

Reconhecimento e personalização de gestos das mãos para interação com páginas web

Daniel Gustavo Favero
Universidade Tecnológica Federal do
Paraná
Pato Branco, Paraná, Brazil
daniel.favero@alunos.utfpr.edu.br

Soelaine Rodrigues Ascari
Universidade Tecnológica Federal do
Paraná
Pato Branco, Paraná, Brazil
soelaine@utfpr.edu.br

Rúbia Eliza de Oliveira Schultz
Ascari
Universidade Tecnológica Federal do
Paraná
Pato Branco, Paraná, Brazil
rubia@utfpr.edu.br

Resumo

The Hand Mouse browser extension presented in this paper enables mouse control on web pages by means of the user's hand gestures. This tool aims to enhance flexibility and ergonomics, offering a natural alternative to the traditional use of the mouse, potentially reducing the risk of repetitive strain injuries. The main innovations of this paper are the usability studies within the extension's user interfaces and the development of the gesture customization feature. This allows the user to define and train the AI model used in the software with new gestures to perform actions such as moving the cursor, clicking, scrolling up, scrolling down, going back a page, going forward a page and free movement.

Keywords

Human-computer interaction, Gesture Interaction, Usability, Open Source Software

1 Introdução

A preocupação principal da Interação Ser Humano-Computador (ISHC) é garantir uma usabilidade flexível, produtos interativos que proporcionam uma experiência agradável para usuários de sistemas digitais, por meio de ferramentas que tornam o uso do computador mais adaptável ao usuário [14]. Assim, novas formas de interação entre os seres humanos e computadores surgiram por meio de estudos e avanços tecnológicos em *hardware* e algoritmos de Inteligência Artificial (IA) [10].

Em um caso particular, a interação que utiliza as mãos e gestos das mãos do usuário para o controle do mouse traz uma abordagem mais natural de manipulação, visto que humanos expressam muitos sentimentos por meio de gestos [8]. Além disso, essa forma de interação tem a possibilidade de reduzir dores causadas por movimentos repetitivos e vigorosos [5].

Diversos *softwares* de interação gestual surgiram nos últimos anos e estão presentes em diversas plataformas, como um aplicativo *desktop*, um console de videogame, um jogo eletrônico, entre outros exemplos. A *Hand Mouse* [15] é uma extensão de navegador (aplicativo instalado no navegador que oferece novas funcionalidades) que permite o controle do mouse em páginas *web* por meio dos gestos das mãos do usuário.

Há poucos anos, era quase impossível executar algoritmos de Aprendizado de Máquina em um ambiente como o navegador, devido à complexidade na hospedagem desses serviços. O cenário mudou quando bibliotecas como o *Tensorflow.js* [16] trouxeram soluções capazes de serem executadas diretamente na *web*, sem a

necessidade de instalação de dependências adicionais. Com isso, a *Hand Mouse* utiliza o modelo *Hand Pose Detection* do *Tensorflow.js* e a biblioteca *Fingerpose* [13] para identificar os *landmarks* de um frame do vídeo de uma *webcam* e classificar o gesto que o usuário está executando.

Entretanto, a versão inicial da *Hand Mouse* permite a classificação de cinco gestos, correspondentes às ações de mover o mouse, clique, rolar para cima, rolar para baixo e mover livremente. Toda a classificação foi definida por meio de código fonte estático. Isso limita a utilização da ferramenta para um público muito específico, pois está fixa a forma como os gestos devem ser executados, tirando a flexibilidade e adaptação do usuário ao *software*.

Diante desse contexto, este artigo tem como objetivo explorar conceitos de usabilidade para o desenvolvimento da funcionalidade de personalização de gestos na *Hand Mouse* e suas interfaces de usuário envolvidas. Além disso, novos gestos de avançar uma página e retroceder uma página foram adicionados à ferramenta.

2 Trabalhos relacionados

Com o objetivo de aprimorar a fundamentação do trabalho e seu desenvolvimento em quesitos relacionados a melhores práticas, ferramentas, as lacunas e oportunidades da problemática da pesquisa, uma busca de trabalhos relacionados foi realizada, utilizando a plataforma *Google Scholar* e projetos de código-fonte aberto no *GitHub*¹. Um breve relato dos principais resultados são apresentados a seguir.

A biblioteca *Handsfree.js* desenvolvida por Ramos [7] é de código-fonte aberto e baseia-se nos modelos da biblioteca *TensorFlow.js*. *Handsfree.js* possibilita a interação gestual com páginas *web* usando as mãos, face e corpo. Em versão recente permite a criação e identificação de gestos da mão de forma personalizada. Essa personalização é feita utilizando a biblioteca *Fingerpose* de Schallwig and Wendel [13], que atua como classificador de gestos, baseando-se em pontos do esqueleto da mão humana, os *landmarks*, obtidos das imagens da *webcam*.

A metodologia MyPGI (*Methodology to yield Personalized Gestural Interaction*) [1], tem como foco usar gestos personalizados para auxiliar na comunicação de pessoas com dificuldades motora e de fala. A metodologia foi validada com o desenvolvimento do sistema PGCA (*Personal Gesture Communication Assistant*), que disponibiliza dois classificadores para classificar gestos personalizados, um baseado em Redes Neurais Convolucionais e outro baseado em Máquina de Vetores de Suporte (SVM do inglês *Support Vector*

¹ *GitHub* é uma plataforma de hospedagem de código-fonte e arquivos com controle de versão usando a ferramenta *Git*.

Machine), que extraem automaticamente informações importantes das imagens, também chamadas de características.

No trabalho desenvolvido por Santos [12] foi criado o sistema *CONTR0LL* para pessoas com problemas motores ou degenerativos, o qual utiliza diversos modelos de Aprendizado de Máquina para reconhecimento de partes do corpo para o controle de jogos de navegador, entre os modelos estão: o *PoseNet* e o *ml5* para reconhecer poses do corpo; o *MobileNetV1* para identificar pontos na face; e a biblioteca *Handtrack.js*.

Com base nos estudos mapeados observou-se a viabilidade de empregar diversas bibliotecas e técnicas de Aprendizado de Máquina para o reconhecimento de gestos. É importante ressaltar também a preocupação dos estudos nos aspectos de Interação Ser Humano-Computador, visto que muitos deles estavam relacionados a um público com limitações motoras, o que exigiu a presença constante de um instrutor para auxiliar na calibração e configuração do *software* de interação. Notou-se também que os trabalhos trouxeram o usuário para interagir em apenas uma aplicação, desenvolvida para o propósito do trabalho.

A fim de proporcionar uma nova forma de interação com páginas *web* para usuários que utilizam individualmente um navegador de internet no cotidiano, neste trabalho foi utilizado a biblioteca *TensorFlow.js* e o seu modelo de CNN *Hand Pose Detection*, similar ao trabalho de Ramos [7] que também é voltado para interação na *web*. Porém no caso deste trabalho, ao invés de uma biblioteca de código aberto, o modelo foi aplicado em uma extensão de *browser*.

3 Fundamentação Teórica

3.1 Interação Ser Humano-Computador

A ISHC é uma ampla área dentro da Ciência da Computação, em que ainda se debate a sua gama de conceitos e definições, visto que novas tecnologias surgem diariamente, apresentando inovações e novas formas de interação. Hewett et al. [3, p.5] afirma que segundo a ACM (*Association for Computing Machinery*) a ISHC é "Uma disciplina preocupada com o *design*, avaliação e implementação de sistemas computacionais interativos para o uso humano e com o estudo dos principais fenômenos que os rodeiam".

Um dos objetivos da ISHC é criar mecanismos capazes de traduzir as ações do usuário para o mundo digital, seja ele uma interface de tela, um dispositivo físico para interação ou o uso de tecnologias existentes adaptadas para o uso de computadores. Com base nesses princípios o conceito de *design* de interação usado por Rogers et al. [9, p.28] se trata do "*Design* de produtos interativos que fornecem suporte às atividades cotidianas das pessoas, seja no lar ou no trabalho". Entende-se, portanto, que a qualidade de produtos interativos está conectada com critérios de usabilidade e acessibilidade do usuário nas suas tarefas rotineiras.

A ISO 9241-11:1998 (*International Organization for Standardization*) define o termo usabilidade como uma extensão na qual os produtos podem ser usados por usuários específicos para alcançar determinado objetivo com eficácia, eficiência e satisfação em um contexto específico de uso [4]. Para que o usuário atinja esses objetivos e aproveite as principais funcionalidades do sistema, esse deve ser flexível e atender ao máximo as necessidades daquele que irá utilizá-lo.

Ao combinar tecnologias e resultados obtidos em diferentes pesquisas em sistemas computacionais, possibilita-se que usuários interajam de maneiras distintas com esses sistemas. É perceptível a aplicação dessa combinação em produtos de casa inteligente controlados por voz, sensores biométricos, tecnologias de *eye tracking* para interação utilizando a visão, entre outros exemplos.

Ao buscar-se uma abordagem mais natural de interação entre seres humanos e computadores, sistemas de interação gestual se tornam uma abordagem viável, capazes de utilizar os gestos de mãos do usuário para o controlar e executar ações dentro de um sistema. Além disso, esses sistemas podem promover a criatividade do usuário e melhorar questões ergonômicas, possivelmente evitando lesões por esforço repetitivo devido ao uso excessivo do mouse [5].

3.2 Visão Computacional

De acordo com Ballard and Brown [2, p.2], "a Visão Computacional é a iniciativa de automatizar e integrar uma grande variedade de processos e representações usadas por percepções visuais". Entre esses processos estão o processamento de imagens e o reconhecimento de padrões em imagens.

No contexto de interação gestual, as imagens das mãos do usuário são capturadas utilizando uma câmera ou dispositivo de sensoriamento. Em algumas situações é preciso de uma etapa de pré-processamento para melhorar a qualidade da imagem [6]. Logo em seguida, o gesto é separado do restante da imagem (geralmente o fundo é excluído) por meio de rótulos atribuídos a grupos de *pixels*. Com essa região de interesse separada é possível extrair algumas características para o reconhecimento e classificação da imagem, permitindo posteriormente, realizar uma ação com base no que foi classificado. São exemplos dessas informações: a mão humana, o tamanho, formato, a posição das juntas dos dedos da mão, a posição dos dedos, cor dos *pixels* da mão, entre outros.

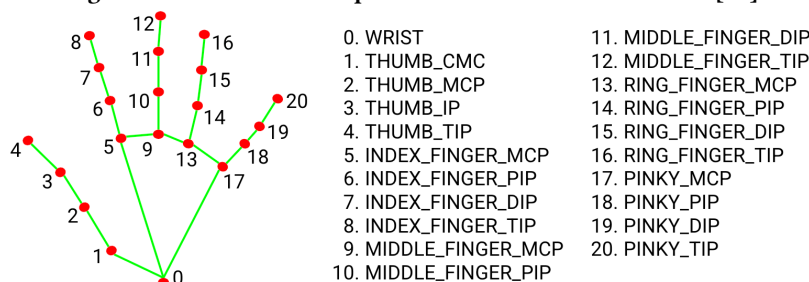
3.3 Classificação de gestos

Samuel [11], um dos pioneiros do ramo de Inteligência Artificial, descreve Aprendizado de Máquina como um processo de ensinar os computadores a aprenderem a solução de um problema por meio de experiências humanas. Sejam eles a classificação de imagens, análise e síntese de texto, carros autônomos e quaisquer outros problemas que o cérebro humano poderia aprender.

Algoritmos que buscam resolver o problema de classificação de dados se encontram dentro do paradigma de Aprendizado Supervisionado. Com relação a classificação de gestos, o método baseado nos *landmarks* da mão obtém um ótimo controle sobre as posições de cada dedo da mão do usuário, assim como no trabalho realizado por Ramos [7].

A biblioteca *Hand Pose Detection* do *Tensorflow.js* extrai 21 *landmarks* que compõem o esqueleto da mão. A representação desses 21 pontos é ilustrada na Figura 1.

Essa abordagem de classificação dos gestos com base nas coordenadas dos *landmarks* consiste na utilização de dois métodos de Aprendizado de Máquina, o uso do modelo *Hand Pose Detection* e logo em seguida o uso de um algoritmo de classificação.

Figura 1: Pontos extraídos pelo modelo *Hand Pose Detection* [16]

4 Metodologia

A combinação das áreas de Visão Computacional, Inteligência Artificial e ISHC deu origem a *Hand Mouse*, uma extensão de navegador de código fonte aberto, cujo objetivo principal é permitir que o usuário crie sua própria forma de interação dentro de páginas *web* utilizando os gestos de sua mão. Essa extensão foi usada como base para o desenvolvimento deste trabalho.

O desenvolvimento da funcionalidade de personalização de gestos foi dividido em três etapas. A primeira envolveu testes de algoritmos de classificação de gestos baseados em *landmarks* extraídos de imagens de mãos. Já a segunda e terceira etapas foram implementadas e testados os algoritmos na extensão *Hand Mouse*, buscando estudar requisitos de usabilidade para que os usuários pudessem usufruir da ferramenta de forma mais eficaz.

4.1 Materiais

Para o propósito de implementação da solução em uma extensão de navegador, as linguagens HTML (*Hyper Text Markup Language*), CSS (*Cascading Style Sheets*) e Javascript foram utilizadas. Entretanto, o Javascript não infere tipagens nas variáveis usadas no código, o que pode causar problemas na passagem de parâmetros com tipos errados para funções, reduzindo a previsibilidade do código, pois a validação de tipos não é feita pela linguagem. Para resolver esse problema, a linguagem Typescript ofereceu a possibilidade de adicionar tipagens estáticas no código Javascript.

Com isso, a lista de materiais que foram utilizados no projeto se encontram na Tabela 1.

Tabela 1: Lista de ferramentas e tecnologias utilizadas para o desenvolvimento deste trabalho

Ferramenta/Tecnologia	Versão
Visual Studio Code	1.81.1
HTML	5.0
CSS	3.0
Javascript	ECMAScript 2021
Typescript	5.1
TensorFlow.js	4.10.0
Webcam	-
Google Chrome	116.0.5845.97

4.2 Método

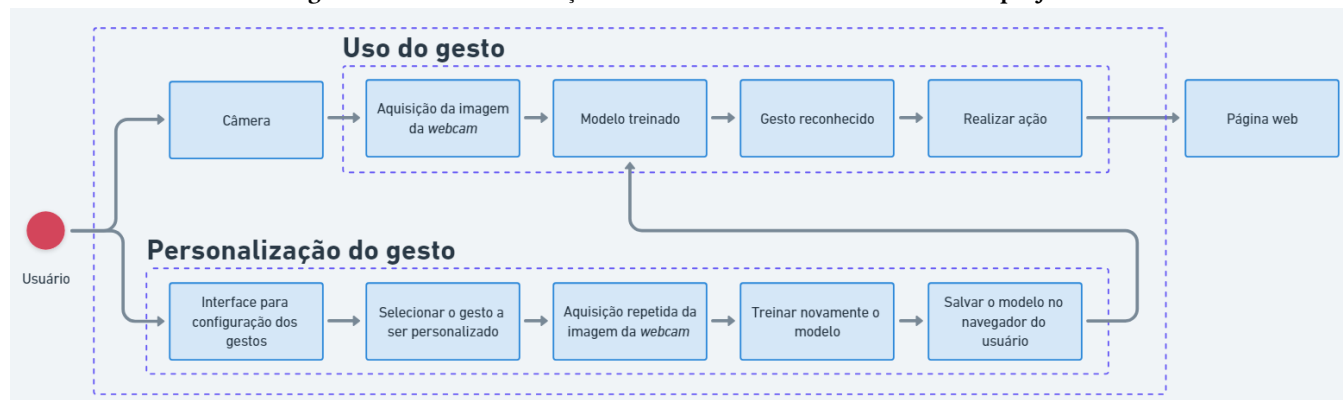
O desenvolvimento da funcionalidade de personalização de gestos da *Hand Mouse* foi dividida em três etapas. Na primeira, em um ambiente do Google Colab, alguns algoritmos de classificação foram testados para encontrar os melhores parâmetros e o tamanho mínimo do conjunto de testes. Além disso, o modelo de reconhecimento de mãos do *Tensorflow* foi utilizado, por meio do *framework* MediaPipe da Google.

Em um segundo momento, uma página *web* foi desenvolvida para possibilitar a personalização de gestos por meio de menus e botões. Para sua elaboração foi utilizado HTML como linguagem de construção de páginas, CSS para sua estilização e Javascript como linguagem de programação.

Na última etapa, o modelo de inteligência artificial foi submetido a testes na extensão, executando ela em páginas *web* para detectar possíveis problemas. O código implementado na *Hand Mouse* precisou ser repensado para acomodar a personalização dos gestos, também foram necessários ajustes na sensibilidade de detecção dos gestos, ou seja, como o modelo identifica os gestos em tempo real e, em alguns casos, por falta de precisão do modelo, o gesto indentificado era o incorreto, resultando na execução de uma ação do mouse indesejada.

Com esses estágios finalizados, a extensão passou a ter dois fluxos de utilização, como ilustrado na Figura 2. O primeiro fluxo (Uso do gesto) consiste no usuário utilizar os gestos já cadastrados, e disponibilizados por padrão na ferramenta, para interagir com as páginas *web*. Inicialmente, sua *webcam* captura as imagens dos gestos em tempo real, que servem como entrada para o modelo de inteligência artificial. Esse, por sua vez, identifica o gesto correspondente, possibilitando a execução de uma ação na página *web* com base nesse gesto.

No segundo fluxo (Personalização do gesto), o usuário acessa a interface destinada à personalização dos gestos. Nessa interface, uma lista exibe os gestos disponíveis para customização. Ao selecionar um desses gestos, uma nova tela é aberta e o usuário é notificado de que uma sequência de capturas de imagem será realizada. Posteriormente, essas imagens serão pré-processadas para registrar os *landmarks*, que representam um gesto, então esses dados são usados para treinar novamente o modelo. Concluído esse processo, o modelo é armazenado no navegador do usuário, passando a ser incorporado como parte integral do primeiro fluxo.

Figura 2: Fluxos de utilização da extensão *Hand Mouse* ao final do projeto

A avaliação dessas etapas envolveu uma profissional da área de ISHC, que analisou a página desenvolvida e os quesitos de usabilidade da ferramenta. Ela sugeriu algumas melhorias na posição dos elementos da página, visando reduzir o esforço do usuário ao realizar as atividades necessárias na interface.

5 Resultados

Neste capítulo, são apresentados os resultados do trabalho realizado, os quais foram obtidos por meio dos materiais e método propostos.

5.1 Escolha do modelo de classificação de gestos

Para realizar os testes nos algoritmos de classificação, um conjunto de treinamento de autoria própria foi criado. Sua construção simula a customização dos gestos do usuário final. Ele consiste em sete classes de gestos, correspondentes às ações de mover o cursor do mouse, clique, rolar para cima, rolar para baixo, voltar uma página, avançar uma página e mover livremente. Cada classe possui 120 imagens, totalizando 840 imagens.

Dentre os algoritmos existentes, o KNN (sigla em inglês para *K-nearest neighbors*) e o SVM foram testados. Ambos citados em trabalhos relacionados que fazem uso das mesmas abordagens que foram utilizadas e possuem resultados testados na classificação de gestos.

Testes foram executados para encontrar a quantidade mínima de imagens necessárias para classificar um gesto, a melhor distância para classificar os gestos e se gestos similares interferem no reconhecimento. Todos considerando a acurácia e o *score* F1 como critérios de análise dos algoritmos.

Ao final, o modelo de classificação escolhido foi o KNN, obtendo uma acurácia de 85,19% e *score* F1 de 83,54%, enquanto o SVM obteve uma acurácia de 77,78% e *score* F1 de 70,82%. Com relação aos parâmetros do KNN escolhidos, tem-se $k = 1$, critério de votação baseado na quantidade de vizinhos. O conjunto de dados mínimo validado para detecção de gestos foi de 20 imagens, em que aproximadamente 15 serão usadas para treino (representando a divisão de aproximadamente 70% do conjunto de 20 imagens).

5.2 Aplicação do Modelo na extensão *Hand Mouse*

O primeiro passo para implementar a nova funcionalidade foi atualizar o código existente na extensão, o qual necessitou de ajustes, pois a versão inicial do projeto contava com um modelo de classificação de gestos, assim como o trabalho atual, porém a classificação dos gestos era definida de forma estática por meio de código Javascript utilizando a biblioteca *Fingerpose* [13], impossibilitando a personalização.

Ao instalar a extensão, a página de edição de gestos é aberta e o modelo de classificação é iniciado com um conjunto de gestos predefinidos. Além disso, ao acessar uma página *web* um *preview* da *webcam* do usuário é exibido na interface, para o usuário verificar a execução dos seus gestos.

Além disso, ao adicionar uma nova extensão, os navegadores adicionam por padrão um botão na barra de ferramentas para o usuário acessar um *popup*, o mesmo ilustrado na Figura 3.

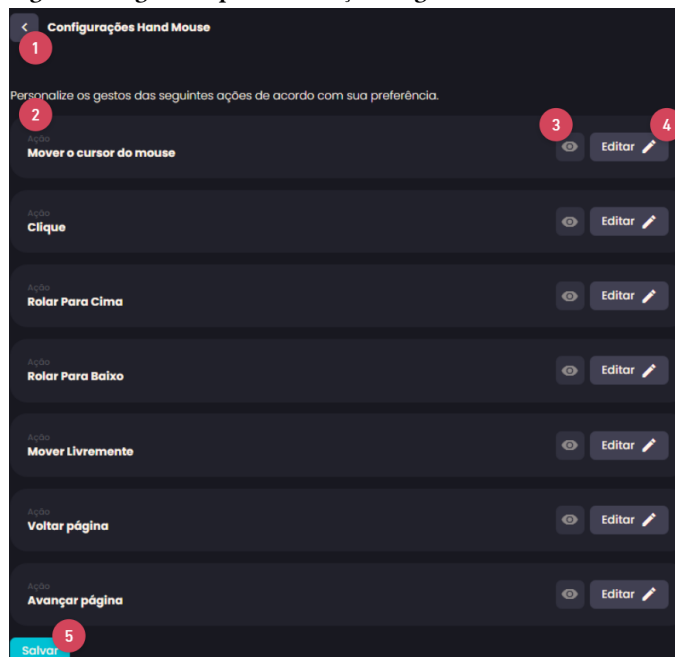
Figura 3: *Pop-up* da extensão *Hand Mouse*

Entre as opções disponíveis no *popup*, estão:
(1) Desabilitar a *Hand Mouse*;

- (2) Desligar o *preview* da *webcam*;
- (3) Acessar a página de personalização de gestos.

A página *web* de personalização de gestos foi criada utilizando HTML, CSS, Javascript e Typescript, como mostrado na Figura 4.

Figura 4: Página de personalização de gestos da *Hand Mouse*



O usuário ao acessar a página de personalização de gestos apresentada da Figura 4 pode realizar as seguintes ações:

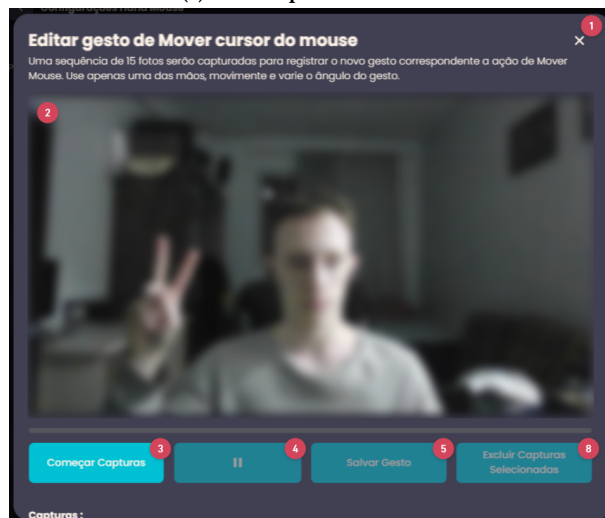
- (1) Sair da página;
- (2) Visualizar a lista de gestos que ele pode personalizar;
- (3) Visualizar uma prévia do gesto cadastrado;
- (4) Personalizar (Editar) o gesto;
- (5) Salvar as alterações realizadas.

A primeira versão da tela não possuía a funcionalidade de visualizar a prévia do gesto cadastrado (3). Essa funcionalidade foi inserida a partir da sugestão de uma especialista da área de ISHC, que justificou sua adição por permitir ao usuário lembrar o gesto cadastrado, especialmente se ele o personalizou há muito tempo e não lembra como executá-lo.

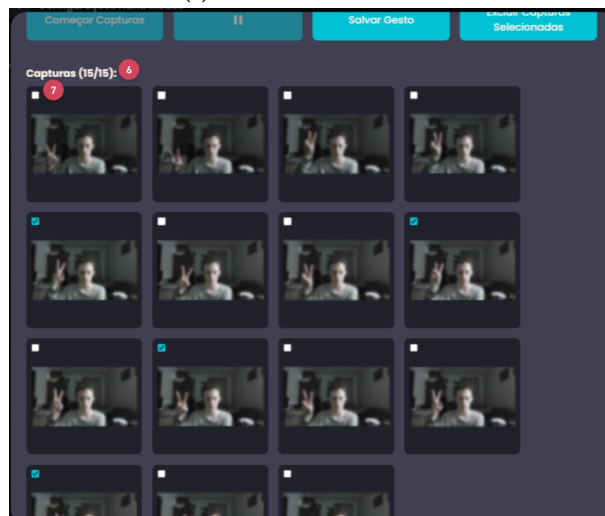
Para personalizar um gesto o usuário clica no botão "Editar" (4) da Figura 4 correspondente a uma ação e uma *modal* (Elemento gráfico que permanece fixo por cima dos outros elementos da página) é aberta, como mostrado nas Figuras 5a e 5b.

Figura 5: Modal de personalização de gestos

(a) Parte superior da modal



(b) Parte inferior da modal



Essa *modal* possibilita a personalização do gesto da ação selecionada. A partir de então, as seguintes ações podem ser executadas dentro da *modal*:

- (1) Fechar a *modal*;
- (2) Visualizar uma prévia da imagem da *webcam*;
- (3) Começar a captura das imagens;
- (4) Pausar as capturas;
- (5) Salvar gestos. Essa ação é habilitada quando a quantidade de 15 capturas foi atingida;
- (6) Visualizar a quantidade de capturas restantes;
- (7) Selecionar uma captura para excluí-la, caso o usuário queira repetir uma captura que não foi devidamente registrada;

- (8) Excluir as capturas selecionadas. O usuário poderá realizar essa ação enquanto o processo de captura estiver pausado (Botão (4) da Figura (5a)), ou logo após todas as capturas terem sido realizadas.

A posição dos botões "Começar Capturas"(3), pausar as capturas (4), "Salvar gestos"(5) e "Excluir Capturas Selecionadas"(8) apresentados na *modal* da Figura 5a foram sugeridos por um profissional em usabilidade. Eles estão posicionados próximos entre si e entre a região de visualização das capturas (7), com o objetivo de reduzir o esforço físico do usuário ao movimentar o mouse para interagir com essas ações.

O processo se inicia no clique do botão "Começar Capturas" (3), a partir de então uma sequência de 15 fotos são capturadas de forma sequencial, o intervalo entre as capturas é de dois segundos, permitindo que o usuário mude a posição, orientação e direção do gesto em tempo.

Durante as capturas o usuário tem a possibilidade de pausar o processo pelo tempo que desejar por meio do botão (4). Isso evita que interrupções, como distrações externas ou falta de atenção do usuário impeçam que as capturas sejam realizadas da forma correta. O mesmo botão é utilizado para dar prosseguimento às capturas.

Somente ao final, com todas as 15 fotos capturadas, o botão "Salvar Gesto" (5) é desbloqueado, os *landmarks* que compõem cada gesto das imagens são obtidos do modelo *Hand Pose Detection* e são salvos no armazenamento interno do navegador. Finalmente, é possível fechar a *modal* (1).

Com o novo gesto salvo, o usuário é redirecionado novamente para a tela da Figura 4, podendo editar novamente o gesto ou escolher um novo gesto para personalizar.

No clique do botão "Salvar" (5) da Figura 4, o modelo de classificação de gestos será retreinado e salvo no armazenamento interno do navegador. Ao abrir uma nova guia do navegador em uma página *web*, a extensão estará pronta para uso com o novo gesto configurado.

Destaca-se que, ao final do desenvolvimento da página de personalização de gestos o cursor do mouse utilizado não é o mesmo controlado pelo mouse físico. Durante a utilização da extensão, o desenho de cursor do mouse é o apresentado na Figura 6.

Figura 6: Cursor de mouse padrão da *Hand Mouse*



O movimento do cursor do mouse segue o centro da mão do usuário e cada gesto modifica esse desenho de uma maneira diferente. A Figura 7 ilustra os desenhos do cursor para os gestos de mover o cursor do mouse (1), clique (2), rolar para cima (3), rolar para baixo (4), voltar uma página (5), avançar uma página (6) e movimento livre (7). Esse detalhe permite que o usuário identifique o gesto sendo executado de forma visual.

Outro detalhe do cursor do mouse é a possibilidade de grudar em elementos clicáveis ao se aproximar deles, como botões, *links*, imagens, entre outros. Facilitando a execução da ação de clique.

Como o gesto é reconhecido rapidamente pelo modelo, em um primeiro teste, algumas ações, como clique, voltar página e avançar página, eram executadas de forma descontrolada pelo usuário, podendo deixá-lo confuso sobre a utilização do mouse. Em função disso, foi adicionado um *delay* de aproximadamente três segundos para execução da ação.

5.3 Desafios encontrados

Durante o desenvolvimento, diversos obstáculos foram encontrados, entre os principais, durante a implementação do modelo na extensão, destaca-se o gerenciamento de memória necessário para utilização da estrutura de dados tensor do *TensorFlow.js*, uma generalização dos vetores em dimensões maiores [16]. O tensor foi utilizado para armazenar os *pixels* das imagens capturadas em tempo real pela *webcam* do usuário. A biblioteca aloca essa estrutura na memória da GPU (sigla em inglês para *Graphics Processing Unit*) e, após a utilização da imagem, o espaço de memória alocado para o tensor não é liberado automaticamente, o que resultava no travamento do *browser* após alguns minutos de uso da extensão. Após a leitura da documentação do *TensorFlow.js*, foi possível perceber que a biblioteca oferece o método *dispose*, que libera a memória da GPU associada a um determinado tensor. Isso resolveu o problema não ocasionando novamente o travamento do navegador.

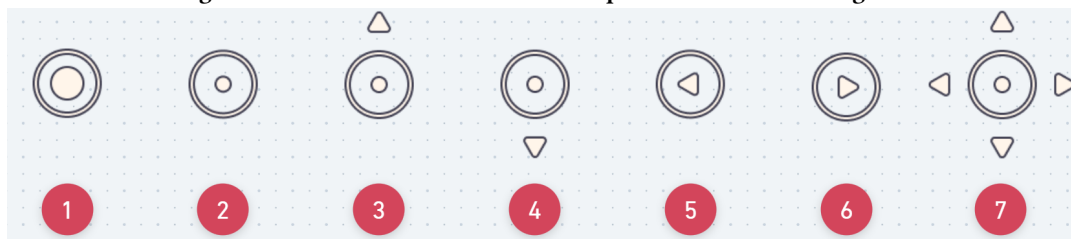
Além disso, outros ajustes relacionados à usabilidade da ferramenta foram realizados. O principal deles foi o mapeamento dos *pixels* da imagem da *webcam* para a tela do navegador. Diferentes *webcams*, que podem ser utilizadas por diferentes usuários, possuem alturas e larguras de imagem diferentes do monitor onde a página *web* está sendo exibida. Essa diferença de proporções resultava em um movimento do mouse limitado apenas às dimensões da câmera, não abrangendo toda a página *web*. A solução para esse problema envolveu redimensionar a imagem da *webcam* para ocupar todos os *pixels* da página *web*. Esse dimensionamento é feito apenas nos *landmarks*, sem distorcer a imagem da *webcam*.

6 Conclusão

Este trabalho teve como objetivo desenvolver a funcionalidade de personalização de gestos de mãos para o controle do mouse em páginas *web* na extensão de navegador *Hand Mouse*. Para atingir esse objetivo, foram revisados diversos conceitos de usabilidade em ISHC, além da aplicação de Visão Computacional e algoritmos de Aprendizado de Máquina.

A base de código original do projeto precisou de alguns ajustes para acomodar o novo classificador de gestos, além do desenvolvimento de uma página *web* específica para permitir a personalização dos gestos.

Logo em seguida, a página criada para a personalização dos gestos seguiu os princípios de usabilidade, proporcionando ao usuário o controle total sobre as ações executadas. Um especialista em usabilidade avaliou a interface do sistema e recomendou mudanças, como a alteração da posição de alguns elementos. Após a implementação dessas modificações, o especialista reavaliou a interface e verificou que as alterações realizadas atenderam aos critérios de usabilidade.

Figura 7: Desenhos do cursor do mouse para os sete diferentes gestos

A extensão *Hand Mouse* está disponível para os usuários do *Mozilla Firefox* na *Firefox Browser Add-ons*. Além disso, o código-fonte do projeto é aberto e pode ser acessado no *GitHub*.

Portanto, conclui-se que a ferramenta atingiu todos os objetivos propostos neste trabalho e abriu portas para novas funcionalidades. Além disso, ela colocou o usuário no centro da interação com o computador, permitindo-lhe utilizar sua criatividade e personalizar uma ferramenta amplamente utilizada na computação, o mouse. A *Hand Mouse* permitirá diversos estudos com a possibilidade de aprimorar a experiência do usuário na *web*, e seu método proposto, utilizando algoritmos de inteligência artificial, permite a personalização das ações que o usuário realiza nessas páginas. Por fim, ela contribui como um *software* que usuários de navegadores podem utilizar no seu dia a dia.

Referências

- [1] Rúbia Eliza de Oliveira Schultz Ascari. 2020. Metodologia baseada em visão computacional e aprendizado de máquina para guiar o projeto de sistemas de comunicação aumentativa e alternativa usando interação gestual personalizada. (2020).
- [2] DH Ballard and CM Brown. 1982. Computer Vision. *Image* 17 (1982), 2.
- [3] Thomas T. Hewett, Ronald Baecker, Stuart Card, Tom Carey, Jean Gasen, Marilyn Mantei, Gary Perlman, Gary Strong, and William Verplank. 1992. *ACM SIGCHI Curricula for Human-Computer Interaction*. Technical Report. New York, NY, USA.
- [4] W Iso. 1998. *The international organization for standardization* 45, 9 (1998).
- [5] Taufiq Morshed, Md Zahid Ferdous, et al. 2020. Outcome of surgical resection of de quervain's stenosing tenosynovitis. *Clin Pract* 17 (2020), 1482–1485.
- [6] Krishna Kumar Patel, A Kar, SN Jha, and MA Khan. 2012. Machine vision system: a tool for quality inspection of food and agricultural products. *Journal of food science and technology* 49 (2012), 123–141.
- [7] Oz Ramos. 2018. *Handsfree.js*. <https://handsfreejs.netlify.app/> Acesso em: 10 nov. 2023.
- [8] Siddharth S Rautaray and Anupam Agrawal. 2015. Vision based hand gesture recognition for human computer interaction: a survey. *Artificial intelligence review* 43 (2015), 1–54.
- [9] Yvonne Rogers, Helen Sharp, and Jennifer Preece. 2013. *Design de interação*. Bookman Editora, Porto Alegre, RS, Brasil.
- [10] S.J. Russell and P. Norvig. 2022. *Inteligência Artificial - Uma Abordagem Moderna*. GEN LTC. <https://books.google.com.br/books?id=5Xf0zwEACAAJ> Acesso em: 01 nov. 2023.
- [11] A. L. Samuel. 1959. Some Studies in Machine Learning Using the Game of Checkers. *IBM Journal of Research and Development* 3, 3 (1959), 210–229. <https://doi.org/10.1147/rd.33.0210>
- [12] André Vieira dos Santos. 2021. *CòNTRòLL: sistema de reconhecimento de gestos e gerenciamento de ações em jogos digitais*. Master's thesis. Universidade Federal do Rio Grande do Norte.
- [13] Andreas Schallwig and Erick Wendel. 2021. *Fingerpose*. <https://github.com/andypotato/fingerpose#define-your-own-gestures> Acesso em: 10 nov. 2023.
- [14] H. Sharp, J. Preece, and Y. Rogers. 2019. *Interaction Design: Beyond Human-Computer Interaction*. Wiley. <https://books.google.com.br/books?id=UDeQDwAAQBAJ>
- [15] Jeferson Souza, Daniel Favero, and Gustavo Nunes. 2023. *Hand Mouse browser extension*. <https://doi.org/10.5281/zenodo.8080440>
- [16] TensorFlow.js. 2018. *TensorFlow.js | Machine learning para desenvolvedores JavaScript*. <https://www.tensorflow.org/js> Acesso em: 15 out. 2023.