"Tente Não Piscar": Um Jogo de Detecção de Piscadas Usando Visão Computacional

Juliana Aragão Pinto[1], **Victor Travassos Sarinho**[1]

[1]Laboratório de Entretenimento Digital Aplicado (LEnDA)
Departamento de Exatas (DEXA)
Universidade Estadual de Feira de Santana (UEFS)
Feira de Santana – BA – Brasil

aragaopintojuli@gmail.com, vsarinho@uefs.br

Abstract. *This article presents the development of an interactive game titled "Try Not Blink" that utilizes computer vision for blink detection. Based on one of the world's most traditional games, "Try not Blink" is designed to be played by two participants who position themselves in front of a camera and authorize its access through the browser. Players are challenged to avoid blinking for as long as possible to outlast their opponents. Through the TensorFlow.js library, the blinking activity is monitored in real time. This research demonstrates the potential of facial landmark detection techniques for playful applications.*
Keywords: *Digital Game; TensorFlow; Landmarks; Human-Computer Interface; Eye Blinking.*

Resumo. *Este artigo apresenta o desenvolvimento de um jogo interativo intitulado "Tente Não Piscar" que utiliza visão computacional para detecção de piscadas. Baseado em um dos jogos mais tradicionais do mundo, "Tente Não Piscar" é projetado para ser jogado por dois participantes, os quais devem se posicionar em frente a uma câmera e autorizar o acesso da mesma pelo navegador. Os jogadores são desafiados a evitar piscar pelo máximo de tempo possível para superar seus oponentes. Por meio da biblioteca TensorFlow.js, a atividade de piscar é monitorada em tempo real. Esta pesquisa demonstra o potencial das técnicas de detecção de landmarks faciais para aplicações lúdicas.*
Palavras-chave: *Jogo Digital; TensorFlow; Landmarks; Interface Homem-Computador; Piscar dos Olhos.*

1. Introdução

A interação humano-computador desempenha um papel fundamental na forma como os indivíduos se envolvem com a tecnologia e as aplicações digitais. Em jogos digitais, especificamente, a capacidade de detectar e interpretar as ações dos jogadores de forma precisa e rápida é crucial para oferecer experiências imersivas e envolventes [Backe 2018]. Nesse sentido, a detecção de piscadas tem se mostrado uma área de pesquisa promissora para aprimorar a interação entre humanos e computadores.

As piscadas são um comportamento natural e frequente durante as interações humanas [Bentivoglio et al. 1997], no entanto, o seu potencial como um sinal de comando ou resposta em jogos digitais ainda é pouco explorado. A detecção de piscadas pode ser utilizada para fornecer novas formas de interação, adicionando um elemento desafiador e divertido aos jogos. Além disso, a detecção precisa de piscadas pode

ser aplicada em diversas áreas, como acessibilidade [Fan et al. 2020], detecção de stress [Merkies et al. 2019] e estudos psicológicos [Cho 2021].

Ao considerar a importância da detecção de piscadas na interação humano-computador, é possível explorar os benefícios dessa tecnologia em jogos digitais. Por exemplo, em jogos de realidade virtual, a detecção de piscadas pode ser utilizada para aprimorar a imersão, permitindo que os jogadores realizem ações sem a necessidade de usar controles físicos [Langbehn et al. 2018].

Este trabalho apresenta o "**Tente Não Piscar**", um jogo cujo objetivo é manter os olhos abertos evitando qualquer movimento de piscar antes do seu adversário durante uma partida. Para tal, ambos os participantes devem se posiciona em frente a uma câmera, a qual é utilizada para capturar as respectivas expressões faciais em tempo real. Como resultado, "Tente Não Piscar" apresenta regras simples, porém desafiadoras do ponto de vista tecnológico, as quais conseguem promover uma interação dinâmica e divertida entre os seus jogadores.

2. Fundamentação Teórica

2.1. TensorFlow e Detecção de Landmarks

O TensorFlow é uma plataforma de código aberto desenvolvida pela Google para aprendizado de máquina e inteligência artificial. Ele oferece uma ampla variedade de ferramentas e recursos que permitem aos desenvolvedores construir, treinar e implantar modelos de aprendizado de máquina de forma eficiente. O TensorFlow é amplamente utilizado em uma variedade de aplicações, desde reconhecimento de imagem e processamento de linguagem natural até previsões e análises avançadas de dados. Sua biblioteca de modelos pré-treinados acelera o desenvolvimento de soluções personalizadas.

A identificação de pontos de referência faciais (Face Landmarks) é crucial para localizar características faciais, como olhos, nariz, boca e contornos, em imagens. Isso fornece informações sobre a estrutura e expressões faciais. O processo usa algoritmos de aprendizado de máquina treinados em dados com *landmarks* pré-anotados. Durante o treinamento, os algoritmos aprendem padrões nas imagens correspondentes aos *landmarks* desejados. Modelos pré-treinados, como a biblioteca face-landmarks-detection do TensorFlow, simplificam a detecção de landmarks em tempo real sem a necessidade de treinamento completo do algoritmo.

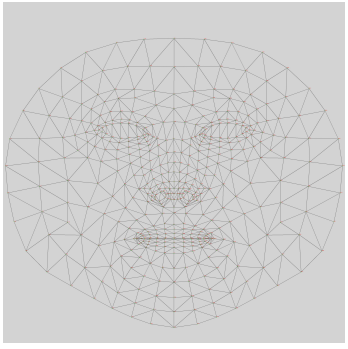

Figura 1. MediaPipe FaceMesh keypoints (Autor: TensorFlow).

A Figura 1 representa um mapa dos pontos-chave faciais que são capturados pela biblioteca de reconhecimento facial do TensorFlow.

2.2. Web Workers

Web Workers são cruciais para habilitar o processamento paralelo em JavaScript. O JavaScript é uma linguagem de programação de thread única, o que significa que todas as operações ocorrem em uma única thread, potencialmente causando atrasos em tarefas intensivas em processamento, como a detecção de landmarks faciais no jogo "Tente Não Pisca". Web Workers permitem a execução de código JavaScript em threads separadas, possibilitando processamento paralelo sem afetar a thread principal. No jogo, eles são usados para a detecção de landmarks faciais em uma thread separada, mantendo a interface do usuário responsiva. Além disso, permitem o compartilhamento eficiente de dados, possibilitando a detecção em tempo real. Em resumo, os Web Workers são essenciais para o processamento paralelo, garantindo uma detecção precisa e ágil de landmarks faciais no jogo.

3. Metodologia

O desenvolvimento do jogo "Tente Não Piscar" envolveu algumas etapas de integração das tecnologias mencionadas. Inicialmente, foram configurados o ambiente de desenvolvimento e os arquivos necessários para o jogo. Isso incluiu a criação de um projeto HTML básico, a importação das bibliotecas JavaScript necessárias e a definição da estrutura do jogo.

Para obter acesso à câmera do dispositivo, por exemplo, o jogo solicita permissão aos jogadores por meio da API do navegador. Isso é feito utilizando a função nativa do JavaScript chamada *getUserMedia*[1], que permite solicitar acesso à câmera e obter as imagens de vídeo em tempo real. Vale salientar que a câmera desempenha um papel essencial no jogo, pois é por meio dela que as expressões faciais são capturadas. Os dados de vídeo obtidos pela câmera são processados em tempo real utilizando o core do TensorFlow em Javascript e o face-landmarks-detection, um dos modelos disponibilizados pela própria biblioteca do TensorFlow. Essas tecnologias permitem a detecção de landmarks faciais, identificando pontos-chave na face dos jogadores, como olhos, nariz e boca.

A detecção de landmarks faciais é realizada com base em modelos pré-treinados, que são capazes de identificar com precisão as características faciais dos jogadores durante o jogo. Essa detecção é processada em uma thread separada, utilizando Web Workers, o que garante um desempenho adequado [Arias et al. 2021] sem interromper a jogabilidade. Neste sentido, para permitir a interação entre os jogadores, a detecção de piscadas de cada jogador é registrada e comparada. Cada frame é analisado pelo serviço de detecção, e a partir de um cálculo com um valor arbitrário, se determina qual dos jogadores teve a piscada detectada. Para o posicionamento dos jogadores, e aproveitando os pontos do rosto de cada jogador, foi escolhido arbitrariamente um ponto central no rosto de cada jogador, chamado "Nose Tip", para determinar qual jogador esta do lado esquerdo e do lado direito, e assim poder exibir em tela qual dos jogadores efetuou a piscada.

Com relação ao jogo produzido, a Figura 2 representa a estrutura da tela quando o programa está apto para ser jogado. A página contém uma sessão na parte superior

[1]https://developer.mozilla.org/en-US/docs/Web/API/MediaDevices/getUserMedia

contendo um botão help para exibição das regras do jogo, o nome do jogo e um botão play para iniciar o jogo, na Figura 2 indicada por 1, 2 e 3 respectivamente. Na parte central da página, indicada por 4 na Figura 2, contendo a maior parte do conteúdo é exibido a imagem em tempo real da câmera.

Figura 2. Tela do jogo após o carregamento das bibliotecas.

Ao decorrer do jogo pode aparecer um banner, exemplificado na Figura 3, informando sobre carregamento de bibliotecas, o cronômetro para iniciar a partida e o indicativo que alguém piscou e outros informativos.

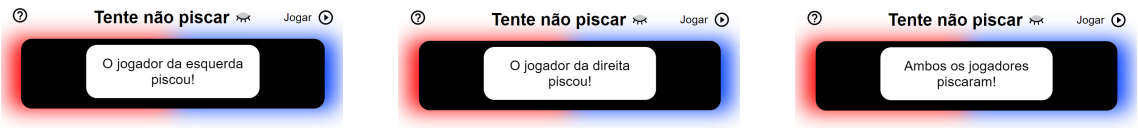

Figura 3. Respectivamente, a identificação de piscada do jogador da esquerda, direita e ambos os jogadores.

Com relação a configuração experimental do jogo proposto, esta envolveu a especificação das características do ambiente de teste, incluindo as configurações do computador, navegador e detalhes das câmeras utilizadas. O experimento foi conduzido em um notebook com processador Intel Core i7 de 7ª geração, 16 GB de memória RAM e placa de vídeo compatível com WebGL. O sistema operacional utilizado foi o Windows 10. Para a captura de vídeo, uma webcam integrada ao notebook foi utilizada, com resolução de 1280x720 e taxa de quadros de 28 fps, seguindo as configurações padrão. Durante os testes, foram realizadas verificações para garantir o correto funcionamento das câmeras e a estabilidade e fluidez na captura de vídeo.

4. Resultados Obtidos

Os testes do jogo foram realizados com 3 duplas de jogadores diferentes, Durante os testes do jogo[2], foram identificadas algumas limitações e desafios. A detecção de piscadas

[2]https://try-not-blink.vercel.app

apresentou imprecisões devido a variações na iluminação, ângulo da câmera e dificuldades do modelo de detecção de landmarks faciais. Uma das duplas não conseguiu ter a face detectada pelo programa, e não obtivemos um motivo claro do acontecimento.

Adicionalmente, realizamos testes do jogo em dois outros dispositivos: um smartphone Poco X3 GT e um notebook da linha Dell voltado para jogos. Durante esses testes, identificamos uma limitação significativa no cálculo da detecção de piscadas, que não estava sendo realizado de forma relativa à detecção dos pontos de referência faciais, e sim utilizando parâmetros constantes. Essa limitação afeta a capacidade de execução do jogo em diferentes dispositivos.

Apesar das limitações e desafios mencionados, durante os testes realizados em computadores com configurações específicas, obtivemos resultados positivos na precisão da detecção de piscadas no jogo. Isso sugere que, com ajustes e otimizações adequadas, o jogo pode funcionar com eficácia na detecção de piscadas, proporcionando uma experiência mais consistente para os jogadores em dispositivos compatíveis.

5. Conclusões e Trabalhos Futuros

Este artigo introduziu o desenvolvimento do jogo ”Tente Não Piscar” e ressaltou suas contribuições no campo da interação humano-computador, aplicando-as ao contexto de jogos digitais. Essas contribuições abrem perspectivas promissoras para o desenvolvimento de jogos com propósitos voltados não apenas para entretenimento, mas também para aplicações mais sérias relacionadas ao ato de piscar os olhos.

Apesar das limitações identificadas no *framework* tecnológico desenvolvido, o jogo oferece uma base sólida para futuras melhorias, criando oportunidades para avanços significativos na área de jogos digitais baseados em câmera e na detecção de comportamentos faciais, a exemplo do piscar dos olhos.

De fato, apesar dos desafios enfrentados, o jogo contribui significativamente para a interação humano-computador e jogos digitais, fazendo uso de tecnologias avançadas, como a detecção de *landmarks* faciais em tempo real com o auxílio do TensorFlow. Os resultados obtidos proporcionam *insights* valiosos para aprimoramentos futuros, incluindo melhorias no modelo de detecção, consideração da resolução da câmera e das posições globais das *landmarks*, estratégias para lidar com variações de iluminação e ângulo da câmera, bem como otimização do desempenho em diversas configurações de hardware. A implementação de atualizações periódicas e a busca por bibliotecas atualizadas podem ajudar a superar as limitações identificadas.

Nesse contexto, como trabalhos futuros, planeja-se explorar o potencial de aplicação dos recursos tecnológicos desenvolvidos em contextos mais amplos, como a interação com dispositivos de assistência pessoal, a criação de interfaces intuitivas para uma variedade de jogos e até mesmo aplicações médicas destinadas a pacientes acamados, demonstrando assim a versatilidade dessas tecnologias.

Referências

Arias, E., Arellano, M., and Cuadros, M. (2021). Optimization and improvement of bi-directional connections with web socket on synapse framework using web workers technology. In *2021 IEEE Sciences and Humanities International Research Conference (SHIRCON)*, pages 1–4.

Backe, H.-J. (2018). Metareferentiality through in-game images in immersive simulation game. In *Proceedings of the 13th International Conference on the Foundations of Digital Games*, FDG '18, New York, NY, USA. Association for Computing Machinery.

Bentivoglio, A. R., Bressman, S., Cassetta, E., Carretta, D., Tonali, P., and Albanese, A. (1997). Analysis of blink rate patterns in normal subjects. *Movement Disorders*, 12.

Cho, Y. (2021). Rethinking eye-blink: Assessing task difficulty through physiological representation of spontaneous blinking. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, CHI '21, New York, NY, USA. Association for Computing Machinery.

Fan, M., Li, Z., and Li, F. M. (2020). Eyelid gestures on mobile devices for people with motor impairments. In *Proceedings of the 22nd International ACM SIGACCESS Conference on Computers and Accessibility*, ASSETS '20, New York, NY, USA. Association for Computing Machinery.

Langbehn, E., Steinicke, F., Lappe, M., Welch, G. F., and Bruder, G. (2018). In the blink of an eye: Leveraging blink-induced suppression for imperceptible position and orientation redirection in virtual reality. *ACM Trans. Graph.*, 37(4).

Merkies, K., Ready, C., Farkas, L., and Hodder, A. (2019). Eye blink rates and eyelid twitches as a non-invasive measure of stress in the domestic horse. *Animals*, 9(8).