, tem o propósito de facilitar a inserção de espaços entre as palavras. Colocado ao final de cada linha, possibilita que o usuário adicione espaços sem esperar que a varredura alcance a última linha do teclado virtual. Essa decisão de design visa otimizar o processo de escrita, permitindo que o usuário crie mensagens de forma fluida e eficiente.

5.8.4 Botão del (localizado na última linha)

O botão Del, localizado ao final da última linha, desempenha um papel importante na experiência do usuário. Ao ser acionado, esse botão apaga todo o conteúdo presente na caixa de texto. Essa funcionalidade é estrategicamente incorporada para oferecer ao usuário a capacidade de encerrar uma conversa ou limpar a caixa de texto quando desejar começar uma nova interação. Isso proporciona flexibilidade e controle ao usuário sobre o que está sendo comunicado, adaptando-se às diversas situações de uso.

5.8.5 Intenção por trás dos botões especiais

A inclusão desses botões especiais visa otimizar a experiência de escrita do usuário, considerando aspectos como correção rápida, agilidade na inserção de espaços e facilidade de navegação entre linhas. Esses recursos são cuidadosamente

projetados para tornar a comunicação mais eficaz, atendendo às necessidades práticas e permitindo uma interação mais suave e intuitiva com o teclado virtual.

Em resumo, os “*botões backspace, voltar, espaço e del*” são elementos estratégicos que aprimoram a usabilidade do teclado virtual, oferecendo soluções práticas para correção, agilidade e controle durante o processo de escrita. Essas características são integradas para proporcionar uma experiência de comunicação mais eficiente e adaptável.

6. TESTES

Este capítulo abordará detalhadamente o processo de teste do sistema de teclado virtual desenvolvido para facilitar a comunicação por meio de movimentos oculares. Cada etapa será minuciosamente explicada, proporcionando uma compreensão aprofundada do funcionamento do sistema e dos testes realizados.

A. TESTE 1:

Captura de vídeo em tempo real:

A captura da câmera em tempo real desempenha um papel fundamental no sucesso deste projeto de teclado virtual baseado no piscar dos olhos. Essencialmente, a câmera atua como os "olhos" do sistema, permitindo a detecção e interpretação dos movimentos oculares do usuário.

A câmera em tempo real possibilita a detecção precisa de rostos na cena. Isso é fundamental para estabelecer uma referência inicial para a localização dos *landmarks* faciais, pontos essenciais para calcular a taxa de piscadas e determinar a área de foco dos olhos do usuário. Ao capturar a imagem em tempo real, é possível identificar landmarks faciais, como os contornos dos olhos. Estes landmarks são cruciais para calcular a taxa de piscadas e, conseqüentemente, interpretar os movimentos oculares do usuário, sendo a base para a interação com o teclado virtual.

A captura contínua da câmera permite uma interação dinâmica e em tempo real. Isso significa que o sistema pode ajustar-se instantaneamente aos movimentos oculares do usuário, garantindo uma experiência de comunicação mais fluida e eficaz. Em um ambiente dinâmico, a captura em tempo real possibilita a adaptação do sistema a diferentes condições de iluminação, variações de fundo e posições do usuário. Essa capacidade de ajuste é crucial para a robustez e confiabilidade do teclado virtual em situações do mundo real.

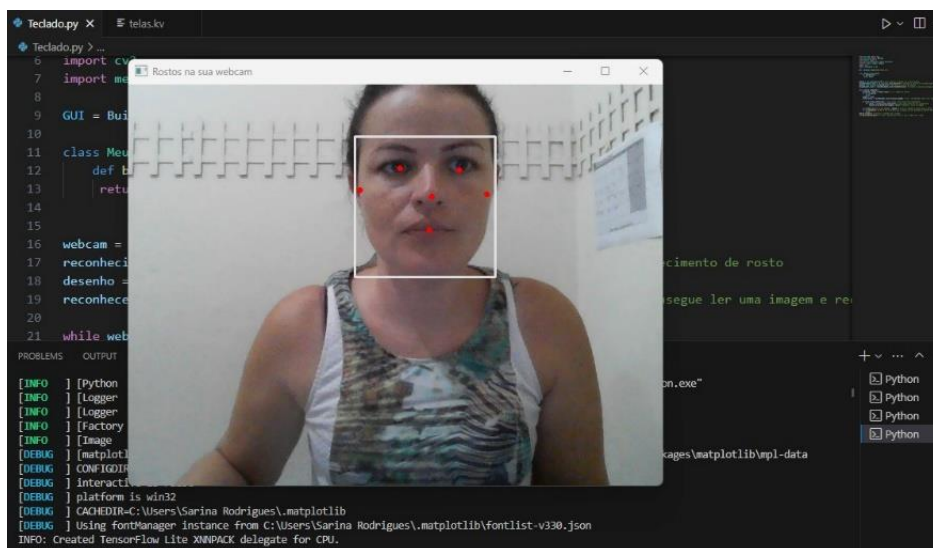
A visualização em tempo real da área de foco dos olhos do usuário contribui diretamente para a usabilidade do sistema. Essa abordagem fornece feedback imediato, permitindo ao usuário ajustar sua interação conforme necessário.

Testes realizados com sucesso (Figura 12):

- Verificação da correta abertura da câmera;
- Renderização adequada da interface gráfica;
- Garantia de que os frames de vídeo são processados sem problemas;

- Confirmação da detecção precisa de rostos na cena;
- Verificação da correta localização dos landmarks no rosto do usuário.

Figura 12 – Teste Captura Video em Tempo 1



Fonte: Elaborado pela Autora

B. TESTE 2:

Detecção de piscadas:

A detecção de piscadas, um componente vital neste projeto, desempenha um papel crucial na interpretação dos movimentos oculares do usuário.

A detecção de piscadas serve como um comando de interação fundamental. Quando o usuário pisca, esse gesto é interpretado pelo sistema como uma ação específica, como a seleção de uma tecla do teclado virtual. Esse método de interação é intuitivo e permite que o usuário comunique suas escolhas de maneira eficiente.

A detecção de piscadas oferece uma abordagem precisa e específica para capturar a intenção do usuário. A detecção de piscadas ajuda a reduzir falsos positivos, garantindo que apenas os gestos intencionais sejam considerados.

A taxa de piscadas é calculada com base na análise dos movimentos dos olhos do usuário. Essa métrica é crucial para determinar a intenção por trás de cada piscada. Uma taxa específica pode ser associada a ações diferentes, proporcionando uma variedade de comandos com base na frequência e duração das piscadas.

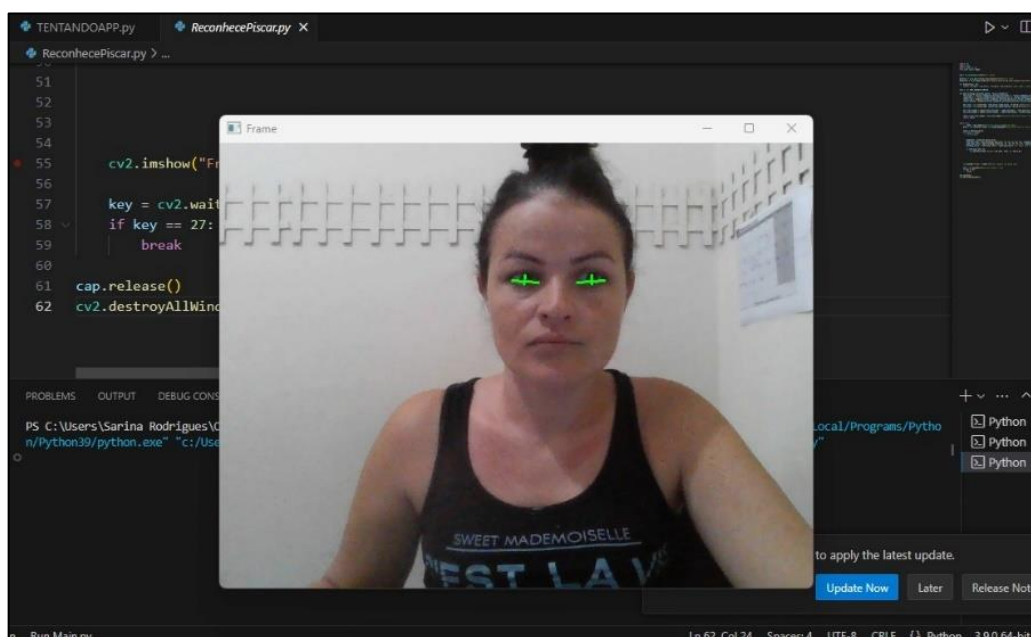
A flexibilidade da detecção de piscadas permite adaptações conforme as necessidades do usuário. Diferentes tempos de piscadas foram colocados para que a

pessoa possa piscar normalmente e caso ela queira seleccionar uma tecla o piscar deve ser um pouco mais demorado para não ter falsas seleções das letras

Testes realizados com sucesso (Figura 13 e 14):

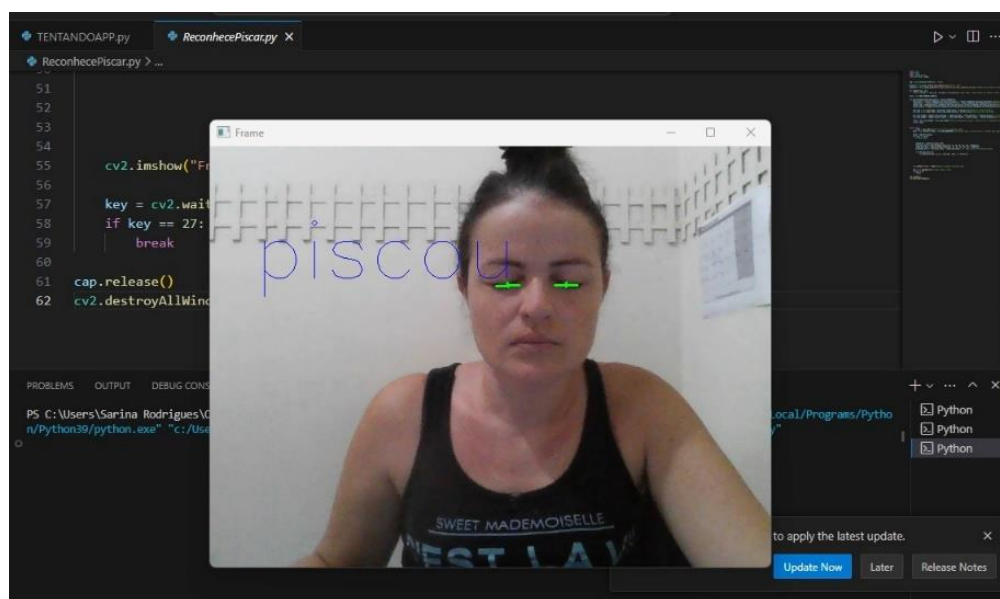
- Avaliação da precisão na detecção de piscadas;
- Alteração no tempo do piscar para seleccionar letras, piscadas normais não seleccionam somente as mais demoradas;
- Verificação da correta associação de movimentos oculares às teclas do teclado virtual;
- Teste das ações especiais, como *backspace*, delete e salvar.

Figura 13 – Desenho Linhas na Soma Pisca 1



Fonte: Elaborado pela Autora

Figura 14 – Detectando Piscar 1



Fonte: Elaborado pela Autora

C. TESTE 3:

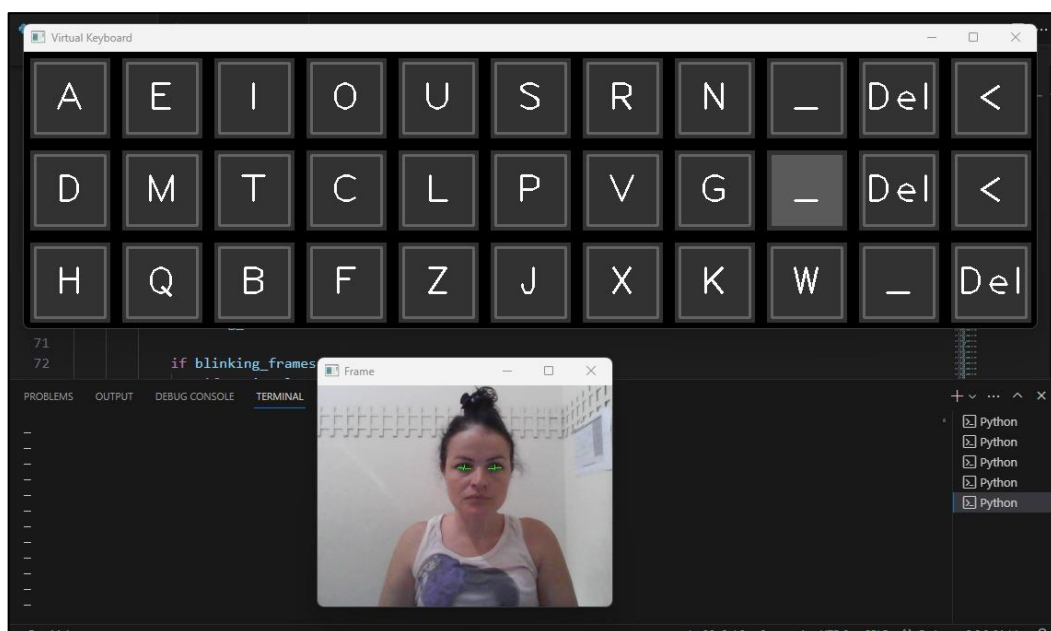
Funcionalidades e integração do teclado virtual

Teste de varredura e funcionamento inicial (Figura 15):

Com a integração da detecção de piscadas, foi essencial realizar testes para garantir que a varredura pelo teclado virtual funcionasse adequadamente. Esse processo envolveu a verificação da precisão da varredura entre as letras, garantindo que o sistema respondesse eficientemente aos movimentos oculares do usuário.

Assegurar que cada letra fosse destacada corretamente durante a varredura foi um passo crucial para estabelecer a base do teclado virtual.

Figura 15 – Teste Varredura 1

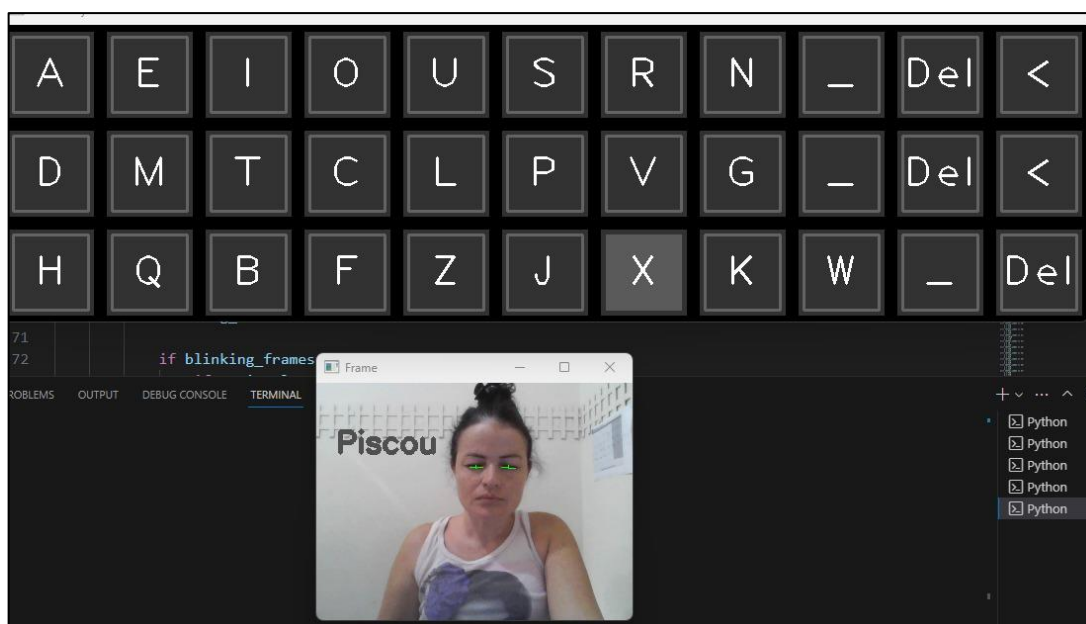


Fonte: Elaborado pela Autora

Integração da Detecção de Piscadas (Figura 16):

Após o teste bem-sucedido da varredura, a integração da detecção de piscadas foi implementada para adicionar uma camada de interação mais intuitiva. A detecção de piscadas foi calibrada para interpretar os gestos do usuário como comandos específicos, como a seleção de uma tecla ou a execução de funções especiais. Essa integração trouxe uma abordagem mais natural e eficaz para a interação com o teclado virtual.

Figura 16 – Integração piscar 1



Fonte: Elaborado pela Autora

Teste abrangente com todas as funcionalidades:

O teste abrangente envolveu a combinação de todas as funcionalidades do teclado virtual, desde a varredura até a detecção de piscadas e a resposta correspondente a cada comando. Durante esse teste, a precisão, a rapidez e a adaptabilidade do sistema foram avaliadas. O usuário teve a oportunidade de formar palavras e frases completas usando apenas os movimentos oculares, enquanto o sistema processava essas entradas em tempo real.

Adaptação contínua e aprimoramentos:

Com base nos resultados do teste, o sistema foi ajustado para otimizar ainda mais a detecção de piscadas, a precisão da varredura e a resposta geral. A capacidade de adaptação contínua do teclado virtual permitiu ajustes personalizados, atendendo às necessidades específicas de cada usuário. Essa flexibilidade garantiu que o teclado virtual pudesse se adequar a diferentes estilos e ritmos de interação.

Testes realizados com sucesso:

- Garantia de que as teclas são selecionadas corretamente com base nos movimentos oculares.
- Verificação das ações associadas a cada tecla.

6.1 Conclusão dos testes

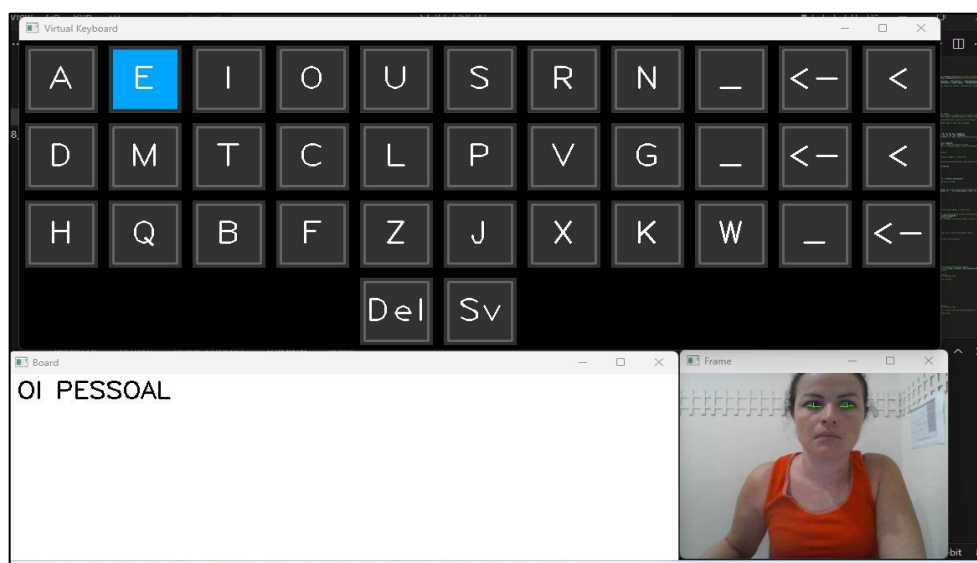
Os testes realizados garantiram a precisão na detecção de piscadas, associando corretamente os movimentos oculares, neste caso se tratando do piscar dos olhos, às teclas do teclado virtual. A detecção de piscadas contribui para uma interação intuitiva e eficiente, reduzindo falsos positivos e permitindo uma comunicação precisa.

Na varredura do teclado virtual, garantiu a precisão na seleção de letras durante os movimentos oculares. A integração da detecção de piscadas adicionou uma camada intuitiva, interpretando gestos como comandos específicos, enriquecendo a experiência de usuário.

O teste abrangente envolveu a combinação de todas as funcionalidades, desde a varredura até a detecção de piscadas. Os resultados (Figura 17) positivos demonstraram a adaptabilidade e eficácia do teclado virtual. Adaptações contínuas foram feitas para otimizar a detecção de piscadas e a precisão da varredura, visando proporcionar uma interação personalizada e eficiente para cada usuário.

O projeto ao alcançar sucesso nos testes, representa não apenas uma alternativa inovadora de comunicação, mas também destaca a importância da captura em tempo real e da detecção de piscadas em soluções de interação baseadas em visão computacional. O constante aprimoramento visa garantir uma experiência acessível e eficaz para todos os usuários.

Figura 17 – Teste Final 1



Fonte: Elaborado pela Autora

7. RESULTADOS

Ao longo do desenvolvimento e testes deste projeto de teclado virtual baseado no piscar dos olhos, obtivemos resultados significativamente positivos que refletem não apenas o sucesso da implementação, mas também a evolução contínua do sistema para atender às necessidades específicas dos usuários.

Inicialmente, enfrentamos o desafio inerente de sincronizar o ato de piscar com a captura exata da letra desejada. A delicadeza desse equilíbrio inicialmente apresentou uma curva de aprendizado, conforme os usuários se adaptavam à dinâmica do sistema. No entanto, com o decorrer dos testes e a constante otimização, observamos uma notável melhoria na facilidade com que os usuários conseguiram sincronizar seus movimentos oculares com a captura das letras.

Um dos pontos cruciais do ajuste contínuo foi o tempo de piscar necessário para efetuar a seleção da letra desejada. Inicialmente, esse intervalo pode ter parecido desafiador, mas, com base nas respostas dos usuários e na análise dos resultados, conseguimos adaptar esse tempo para garantir uma experiência mais intuitiva e amigável. A flexibilidade no ajuste desse parâmetro foi fundamental para a personalização do sistema, levando em consideração a diversidade de estilos e ritmos de interação.

Outro aspecto fundamental que demandou refinamento foi o tempo de varredura entre as letras no teclado virtual. Este foi cuidadosamente adaptado para evitar a lentidão excessiva, que poderia tornar a interação cansativa, e, ao mesmo tempo, impedir que fosse rápido demais, dificultando a precisão na seleção das letras. O objetivo era encontrar o equilíbrio ideal que proporcionasse uma experiência eficiente e agradável para o usuário.

Os resultados finais foram verdadeiramente satisfatórios, validando a eficácia e a utilidade prática do teclado virtual. Observamos um aumento significativo na facilidade de uso à medida que eu testava se acostumavam à interface, resultando em interações mais precisas e fluidas.

8. LIMITAÇÕES E TRABALHOS FUTUROS

A principal limitação constatada neste projeto foi na complexidade de recrutar voluntários que apresentam a condição de Tetraplegia Completa, para a realização de testes mais aprofundados.

Como continuidade desse trabalho, listaremos algumas sugestões de melhorias e novas funcionalidades que podem ser criadas:

- Um teclado virtual altamente personalizáveis que possam ser adaptados a diferentes níveis de habilidades motoras e preferências do usuário;
- Implementação para versão Web e aplicativo Mobile;
- Integração do teclado com o WhatsApp, através de um botão que faça o envio do texto via WhatsApp de terceiro;
- Implementar algoritmos de aprendizado de máquina para prever as intenções do usuário e melhorar a precisão na seleção de palavras, reduzindo a necessidade de entrada manual;
- Versão web e/ou app.

Por fim, não foi possível à realização do teste com uma pessoa portadora de Tetraplegia Completa e também com um profissional da área , este, seria muito interessante e importante para a validação e considerações finais.

9. CONCLUSÃO

O desenvolvimento do teclado virtual, acionado pelo piscar dos olhos, representa um avanço notável na convergência entre tecnologia e acessibilidade, especialmente para aqueles que dependem exclusivamente do movimento ocular para se comunicar, como os indivíduos com Tetraplegia do tipo Completa.

O resultado obtido não apenas atingiu as expectativas, demonstrando não apenas a funcionalidade esperada, mas também a incorporação bem-sucedida de funções adicionais que se destacaram como diferenciais notáveis. A adição dessas características extras enriqueceu consideravelmente a utilidade e versatilidade do teclado virtual.

É crucial ressaltar que o resultado, representa um avanço significativo no campo da tecnologia assistiva no qual pode proporcionar uma ferramenta eficiente e adaptativa, transcendendo as limitações físicas e facilitando a expressão e comunicação de maneira intuitiva.

Adicionalmente, este projeto marca uma conquista notável na interseção entre tecnologia e acessibilidade, promovendo uma mudança significativa na vida cotidiana das pessoas tetraplégicas. Além de demonstrar a viabilidade de um teclado virtual acionado pelo olhar, destaca-se pela incrível funcionalidade que impacta positivamente a vida daqueles que dependem dessas inovações.

Este projeto pode abrir portas para uma era mais inclusiva, onde a tecnologia se alinha perfeitamente com as necessidades essenciais da comunidade. Apesar de ser a primeira versão do software, as funcionalidades implementadas não apenas atendem, mas superam as expectativas, tornando-se uma fonte de esperança e conforto para aqueles que enfrentam desafios únicos em sua jornada de comunicação.

10. REFERÊNCIAS BIBLIOGRÁFICAS

CASEMIRO, R. CESAR. **Comunicação visual por computador na esclerose lateral amiotrófica.** Disponível:

<https://www.scielo.br/j/abo/a/VXXRDXR4zCS7STxJfZRXzXc/?lang=pt&format=pdf> .

Acesso em: 14. Mar. 2023.

CESUMAR, A.R. SILMARA. **A informática como ferramenta de auxílio na inclusão do portador de paralisia cerebral: um estudo de caso.** Disponível:

<https://periodicos.unicesumar.edu.br/index.php/iccesumar/article/view/48> . Acesso em:

14. Mar. 2023.

SILVA, AUGUSTO RAFAEL. **Desenvolvimento teclado virtual para comunicação por meio de gestos visuais.** Tese de doutorado. Faculdade de Engenharia elétrica da universidade Federal de Uberlandia. 2021.

SOUSA, NATASHA FERNANDES. **O respeito pela autonomia da pessoa com tetraplegia completa em reabilitação pelos profissionais de saúde.** Dissertação em Biomédica. 2017

GUANDELIN, B. RHAYON. **Um outro olhar sobre a paralisia Cerebral.** 1. ed. Belo Horizonte: Dialética, 2021.

GOOGLE, Inc. – **Mecanismo de buscas.** Disponível em: <https://www.google.com.br> . Acesso em: 20. Out. 2023.

PETRONI, N. NATALIA. **Introdução ao uso do tablet para comunicação alternativa por uma jovem com paralisia cerebral.** Disponível:

<https://www.scielo.br/j/rbee/a/zMFQvQLVBqc7vRkT9LXc6Gk/> . Acesso em: 20. Mar.

2023.

FERREIRA, F.S. ROSEMEIRE. **Jaque, aplicativo de comunicação via dispositivos móveis para crianças com paralisia cerebral.** Disponível:

https://drive.google.com/drive/folders/1jSu4QWBttR3aYeGRpoMqTAPhD2asJq_9 .

Acesso em: 20. Mar. 2023.

SAVIOLI, M. T. CAROLINA. **Diretrizes projetuais para interfaces de dispositivos touchscreen voltados a usuários com paralisia cerebral.** Disponível: <https://doi.org/10.23972/det2022iss25pp70-84> . Acesso em: 14. Mar. 2023.

SABINO, THAIS. **Tetraplégicos superam limites, viram artistas e até voltam a andar.** Disponível em: https://www.terra.com.br/vida-e-estilo/saude/doencas-e-tratamentos/tetraplegicos-superam-limites-viram-artistas-e-ate-voltam-a-andar-veja-casos,fd55db3f42124410VgnVCM4000009bcceb0aRCRD.html?utm_source=clipboard . Acesso em: 3. Dez. 2023.

NORONHA, HENRIQUE. **Tetraplegia: O que é, quais as causas, sintomas, tratamentos e muitos mais!.** Disponível em: <https://henriquenoronha.com.br/tetraplegia-o-que-e-quais-as-causas-sintomas-tratamentos-e-muito-mais/>> . Acesso em: 14.Mar. 2023.

SOUSA, ROGÉRIO. **Lesão Medular: O que são os níveis medulares?.** Disponível em: <https://www.youtube.com/watch?v=exemplo> . Acesso em: 18.Jul. 2023.

KOSMINSKY, ELLEN. **Trauma Raquimedular: Principais Causas, diagnósticos e Conduta.** Disponível em: <https://www.eumedicoresidente.com.br/post/trauma-raquimedular>> . Acesso em: 14.Mar. 2023

VARELLA, MARIA HELENA. **Traumas na Coluna – Entrevista.** Disponível em: <https://drauziovarella.uol.com.br/entrevistas-2/traumas-na-coluna-entrevista/> . Acesso em: 18.Jul. 2023

Qualidade de Vida. Biblioteca Virtual em Saúde - Ministério da Saúde, [s.d.]. Disponível em: https://bvsms.saude.gov.br/bvs/dicas/260_qualidade_de_vida.html . Acesso em: 12.Ago. 2023.

HIRATA, GISELLE. **Quais são as tecnologias para ajudar as pessoas com deficiências** Disponível em: <https://super.abril.com.br/mundo-estranho/quais->

[tecnologias-sao-desenvolvidas-para-ajudar-pessoas-com-deficiencia](#) . Acesso em: 18.Jul. 2023

PcEye Plus EyeR – Controle Ocular e Automação Residencial, [s.d.]. Disponível em: <https://www.tobiibrasil.com/produto/tobiipceyepluseyer//>> Acesso em: 29.Nov. 2023

VIANA, SUSANA. **Aaaaa menina! Identificando piscadas com visão computacional.** Disponível em: <https://suzana-svm.medium.com/acorda-menina-identificando-piscadas-com-vis-c3-a3o-computacional-dlib-ee7672da4d38/>> Acesso em: 16.Out. 2023

Rodas compiladas Dlib para Python 3.7,3.8,3.9 para Windows 10x64. Disponível em: <https://github.com/sachadee/Dlib> Acesso em: 16.Out. 2023

PySource **Gaze controlled keyboard with Python and Opencv.** Disponível em: <https://www.youtube.com/watch?v=xHxWsGq2V9c> Acesso em: 16.Out. 2023

Reconhecimentos de pontos de referencias faciais Disponível em: https://github.com/italojs/facial-landmarks-recognition/blob/master/shape_predictor_68_face_landmarks.dat Acesso em: 16.Out. 2023

Decifrando Textos em Portugues Disponível em: https://www.gta.ufrj.br/grad/06_2/alexandre/criptoanalise.html#:~:text=As%20vogais%20A%2C%20E%2C%20I,10%20letras%20%C3%A9%20de%204.88 Acesso em: 10.Set. 2023

VOLOSKI, JOANNA. **Tecnologias assistivas e suas importâncias** Disponível em: <https://cwi.com.br/blog/tecnologias-assistivas-e-sua-importancia/> Acesso em: 29.Nov. 2023

Bem vindo ao Telepatix!!! Disponível em: <https://telepatix.tix.life/welcome> Acesso em: 05.Out. 2023

SURESH, VIGNESH **Detecção de rosto e pontos de referencia usando Dlib e OpenCV** Disponível em: <https://vigneshs4499.medium.com/face-detection-and-landmarks-using-dlib-and-opencv-8c824f50cc78> Acesso em: 05.Set. 2023

11. ANEXO

```

import cv2
import dlib
import numpy as np
from math import hypot
import pygame
import time
import pytsx3
import docx
import os

#ler os sons
sound = pygame.media.load("sound.wav", streaming=False)

cap = cv2.VideoCapture(0)#abrir camera
board = np.zeros((300,800), np.uint8)#caixa de texto
board[:] = 255
max_characters_per_line = 27
last_action_del = False
backspace_pressed = False
letter_index = 0
doc = docx.Document()
file_name = "tecladovirtual.docx"
save_action_triggered = False

detector = dlib.get_frontal_face_detector()#detector rosto
predictor = dlib.shape_predictor("shape_predictor_68_face_landmarks_GTX.dat") #import doc
#externo com pontos de reconhecimento rosto

keyboard = np.zeros((400, 1100, 3),np.uint8) #tamanho teclado

```

```

keys_set_1 = {0: "A", 1:"E", 2:"I", 3:"O", 4:"U", 5:"S", 6:"R", 7:"N", 8:"_",
9:"U+232B", 10:"<",
               11:"D", 12: "M", 13:"T", 14:"C", 15:"L", 16:"P", 17:"V", 18:"G", 19:"_",
20:"U+232B", 21:"<",
               22:"H", 23:"Q", 24: "B", 25:"F", 26:"Z", 27:"J", 28:"X", 29:"K", 30:"W",31:
"_" , 32:"U+232B",
               33:"Del", 34:"Sv" }

```

```
def letter(letter_index, text, letter_light):
```

```

    if letter_index == 0:
        x = 10
        y = 10
    elif letter_index == 1:
        x = 110
        y = 10
    elif letter_index == 2:
        x = 210
        y = 10
    elif letter_index == 3:
        x = 310
        y = 10
    elif letter_index == 4:
        x = 410
        y = 10
    elif letter_index == 5:
        x = 510
        y = 10
    elif letter_index == 6:
        x = 610
        y = 10
    elif letter_index == 7:
        x = 710
        y = 10

```



```
elif letter_index == 8:
    x = 810
    y = 10
elif letter_index == 9:
    x = 910
    y = 10
elif letter_index == 10:
    x = 1010
    y = 10
elif letter_index == 11:
    x = 10
    y = 110
elif letter_index == 12:
    x = 110
    y = 110
elif letter_index == 13:
    x = 210
    y = 110
elif letter_index == 14:
    x = 310
    y = 110
elif letter_index == 15:
    x = 410
    y = 110
elif letter_index == 16:
    x = 510
    y = 110
elif letter_index == 17:
    x = 610
    y = 110
elif letter_index == 18:
    x = 710
    y = 110
elif letter_index == 19:
```

```
x = 810
y = 110
elif letter_index == 20:
    x = 910
    y = 110
elif letter_index == 21:
    x = 1010
    y = 110
elif letter_index == 22:
    x = 10
    y = 210
elif letter_index == 23:
    x = 110
    y = 210
elif letter_index == 24:
    x = 210
    y = 210
elif letter_index == 25:
    x = 310
    y = 210
elif letter_index == 26:
    x = 410
    y = 210
elif letter_index == 27:
    x = 510
    y = 210
elif letter_index == 28:
    x = 610
    y = 210
elif letter_index == 29:
    x = 710
    y = 210
elif letter_index == 30:
    x = 810
```

```

        y = 210
    elif letter_index == 31:
        x = 910
        y = 210
    elif letter_index == 32:
        x = 1010
        y = 210
    elif letter_index == 33:
        x = 410
        y = 310
    elif letter_index == 34:
        x = 510
        y = 310

width = 80
height = 80
th = 2 # thickness espessura
shadow_color = (50,50,50)#sombra teclas

        cv2.rectangle(keyboard, (x+th-5,y+th-5), (x+width-th+5,y+height-
th+5),shadow_color, -1)#este desenha a sombra nos quadrados das letras
        #este if e else é pra mudar a cor da tecla caso seja selecionado
        if letter_light is True:
            cv2.rectangle(keyboard, (x + th, y + th), (x + width - th, y + height - th),
(255,165,0),-1) #borda do quadradinho
        else:
            cv2.rectangle(keyboard, (x + th, y + th), (x + width - th, y + height - th),
(100,100,100),th)#quadrados que estao as letras

#config das letras
font_letter =cv2.FONT_HERSHEY_PLAIN
font_scale = 3
font_th = 2
text_size = cv2.getTextSize(text, font_letter, font_scale, font_th)[0]

```

```

width_text, height_text = text_size[0], text_size[1]
text_x = int ((width - width_text) / 2) + x
text_y = int ((height + height_text) / 2) + y
cv2.putText(keyboard, text, (text_x,text_y), font_letter, font_scale,
(255,255,255), font_th)

```

#for pra imprimir as teclas e tbm continuar na mesma cor caso nao seja selecionado

```

for i in range(35):
    if i == letter_index:
        light = True
    else:
        light = False

```

```
key_text = keys_set_1[i]
```

```
if key_text == "U+232B":
```

```
    key_text = "<-" # "U+232B" é o numero que significa backspace
```

```
letter(i, key_text, light)
```

def save_to_word(text, file_name):#funcao para salvar o texto em word e na area de trabalho

```

doc = docx.Document()
doc.add_paragraph(text)
documentos_folder=os.path.expanduser('~\Documents')
folder_name = 'APP'
folder_path = os.path.join(documentos_folder,folder_name)
os.makedirs(folder_path,exist_ok=True)
doc_file = os.path.join('C:\\Users\\Sarina Rodrigues\\OneDrive\\Área de
Trabalho', file_name) # Define o caminho completo para o arquivo na pasta "App"

doc.save(doc_file)

```

```
def midpoint(p1, p2):
    return int((p1.x + p2.x)/2), int((p1.y + p2.y)/2)#aqui vamos pegar o ponto
medio pra termos a linha vertical, e sera int pois como vamos dividir por 2 o pixels na
pode ser numero quebrado
```

```
font = cv2.FONT_HERSHEY_SIMPLEX
```

```
def get_blinking_ratio(eye_points, facial_landmarks):
    left_point      =      (facial_landmarks.part(eye_points[0]).x,
facial_landmarks.part(eye_points[0]).y)#ponto referencia linha horizontal inicio
    right_point     =      (facial_landmarks.part(eye_points[3]).x,
facial_landmarks.part(eye_points[3]).y)#ponto referencia linha horizontal fim
                                center_top      =
midpoint(facial_landmarks.part(eye_points[1]),facial_landmarks.part(eye_points[2]))#
ponto referencia linha vertical inicio
                                center_bottom   =
midpoint(facial_landmarks.part(eye_points[5]),facial_landmarks.part(eye_points[4]))#
ponto referencia linha vertical fim
```

```
    hor_line = cv2.line(frame, left_point,right_point, (0,255,0),1)#desenho da
linha horizontal
```

```
    ver_line = cv2.line(frame, center_top,center_bottom, (0,255,0),1)#desenho
da linha horizontal
```

```
    hor_line_lenght = hypot((left_point[0] - right_point[0]), (left_point[1] -
right_point[1]))#é a largura olho que usaremos pra soma de deteccao piscar
```

```
    ver_line_lenght = hypot((center_top[0] - center_bottom[0]), (center_top[1] -
center_bottom[1]))#é a altura do olho usaremos para soma de deteccao piscar
```

```
    ratio = hor_line_lenght / ver_line_lenght #métrica usada para avaliar a
abertura ou o fechamento do olho
```

```
    return ratio
```

```
#este é funcao pra falar oqe esta sendo escrito
```

```

def falar_texto(text):
    engine = pyttsx3.init()
    engine.say(text)
    engine.runAndWait()

#contador
frames = 0

blinking_frames = 0
frames_to_blink = 6
text = ""
text_salvo = ""

while True:
    _, frame = cap.read()#quadro que entrara imagem da camera
    frame = cv2.resize(frame, None, fx=0.5, fy=0.5)#este diminui o tamanho da
tela em que a camera esta aberta
    keyboard[:] = (0,0,0)#este significa que a cada letra que a varredura passar
a anterior deve ficar branca
    frames += 1
    gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)#convertemos para o
cinza pra ter o formato mais grave

    active_letter = keys_set_1[letter_index] #este define pegar a tecla com a
piscada

    faces = detector(gray)
    for face in faces:
        # print(face)

        landmarks = predictor(gray,face)
        left_eye_ratio = get_blinking_ratio([36, 37, 38, 39, 40, 41], landmarks)
        right_eye_ratio = get_blinking_ratio([42, 43, 44, 45, 46, 47], landmarks)

```

```

        blinking_ratio = (left_eye_ratio + right_eye_ratio) / 2 #aqui soma para
deteccao piscar
        if blinking_ratio >4:
            if active_letter == "Sv" and not save_action_triggered:
                output_file_name = "tecladovirtual.docx" # Nome do arquivo de
saída
                save_to_word(text, file_name) # Chama a função para salvar o texto
no documento do Word
                save_action_triggered=True
            #save_to_word(text)

        if active_letter == "U+232B":#funcao backspace
            if not backspace_pressed:
                if text:
                    text = text[:-1] # Usa slicing para apagar a última letra
                    backspace_pressed = True
            else:
                backspace_pressed = False # Reset backspace_pressed se outra
tecla estiver ativa

        if active_letter == "Del" and not last_action_del:
            if text:
                text_salvo = text
                text = text[:-1]
                last_action_del = True
            else:
                last_action_del = False

        cv2.putText(frame,"Piscou",(5,70), font, 1, (0,255,0),thickness=3)
        blinking_frames += 1
        frames -= 1 #este faz a varredura parar qd fecho os olhos

        #print(active_letter)

```

```

        if blinking_frames == frames_to_blink: #tempo que o olho tem que ficar
            fechado pra pegar a letra
                if active_letter != "<" and active_letter != "_" and active_letter != "Del"
                    and active_letter != "U+232B" and active_letter != "Sv":
                        text += active_letter
                        print(board)
                    if active_letter == "_":
                        text += " "
                    if active_letter == "Del":
                        if text :
                            text_salvo = text
                            text=text[:-1]
                    if active_letter == "U+232B":
                        if not backspace_pressed:
                            if text:
                                text = text[:-1] # Usa slicing para apagar a última letra
                                backspace_pressed = True
                        else:
                            backspace_pressed = False # Reset backspace_pressed se outra
tecla estiver ativa
                    if active_letter == "Sv" and not save_action_triggered:
                        output_file_name = "tecladovirtual.docx"
                        save_to_word(text, file_name) # Chama a função para salvar
o texto no documento do Word
                        save_action_triggered=True

        if active_letter == "<":
            select_keyboard = True
            letter_index = 0

```


letra sound.play()#esta linha e a de baixo faz sair o som ao selecionar a

```
#select_keyboard = True
#time.sleep(1)
```

```
falar_texto(text)#esta linha pede pra falar oqe ta escrito
```

```
else:
    blinking_frames = 0
```

```
#varredura
#controleFrame();
    if frames == 21: #tempo varredura
        letter_index += 1
        frames = 0
    if letter_index == 35:
        letter_index = 0
        #frames = 0
```

codigo #aqui preciso desenhar as letras novamente para que apareca dentro deste

```
        lines = [text[j:j + max_characters_per_line] for j in range(0, len(text),
max_characters_per_line)]
        board[:] = 255 # Limpar o conteúdo anterior de 'board'
        y = 30
        for line in lines:
            cv2.putText(board, line, (10, y), font, 1.0, 0, 2)
            y += 30 # Incrementa a posição vertical para a próxima linha
```

```
# Laço para desenhar as letras no teclado e manter a cor correta
for i in range(35):
    if i == letter_index:
```

```

        light = True
    else:
        light = False

```

```

key_text = keys_set_1[i]

```

```

if key_text == "U+232B":
    key_text = "<-"

```

```

letter(i, key_text, light)

```

```

y = 30

```

```

for line in lines:

```

```

    cv2.putText(board, line, (10, y), font, 1.0, 0, 2)

```

```

    y += 30 # Incrementa a posição vertical para a próxima linha

```

```

#cv2.putText(board, text, (10,60), font, 1.5, 0, 3) # cria a caixa de texto que
mostra as letras capturada.

```

```

cv2.imshow("Frame", frame) #mostra o quadro com nossa face

```

```

cv2.imshow ("Virtual Keyboard", keyboard)

```

```

cv2.imshow ("Board", board)

```

```

key = cv2.waitKey(1)#esc para fechar a tela

```

```

if key == 27:

```

```

    break

```

```

cap.release()

```

```

cv2.destroyAllWindows()

```

