Deivison Cassimiro dos Santos

Reconhecimento de Gestos Manuais para Controle de Interfaces de Computador Usando Visão Computacional

Salvador
2026

Deivison Cassimiro dos Santos

Reconhecimento de Gestos Manuais para Controle de Interfaces de Computador Usando Visão Computacional

Trabalho de Conclusão de Curso apresentado ao Instituto Federal da Bahia como requisito parcial para obtenção do título de tecnólogo.

Instituto Federal da Bahia
Curso de Tecnologia em Análise e Desenvolvimento de Sistemas
Campus Salvador

Orientador: Domingos Mainart

Salvador
2026

Resumo

Este trabalho apresenta o desenvolvimento de um sistema de reconhecimento de gestos manuais para controle da interface de computador utilizando Visão Computacional e Aprendizado de Máquina. A solução proposta emprega uma webcam convencional para captura das imagens, a biblioteca MediaPipe Hands para detecção das mãos e extração dos landmarks, e o algoritmo Random Forest para classificação dos gestos. O processo de desenvolvimento compreendeu a coleta de imagens dos gestos, a construção de um dataset por meio da extração e normalização dos landmarks e o treinamento do modelo de classificação. Foram definidos seis gestos para controlar ações do sistema operacional: clique esquerdo, clique direito, movimentação do cursor, rolagem para cima, rolagem para baixo e estado de espera (standby). O modelo foi avaliado utilizando um dataset independente, por meio das métricas de acurácia, precisão, recall, F1-score e validação cruzada. Os resultados obtidos demonstraram desempenho superior a 99% de acurácia, evidenciando a capacidade do modelo em reconhecer corretamente os gestos propostos. Os resultados obtidos indicam que a solução proposta constitui uma alternativa viável para aplicações de interação humano-computador *touchless*, baseadas em reconhecimento de gestos em tempo real.

Palavras-chave: Reconhecimento de gestos, Visão Computacional, MediaPipe, Random Forest, Interação Humano-Computador.

Abstract

This work presents the development of a hand gesture recognition system for computer interface control using Computer Vision and Machine Learning. The proposed solution employs a conventional webcam for image acquisition, the MediaPipe Hands framework for hand detection and landmark extraction, and the Random Forest algorithm for gesture classification. The development process comprised gesture image acquisition, dataset construction through landmark extraction and normalization, and model training. Six gestures were defined to control operating system actions: left click, right click, mouse movement, scroll up, scroll down, and standby. The model was evaluated using an independent test dataset through accuracy, precision, recall, F1-score, and cross-validation metrics. The results achieved an accuracy greater than 99%, demonstrating the model's ability to correctly recognize the proposed gestures. The obtained results indicate that the proposed solution is a viable alternative for *touchless* human-computer interaction applications based on real-time hand gesture recognition.

Keywords: Hand Gesture Recognition, Computer Vision, MediaPipe, Random Forest, Human-Computer Interaction.

Sumário

1 Visão Geral

1.1 Declaração do Problema

A interação humano-computador tem evoluído rapidamente com o avanço das tecnologias de interfaces naturais, tornando os gestos manuais uma forma cada vez mais intuitiva e acessível de controle de dispositivos digitais. No entanto, o uso de gestos para substituir periféricos tradicionais, como mouse e teclado, em computadores pessoais ainda apresenta desafios significativos.

Como implementar um sistema preciso, eficiente e acessível de reconhecimento de gestos manuais em tempo real, utilizando apenas visão computacional, para complementar ou auxiliar o uso de dispositivos tradicionais em operações básicas de navegação computacional (movimentação do cursor, cliques, rolagem de páginas)?

Embora os gestos manuais sejam amplamente adotados em dispositivos móveis, consoles de videogame e aplicações de realidade aumentada, sua aplicação em ambientes de desktop enfrenta barreiras estruturais. As soluções comerciais existentes geralmente dependem de hardware especializado - como sensores de profundidade (Kinect, Leap Motion) ou câmeras proprietárias -, o que eleva o custo, reduz a acessibilidade e limita a adoção em larga escala. Adicionalmente, muitos sistemas comerciais exigem configurações complexas, calibração frequente e ambientes controlados, tornando-os impraticáveis para uso diário.

Essa limitação é especialmente crítica em cenários específicos: como em ambientes laboratoriais ou hospitalares que exigem esterilização e interações *touchless*; espaços públicos ou salas de aula onde o compartilhamento de periféricos é indesejável; e situações em que o usuário precisa manter as mãos livres para outras atividades. Nesses contextos, a ausência de uma solução de baixo custo, baseada em webcam convencional e bibliotecas de código aberto, representa uma lacuna importante na área de Interação Humano-Computador (IHC) e Visão Computacional.

O problema, portanto, não se restringe apenas à precisão técnica do reconhecimento, mas abrange também a eficiência computacional (baixa latência em tempo real), a acessibilidade econômica (sem necessidade de hardware adicional) e a robustez ambiental (funcionamento em diferentes condições de iluminação, fundo e presença de múltiplas pessoas). Superar essas barreiras é fundamental para democratizar o controle gestual e promover interfaces mais naturais, inclusivas e sustentáveis no cotidiano dos usuários de computadores.

Esta pesquisa busca preencher essa lacuna ao propor um protótipo funcional, modular e de fácil replicação, contribuindo tanto para o avanço científico na área quanto para aplicações práticas da tecnologia no cotidiano.

1.2 Objetivo

O objetivo geral deste trabalho é projetar, implementar e analisar um protótipo de interação gestual para controle básico de interface computacional, utilizando técnicas de visão computacional e uma webcam convencional como dispositivo de captura, de modo a possibilitar a execução de comandos essenciais de navegação, como movimentação do cursor, cliques e rolagem de páginas.

Para que o objetivo geral seja atingido, foram delineados os seguintes objetivos específicos:

- Implementar detecção e rastreamento preciso de mãos em tempo real utilizando MediaPipe [5] e OpenCV;
- Definir e implementar um conjunto mínimo de gestos intuitivos para controle básico de navegação;
- Desenvolver a integração entre os gestos reconhecidos e os comandos do sistema operacional;
- Avaliar o desempenho do sistema por meio de métricas como precisão, latência, taxa de falsos comandos e robustez da solução;
- Realizar testes exploratórios com usuários para obter feedback preliminar sobre a usabilidade e aceitação da solução.

1.3 Definições, Siglas e Abreviações

- **Visão Computacional**: Área da Ciência da Computação e Inteligência Artificial que permite aos computadores interpretar e compreender informações extraídas de imagens e vídeos.
- **IHC / HCI**: Interação Humano-Computador (Human-Computer Interaction), área dedicada ao estudo, projeto e avaliação de sistemas computacionais centrados no usuário.
- **NUI**: Natural User Interface, interfaces de usuário naturais baseadas em gestos, voz, toque ou movimentos corporais.
- **MediaPipe**: Framework do Google para construção de pipelines de percepção multimídia em tempo real, utilizado para detecção e rastreamento de mãos.
- **OpenCV**: Biblioteca de código aberto de visão computacional, amplamente utilizada para processamento de imagens e vídeos em tempo real.
- **PyAutoGUI**: Biblioteca Python para automação de interface gráfica do sistema operacional (controle de mouse e teclado).

- **Random Forest**: Algoritmo de aprendizado de máquina baseado em ensemble de árvores de decisão, utilizado para classificação dos gestos.
- **KNN**: K-Nearest Neighbors, algoritmo de aprendizado supervisionado baseado em instâncias e proximidade entre amostras.
- **HMM**: Hidden Markov Model (Modelo Oculto de Markov), modelo probabilístico utilizado para modelagem de sequências temporais.
- **landmarks**: Pontos de referência anatômicos (21 pontos tridimensionais) detectados pelo MediaPipe Hands que representam a estrutura da mão.

1.4 Proposta de Solução de Software

O sistema proposto consiste em um pipeline completo que captura imagens da webcam, detecta e rastreia as mãos, reconhece gestos pré-definidos e converte-os em ações do sistema operacional.

Utiliza apenas hardware comum (webcam) e bibliotecas de código aberto, garantindo baixo custo e alta acessibilidade. O fluxo principal consiste em:

- Captura contínua de frames da câmera.
- Pré-processamento da imagem.
- Detecção das mãos e extração dos landmarks utilizando MediaPipe Hands [5].
- Normalização dos landmarks extraídos.
- Classificação dos gestos por meio de um modelo de aprendizado de máquina treinado utilizando Random Forest.
- Execução da ação correspondente (mouse, clique, scroll etc.).

O treinamento do modelo é realizado previamente a partir de um conjunto de imagens coletadas para cada gesto. As imagens passam por um processo de extração dos landmarks e geração de dataset, permitindo que o classificador aprenda padrões associados às diferentes configurações da mão. Durante a execução em tempo real, o sistema utiliza o mesmo processo de extração de landmarks e envia os dados ao modelo treinado para determinar qual gesto está sendo realizado pelo usuário.

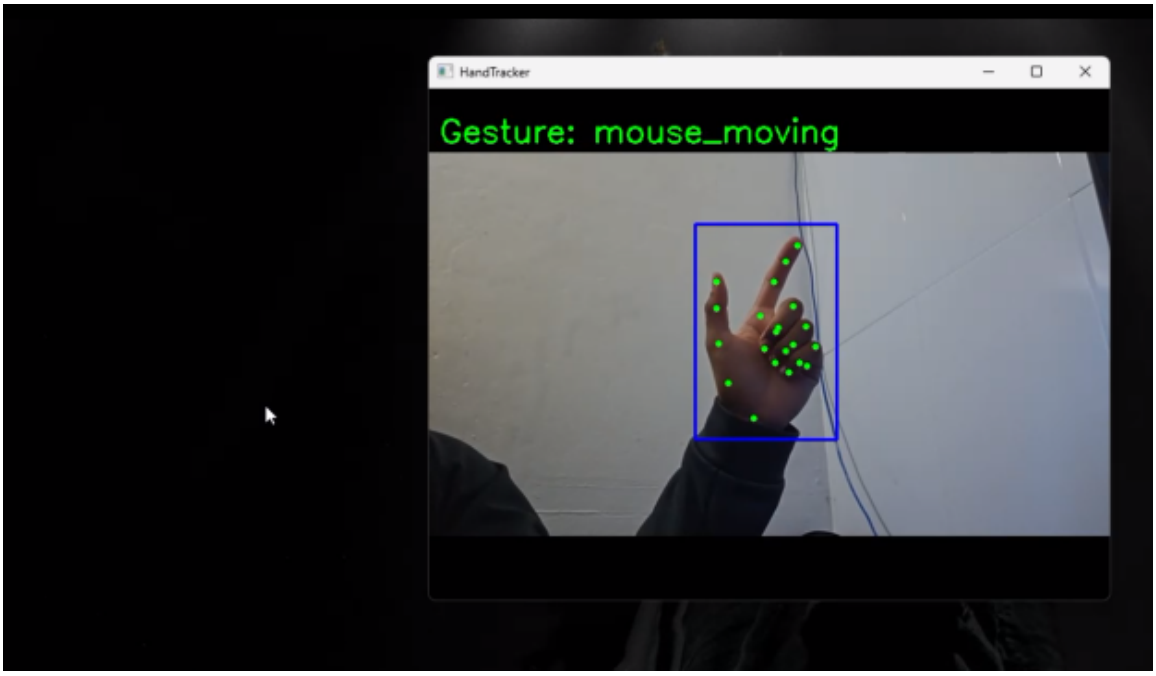

Figura 1 – Visão geral do sistema em execução, controlando o cursor com o gesto correspondente

2 Requisitos

2.1 Requisitos Funcionais

RF01 Como um usuário de computador, eu quero que o sistema capture continuamente o vídeo da webcam para que meus gestos sejam detectados em tempo real sem interrupções durante o uso da ferramenta.

RF02 Como um usuário de computador, eu quero que o sistema detecte e rastreie minha mão para identificar com precisão a posição e os movimentos realizados.

RF03 Como um usuário de computador, eu quero que o sistema reconheça gestos pré-definidos para executar ações básicas de navegação sem usar mouse ou teclado.

RF04 Como um usuário de computador, eu quero que os comandos correspondentes ao gesto sejam executados imediatamente no sistema operacional para ter uma experiência de navegação fluida e natural.

RF05 Como um usuário de computador, eu quero que o sistema ignore gestos com baixa confiança ou ausência de mão para evitar comandos acidentais em ambientes com múltiplas pessoas.

2.2 Requisitos Não-Funcionais

RNF01 Latência máxima de 100 ms por frame.

RNF02 Precisão mínima de 90% nas condições de operação previstas para o sistema.

RNF03 Funcionar com webcam padrão (720p ou superior).

RNF04 Interface leve e sem instalação de hardware extra.

3 Fundamentação Teórica

3.1 Interação Humano-Computador

A Interação Humano-Computador (IHC) é uma área dedicada ao estudo, projeto e avaliação de sistemas computacionais voltados para o uso humano. Seu objetivo principal consiste em desenvolver interfaces que permitam uma comunicação eficiente, intuitiva e acessível entre usuários e sistemas computacionais.

Com a evolução dos dispositivos digitais, surgiram novas formas de interação que buscam reduzir a dependência de dispositivos físicos tradicionais, como teclado e mouse. Nesse contexto, interfaces naturais ganharam destaque por possibilitarem a utilização de movimentos corporais, gestos, voz e expressões faciais como mecanismos de entrada. A Alexa, por exemplo, assistente virtual da Amazon, ganhou bastante destaque nos últimos anos. Recentemente, foram lançados os celulares da linha Huawei Mate, que também oferecem recursos de reconhecimento de gestos para realizar algumas atividades. Essas abordagens tornam a interação mais intuitiva e aproximam a comunicação homem-máquina das formas de interação presentes no cotidiano.

O reconhecimento de gestos manuais representa uma das aplicações mais relevantes das interfaces naturais. Por meio da interpretação de movimentos das mãos, torna-se possível executar comandos sem contato físico direto com dispositivos convencionais. Essa característica torna esses sistemas particularmente interessantes para aplicações relacionadas à acessibilidade, ambientes hospitalares, laboratórios, sistemas públicos e cenários onde a interação *touchless* é desejável.

Além da praticidade, alguns sistemas baseados em gestos visam contribuir para a inclusão digital de usuários que apresentam limitações motoras ou dificuldades na utilização de periféricos convencionais. Dessa forma, o desenvolvimento de soluções de reconhecimento gestual está diretamente alinhado aos objetivos da IHC de promover interfaces mais eficientes e centradas no usuário.

3.2 Visão Computacional

A Visão Computacional é uma área da Ciência da Computação que busca capacitar sistemas computacionais a interpretar e compreender informações extraídas de imagens e vídeos. Seu propósito é permitir que máquinas realizem tarefas tradicionalmente associadas à percepção visual humana, como identificação de objetos, rastreamento de movimentos e reconhecimento de padrões.

Os avanços recentes em processamento digital de imagens e aprendizado de máquina ampliaram significativamente as aplicações da visão computacional em diferentes áreas, incluindo se-

gurança, medicina, automação industrial, veículos autônomos e interação humano-computador.

No contexto do reconhecimento de gestos, a visão computacional é responsável por capturar informações visuais do usuário por meio de câmeras e transformá-las em representações computacionais capazes de serem processadas por algoritmos de classificação. Esse processo envolve etapas como aquisição da imagem, detecção da mão, extração de características relevantes e reconhecimento do gesto executado.

Uma das principais vantagens dos sistemas baseados em visão é a possibilidade de utilização de câmeras convencionais, reduzindo custos e eliminando a necessidade de dispositivos especializados. Essa característica favorece a democratização da tecnologia e amplia sua aplicabilidade em diferentes contextos.

3.3 Reconhecimento de Gestos

O reconhecimento de gestos consiste no processo de identificação e interpretação de movimentos realizados por um usuário, convertendo-os em comandos compreensíveis por um sistema computacional.

Segundo Mitra e Acharya (2007) [1], os gestos podem ser classificados em duas categorias principais: gestos estáticos e gestos dinâmicos. Gestos estáticos correspondem a configurações específicas das mãos ou dos dedos observadas em um único instante de tempo, enquanto gestos dinâmicos envolvem movimentos contínuos e dependem da análise temporal de múltiplos quadros de vídeo. O trabalho atual foca apenas no reconhecimento de gestos estáticos.

Os sistemas de reconhecimento gestual normalmente seguem um fluxo composto por aquisição dos dados visuais, segmentação da região de interesse, extração de características e classificação. O desempenho dessas etapas influencia diretamente a precisão do sistema final.

Embora os avanços recentes tenham aumentado significativamente a confiabilidade desses sistemas, diversos desafios ainda permanecem. Fatores como variações de iluminação, diferenças entre usuários, oclusões parciais das mãos, mudanças de escala e fundos complexos podem afetar a qualidade das informações extraídas e reduzir a taxa de reconhecimento.

Além disso, a transição entre diferentes gestos pode gerar ambiguidades durante a classificação, exigindo estratégias adicionais para reduzir falsos positivos e aumentar a robustez do sistema.

3.4 MediaPipe Hands

O MediaPipe Hands [5] é uma solução desenvolvida pelo Google para detecção e rastreamento de mãos em tempo real. A ferramenta utiliza técnicas de visão computacional e deep learning para localizar e acompanhar pontos de referência anatômicos das mãos com precisão.

O modelo é capaz de identificar até 21 pontos de referência tridimensionais, denominados landmarks, distribuídos ao longo da palma e dos dedos. Cada landmark é representado pelas

coordenadas espaciais x, y e z, que descrevem sua posição relativa na imagem capturada pela câmera.

A utilização desses pontos permite representar a configuração da mão de forma compacta e informativa, reduzindo significativamente a quantidade de dados necessários para o processo de classificação. Em vez de utilizar a imagem completa como entrada do modelo, torna-se possível trabalhar apenas com as informações geométricas extraídas dos landmarks.

Outra vantagem do MediaPipe Hands é sua capacidade de operar em tempo real utilizando hardware convencional. Essa característica o torna especialmente adequado para aplicações de interação humano-computador que exigem baixa latência e execução contínua.

No presente trabalho, os landmarks extraídos pelo MediaPipe constituem a principal fonte de dados utilizada para o treinamento e execução do modelo de reconhecimento gestual.

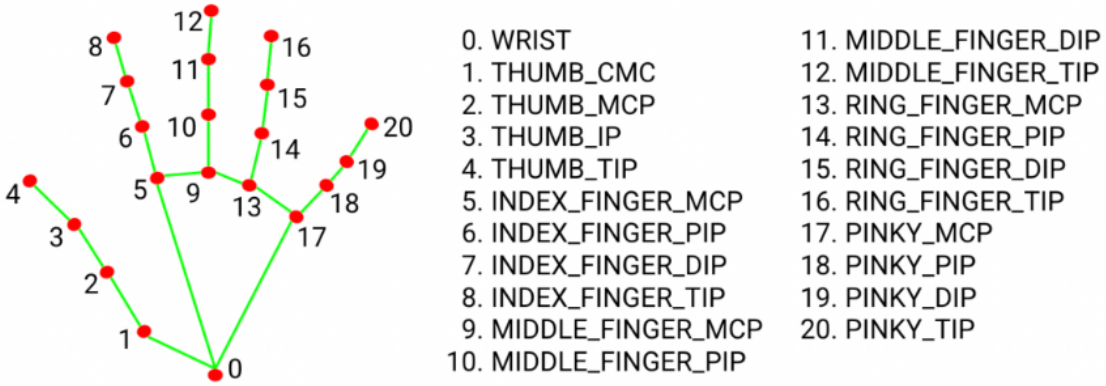

Figura 2 – Representação dos landmarks presente no site oficial do MediaPipe

3.5 Aprendizado de Máquina

O Aprendizado de Máquina é uma subárea da Inteligência Artificial dedicada ao desenvolvimento de algoritmos capazes de identificar padrões em dados e realizar previsões ou classificações sem a necessidade de programação explícita para cada situação.

De acordo com Mitchell (1997) [2], um sistema aprende a partir da experiência quando seu desempenho em uma determinada tarefa melhora progressivamente à medida que recebe novos exemplos.

Entre os paradigmas existentes, o aprendizado supervisionado é um dos mais utilizados em problemas de classificação. Nesse tipo de abordagem, o algoritmo é treinado utilizando exemplos previamente rotulados, permitindo que aprenda a associar padrões de entrada às respectivas categorias de saída.

No contexto deste trabalho, o conjunto de treinamento é composto pelos landmarks extraídos das imagens dos gestos coletados. Cada amostra é associada a uma das seis classes definidas para o sistema: Clique Esquerdo, Clique Direito, Mover Cursor, Scroll para Cima, Scroll para Baixo e Standby.

Após o treinamento, o modelo torna-se capaz de classificar novos gestos observados durante a execução em tempo real, permitindo a conversão dos movimentos do usuário em comandos computacionais.

4 Trabalhos Relacionados

O reconhecimento de gestos manuais tem sido amplamente estudado nas áreas de Visão Computacional e Interação Humano-Computador devido ao seu potencial para tornar a comunicação entre usuários e sistemas computacionais mais natural e intuitiva. Diversas pesquisas têm explorado diferentes técnicas para captura, processamento e interpretação dos movimentos das mãos, utilizando desde sensores especializados até abordagens baseadas exclusivamente em visão computacional.

Um dos trabalhos relevantes na área é apresentado por Macedo e Gomes (2021) [3], que propõem um sistema de interação computacional baseado na detecção das mãos utilizando uma webcam e o algoritmo Haar Cascade. O sistema desenvolvido permite controlar as teclas direcionais do computador por meio do posicionamento das mãos em regiões específicas da imagem capturada. Os autores obtiveram resultados satisfatórios, alcançando acurácia de 82%, recall de 80% e F1-Score de 81%, demonstrando a viabilidade da utilização de soluções de baixo custo para interação humano-computador.

Embora apresente bons resultados, a abordagem baseada em Haar Cascade possui limitações relacionadas à representação dos gestos. O método concentra-se principalmente na detecção da presença da mão e não na análise detalhada da configuração dos dedos e articulações, restringindo a quantidade de comandos distintos que podem ser reconhecidos.

Outra abordagem relevante é apresentada por Bentes (2024) [4] no trabalho denominado MusicalGestures. A proposta utiliza visão computacional e aprendizado de máquina para reconhecer gestos manuais em tempo real e convertê-los em notas musicais. O sistema emprega detecção das mãos e um modelo treinado para classificar diferentes gestos, permitindo a criação de uma interface musical baseada exclusivamente em movimentos das mãos. Os resultados demonstraram o potencial do reconhecimento gestual como mecanismo de interação natural, especialmente em aplicações criativas e artísticas.

Além dos trabalhos específicos, a literatura apresenta diversas técnicas para reconhecimento de gestos. Mitra e Acharya (2007) [1] classificam os métodos existentes em diferentes categorias, incluindo Máquinas de Estados Finitos, Modelos Ocultos de Markov, Redes Neurais Artificiais e outros algoritmos de aprendizado de máquina. Os autores destacam que sistemas de reconhecimento gestual enfrentam desafios relacionados à segmentação dos movimentos, variabilidade entre usuários, iluminação, oclusões e transições entre gestos.

Observa-se que muitas soluções comerciais e acadêmicas dependem de sensores especializados, câmeras de profundidade ou dispositivos dedicados para alcançar elevados níveis de precisão. Embora esses equipamentos proporcionem resultados satisfatórios, seu custo e complexidade de implantação podem limitar sua adoção em larga escala.

O presente trabalho, por sua vez, propõe um sistema capaz de reconhecer gestos manuais utilizando apenas uma webcam convencional, sem a necessidade de sensores adicionais. A

solução utiliza o MediaPipe Hands [5] para extração dos landmarks das mãos e um modelo Random Forest para classificação dos gestos. Essa combinação busca oferecer uma alternativa de baixo custo, acessível e compatível com computadores convencionais, mantendo desempenho adequado para aplicações de controle de interface em tempo real.

5 Metodologia

5.1 Visão Geral da Metodologia

A metodologia adotada neste trabalho consiste no desenvolvimento de um sistema de reconhecimento de gestos manuais baseado em visão computacional e aprendizado de máquina. O processo é composto por cinco etapas principais: coleta dos dados, aumento de dados, extração e normalização dos landmarks, treinamento do modelo de classificação e execução dos comandos em tempo real.

Inicialmente, são coletadas imagens representando os gestos previamente definidos. Em seguida, essas imagens passam por um processo de pré-processamento e aumento de dados com o objetivo de ampliar a diversidade das amostras utilizadas no treinamento. Posteriormente, os landmarks das mãos são extraídos e normalizados utilizando o MediaPipe Hands [5] e armazenados em um dataset estruturado. Por fim, os dados são utilizados para treinamento de um modelo Random Forest responsável pela classificação dos gestos durante a execução do sistema.

5.2 Definição dos Gestos

Para garantir clareza e viabilidade do projeto, foi definido um conjunto inicial de gestos básicos, cada um associado a um comando específico do sistema operacional.

Comando	**Gesto Manual**
Mover cursor	Dedo indicador levantado com polegar aberto
Clique esquerdo	Dedo indicador levantado com polegar fechado
Clique direito	Dedo indicador e mínimo levantado com polegar aberto
Standby	Dedos indicador e médio levantados juntos, com o polegar ao lado do indicador
Scroll para cima	Mão fechada com polegar para cima
Scroll para baixo	Mão fechada com polegar para baixo

Tabela 1 – Descrição dos gestos definida na documentação do projeto.

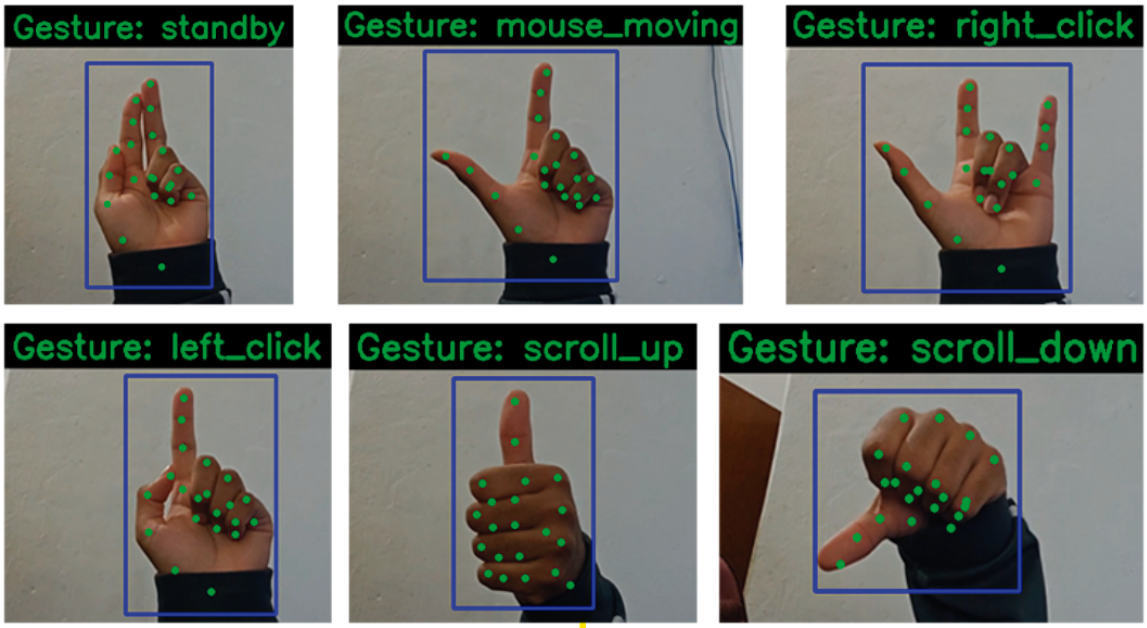

Figura 3 – Gestos retirados da aplicação em execução com debug ativo.

A inclusão da classe Standby tem como objetivo reduzir ativações involuntárias e minimizar a ocorrência de falsos positivos durante a utilização do sistema.

5.3 Coleta das Imagens

Para a construção do conjunto de treinamento, foram coletadas imagens contendo exemplos dos gestos definidos. Durante a coleta, buscou-se capturar diferentes variações de posicionamento da mão, incluindo mudanças de distância em relação à câmera e pequenas alterações de orientação.

Foram coletadas 450 imagens para cada gesto, e para cada imagem coletada foi realizado o espelhamento horizontal da mesma, visando evitar que o modelo aprenda a classificar apenas os gestos feitos por uma das mãos. No total, com o flip das imagens, obtivemos 900 imagens por gesto e um total de 5.400 imagens, somando todos os gestos.

A utilização de diferentes ângulos e distâncias busca aumentar a capacidade de generalização do modelo, tornando-o mais robusto diante das variações encontradas em cenários reais de utilização.

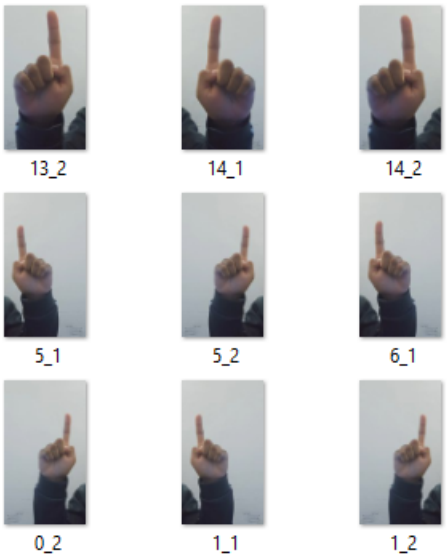

Figura 4 – Amostra dos gestos coletados (Clique Esquerdo)

5.4 Aumento de Dados

Após a coleta, as imagens passaram por um processo de aumento de dados.

As técnicas utilizadas incluem:

- Espelhamento horizontal;
- Rotação em diferentes ângulos.

O objetivo dessas transformações é gerar novas variações dos gestos sem a necessidade de realizar novas coletas, contribuindo para aumentar a diversidade do conjunto de treinamento e reduzir problemas de *overfitting* durante o treinamento do modelo.

Nesse processo, para cada imagem foram geradas 3 imagens adicionais, variando as características listadas anteriormente. Com isso, temos um total de 3.600 imagens por gesto, e um total de 21.600 imagens, somando todos os gestos.

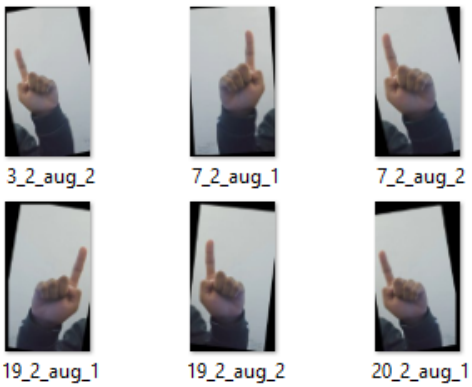

Figura 5 – Amostra dos gestos aumentados (Clique Esquerdo)

5.5 Extração dos Landmarks

A extração das características é realizada utilizando o MediaPipe Hands [5].

Para cada imagem processada, o MediaPipe identifica os principais pontos anatômicos da mão e retorna um conjunto de 21 landmarks tridimensionais. Cada landmark é representado por coordenadas espaciais que descrevem sua posição relativa na imagem.

Durante os testes, foi observado que a coordenada de profundidade, o eixo z, não era tão precisa e variava de forma considerável, mesmo com a mão relativamente estável na imagem. Dessa forma, o eixo z foi removido da construção do dataset.

Esses pontos fornecem uma representação compacta da configuração da mão, permitindo que o processo de classificação seja realizado utilizando informações geométricas em vez das imagens completas. Essa abordagem reduz significativamente a dimensionalidade dos dados e diminui o custo computacional do treinamento.

5.6 Normalização dos Landmarks e Construção do Dataset

Após a detecção da mão pelo MediaPipe Hands [5], os landmarks extraídos passam por uma etapa de normalização antes de serem armazenados no dataset. O objetivo desse procedimento é reduzir a influência de fatores externos, como distância da câmera, tamanho aparente da mão na imagem e pequenas variações de posicionamento, permitindo que o modelo de classificação aprenda padrões relacionados ao gesto executado e não às condições de captura.

O processo inicia-se com a identificação dos valores mínimos e máximos das coordenadas horizontais e verticais dos 21 landmarks detectados. Esses valores são utilizados para definir uma região delimitadora (bounding box) correspondente à área ocupada pela mão na imagem.

A partir dessa região, cada coordenada é convertida para uma escala relativa compreendida entre 0 e 1. Para cada landmark, as coordenadas normalizadas são calculadas utilizando a diferença entre a coordenada original e o valor mínimo observado, dividida pela amplitude correspondente do eixo analisado.

$$x_i^{\text{norm}} = \frac{x_i - x_{\min}}{x_{\max} - x_{\min}} \tag{5.1}$$

Dessa forma, a posição de cada ponto passa a ser representada relativamente ao espaço ocupado pela própria mão, independentemente de sua localização absoluta na imagem.

Essa estratégia apresenta uma importante vantagem para o problema abordado neste trabalho. Como os gestos podem ser executados a diferentes distâncias da câmera e em diferentes regiões do campo de visão, utilizar coordenadas absolutas poderia fazer com que um mesmo gesto fosse representado por valores significativamente distintos. Com a normalização, gestos equivalentes tendem a produzir representações mais consistentes, facilitando o aprendizado do modelo.

Ao final do processo, cada amostra do dataset é representada por um vetor numérico composto pelas coordenadas normalizadas dos 21 landmarks fornecidos pelo MediaPipe Hands e pelas

características auxiliares. Essas características adicionais são obtidas a partir do cálculo do ponto médio entre a base e a extremidade de cada dedo, fornecendo uma representação complementar da configuração da mão naquele gesto.

Considerando que ao final possuímos 21 landmarks, e cada landmark é composto pelas coordenadas (x e y), são gerados inicialmente 42 atributos. Em seguida, são adicionadas 10 features provenientes dos cinco pontos médios calculados (x e y), resultando em um vetor final contendo 52 features numéricas. Esse vetor é então associado à respectiva classe do gesto executado e armazenado no dataset para utilização durante o treinamento do modelo Random Forest.

5.7 Treinamento do Modelo

O treinamento é realizado utilizando o algoritmo Random Forest Classifier.

Durante essa etapa, o modelo recebe como entrada os vetores de features extraídos dos landmarks e aprende padrões associados a cada uma das classes definidas para o sistema.

O modelo foi treinado utilizando 80% do dataset, enquanto os 20% restantes foram reservados para a etapa de teste. Essa divisão permite avaliar a capacidade de generalização do modelo em relação ao dataset de treino, verificando seu desempenho na identificação de gestos que não foram utilizados durante o treinamento.

Ao final do processo, o modelo alcançou uma acurácia de 100% sobre os dados de treinamento e de 99,84% sobre o conjunto de teste obtido a partir da divisão treino/teste utilizada durante o treinamento. Esse resultado demonstra uma excelente capacidade de generalização dos dados de treinamento, evidenciando a importância da separação entre os conjuntos de treinamento e teste para uma avaliação confiável do desempenho do modelo. Posteriormente, foi realizada uma avaliação adicional utilizando um conjunto independente de imagens, apresentada no Capítulo 7, na qual foi obtida acurácia de 99,32%.

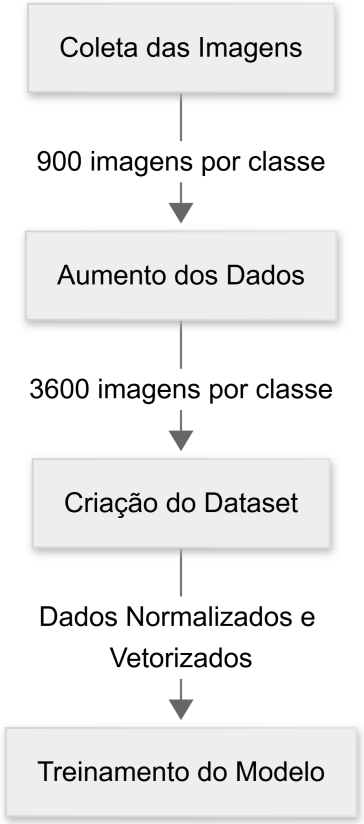

Figura 6 – Pipeline de Treinamento

5.8 Seleção do Algoritmo de Classificação

A escolha do algoritmo de aprendizado de máquina representa uma etapa fundamental no desenvolvimento de sistemas de reconhecimento de gestos. Diferentes algoritmos apresentam características distintas quanto à forma de aprendizado, complexidade computacional e capacidade de modelar padrões presentes nos dados. Considerando o objetivo deste trabalho, foram analisadas três abordagens: K-Nearest Neighbors (KNN), Hidden Markov Model (HMM) e Random Forest.

5.8.1 K-Nearest Neighbors (KNN)

O algoritmo K-Nearest Neighbors (KNN) é um método de aprendizado supervisionado baseado em instâncias. Diferentemente de algoritmos que constroem explicitamente um modelo durante o treinamento, o KNN armazena todas as amostras do conjunto de treinamento e realiza a classificação de novas instâncias comparando-as com seus vizinhos mais próximos.

O funcionamento do algoritmo consiste em calcular a distância entre a nova amostra e todas as amostras previamente armazenadas. Em seguida, são selecionados os k vizinhos mais próximos e a classe é definida por votação majoritária entre esses vizinhos.

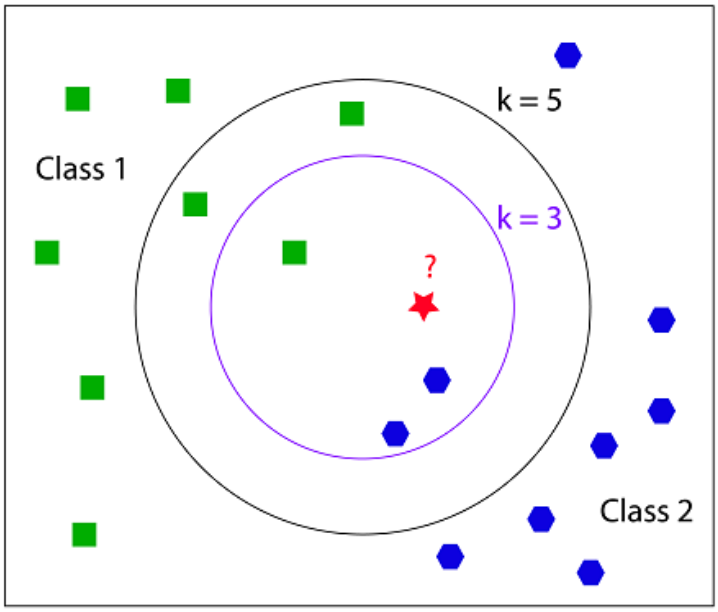

Figura 7 – Gráfico de exemplo do algoritmo KNN.

Fonte: Adaptado de [7].

Entre suas principais vantagens destacam-se a simplicidade de implementação, a facilidade de interpretação e o bom desempenho em problemas de baixa complexidade e reduzido número de atributos.

Entretanto, para aplicações em tempo real, como o reconhecimento contínuo de gestos, o KNN apresenta algumas limitações. Como todas as comparações são realizadas durante a inferência, o tempo de classificação cresce proporcionalmente ao tamanho do conjunto de treinamento. Além disso, seu desempenho pode ser significativamente afetado pela presença de ruído, pela escolha do valor de k e por diferenças de escala entre as características.

Considerando que o sistema proposto deve realizar classificações continuamente a cada quadro capturado pela webcam, o custo computacional da etapa de inferência torna o KNN uma alternativa menos adequada para este trabalho.

5.8.2 Hidden Markov Model (HMM)

Os Hidden Markov Models (HMM) constituem modelos probabilísticos capazes de representar sequências temporais por meio de estados ocultos e probabilidades de transição entre esses estados.

Segundo Mitra e Acharya (2007) [1], um processo pode ser descrito por um HMM quando sua evolução temporal satisfaz a propriedade de Markov, isto é, a probabilidade do estado atual depende apenas de um número limitado de estados anteriores.

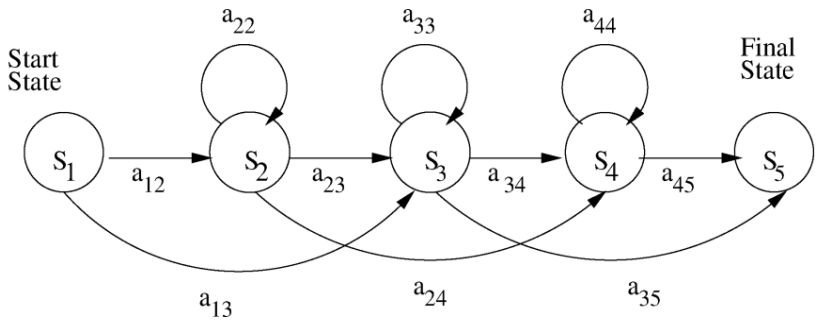

Figura 8 – Representação da transição entre estados do algoritmo HMM.

Fonte: Adaptado de [1].

Os HMMs são amplamente empregados em problemas cujo comportamento temporal é determinante para a classificação, como reconhecimento de fala, escrita manuscrita, análise de movimentos humanos e reconhecimento de gestos dinâmicos.

Durante a fase de planejamento deste trabalho, o HMM foi considerado uma alternativa promissora, uma vez que havia a intenção de incluir gestos dinâmicos, caracterizados por movimentos executados ao longo do tempo. Nesses casos, a ordem temporal das posições da mão constitui uma informação essencial para diferenciar gestos semelhantes, tornando os Modelos Ocultos de Markov particularmente adequados.

Entretanto, ao longo do desenvolvimento, optou-se por restringir o escopo do sistema ao reconhecimento de gestos estáticos. Nesse contexto, cada gesto pode ser representado por uma única configuração espacial da mão, não havendo necessidade de modelar relações temporais entre quadros consecutivos. Dessa forma, a principal vantagem dos HMMs deixou de ser relevante para o problema abordado, enquanto sua maior complexidade de implementação e treinamento passou a representar uma desvantagem em relação a outros algoritmos supervisionados.

5.8.3 Random Forest

O Random Forest é um algoritmo de aprendizado supervisionado baseado na combinação de múltiplas árvores de decisão. Durante o treinamento, cada árvore é construída utilizando subconjuntos aleatórios das amostras e dos atributos disponíveis, processo conhecido como *bootstrap aggregating (bagging)*.

Na etapa de classificação, cada árvore realiza uma predição de forma independente, sendo a classe final determinada por votação majoritária entre todas as árvores da floresta.

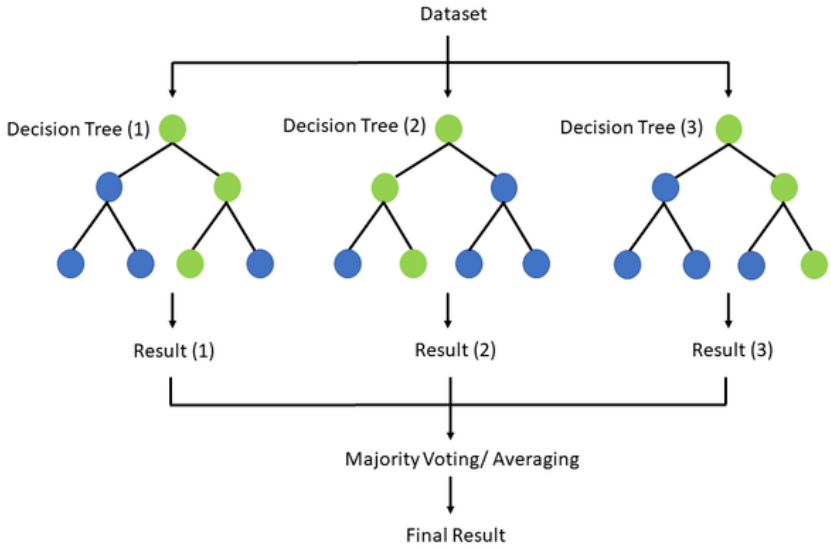

Figura 9 – Gráfico de exemplo do algoritmo Random Forest.

Fonte: Adaptado de [6].

Essa estratégia reduz significativamente o risco de *overfitting* presente em árvores de decisão individuais, além de aumentar a capacidade de generalização do modelo.

Outra característica importante do Random Forest é sua capacidade de lidar naturalmente com problemas multiclasse, atributos numéricos e conjuntos de dados de dimensão relativamente elevada, sem exigir procedimentos complexos de pré-processamento.

Além disso, o algoritmo apresenta baixa sensibilidade a ruídos presentes no conjunto de treinamento, elevada velocidade durante a inferência e permite estimar a importância relativa de cada atributo utilizado na classificação. Essa última característica representa uma vantagem adicional, pois possibilita analisar quais landmarks exercem maior influência na distinção entre os gestos.

Essas propriedades tornam o Random Forest particularmente adequado para aplicações em tempo real, nas quais milhares de classificações são realizadas continuamente durante a execução do sistema.

5.8.4 Justificativa da Escolha

Após a análise das alternativas, optou-se pela utilização do algoritmo Random Forest como modelo de classificação dos gestos.

Embora o KNN apresente simplicidade de implementação, seu custo computacional durante a inferência aumenta conforme o crescimento do conjunto de treinamento, uma vez que o algoritmo percorre todo o dataset a cada predição, característica pouco desejável em aplicações que exigem baixa latência, além da sua alta sensibilidade a ruídos.

Os Hidden Markov Models, por sua vez, mostraram-se adequados para cenários envolvendo gestos dinâmicos, nos quais a informação temporal exerce papel fundamental durante a classificação. Entretanto, como o escopo final deste trabalho foi direcionado exclusivamente ao

reconhecimento de gestos estáticos, a modelagem temporal oferecida pelos HMMs deixou de representar uma vantagem prática.

Por fim, com base na análise teórica realizada e nas características do problema abordado, o Random Forest foi selecionado por apresentar um equilíbrio entre simplicidade de implementação, capacidade de classificação multiclasse, robustez e potencial de execução em tempo real.

Assim, considerando o escopo e as restrições de desenvolvimento deste trabalho, o Random Forest foi considerado uma escolha adequada, por atender de forma satisfatória aos requisitos estabelecidos e oferecer um resultado consistente de desempenho, interpretabilidade e viabilidade de implementação.

Critério	**KNN**	**HMM**	**Random Forest**
Gestos estáticos	Bom	Regular	Excelente
Gestos dinâmicos	Regular	Excelente	Regular
Velocidade de inferência	Média	Alta	Alta
Facilidade de implementação	Alta	Baixa	Alta
Robustez a ruído	Baixa	Alta	Alta
Interpretabilidade	Alta	Baixa	Alta

Tabela 2 – Quadro comparativo entre os algoritmos avaliados

6 Desenvolvimento da Solução

6.1 Tecnologias da solução

- **Python 3.12**: Linguagem escolhida por sua sintaxe simples e legível, excelente ecossistema para visão computacional e automação, e suporte nativo a bibliotecas como OpenCV, MediaPipe e PyAutoGUI. Permite desenvolvimento rápido, prototipagem ágil e fácil integração entre módulos.
- **OpenCV**: Biblioteca open-source e uma das principais escolhas para processamento de imagens em tempo real. Justificada por oferecer captura eficiente de frames da webcam, conversão de cores (BGR e RGB) e manipulação de imagens com performance otimizada, essencial para o pipeline de captura contínua.
- **MediaPipe Hands (v0.10.14)**: Framework do Google otimizado para percepção em tempo real. Escolhido por fornecer detecção de mãos com alta precisão e extração de landmarks das mãos em tempo real, permitindo o treinamento do modelo para reconhecimento dos gestos, sem necessidade de hardware especializado (como sensores de profundidade), alinhando-se diretamente ao objetivo de acessibilidade e baixo custo.
- **PyAutoGUI**: Biblioteca Python para automação direta de mouse e teclado no sistema operacional. Selecionada por sua simplicidade e eficiência na conversão de gestos em ações reais, sem depender de APIs complexas do SO, garantindo compatibilidade ampla e latência mínima.
- **Webcam padrão (720p ou superior)**: Hardware de teste selecionado para garantir que o sistema funcione com equipamentos comuns do usuário final, sem exigir investimento extra, conforme a justificativa de acessibilidade do projeto.

6.2 Visão Arquitetural

A arquitetura do sistema foi projetada com o objetivo de ser modular, de fácil manutenção e otimizada para processamento em tempo real. Ela segue o padrão de pipeline de dados, no qual cada módulo tem uma responsabilidade bem definida, garantindo separação de responsabilidades e facilitando testes e futuras extensões.

O sistema é composto pelos seguintes módulos principais:

- Módulo principal
- Módulo de Rastreamento das Mãos

- Módulo de Reconhecimento dos Gestos
- Módulo de Execução dos Comandos

Essa arquitetura em módulos independentes permite que cada componente possa ser testado isoladamente, facilita a substituição de tecnologias no futuro (ex: trocar PyAutoGUI por outra biblioteca de automação) e mantém o código limpo e escalável.

A comunicação entre os módulos é realizada por meio de chamadas diretas de funções, garantindo o mínimo de overhead e atendendo ao requisito de baixa latência (< 100 ms por frame).

6.3 Módulo Principal

O módulo principal é responsável pela inicialização dos componentes do sistema e pelo controle do fluxo de execução da aplicação.

Durante a inicialização são carregados:

- A webcam;
- O módulo de rastreio das mãos;
- O módulo de reconhecimento de gestos;
- O módulo de execução de comandos.

Após a configuração inicial, o sistema entra em um loop contínuo de processamento. A cada frame capturado, o módulo principal coordena a detecção das mãos, a classificação do gesto e a execução do comando correspondente.

6.4 Módulo de Rastreamento das Mãos

O módulo de rastreamento das mãos encapsula a utilização do MediaPipe Hands [5].

Sua principal responsabilidade é processar os frames provenientes da webcam e identificar a presença de mãos na imagem. Para isso, cada frame é convertido para o formato RGB e submetido ao modelo de detecção do MediaPipe.

Para cada frame processado, os landmarks detectados passam pela mesma etapa de normalização dos dados citadas na seção 5.6, além de calcular a região delimitadora (bounding box) correspondente à mão detectada.

6.5 Módulo de Reconhecimento de Gestos

O módulo de reconhecimento dos gestos é responsável pelo carregamento e utilização do modelo Random Forest previamente treinado. Durante a inicialização, o módulo carrega o modelo serializado em arquivo utilizando a biblioteca *Pickle*.

O módulo recebe os dados já normalizados e vetorizados, e o Random Forest realiza a classificação e retorna uma das classes definidas pelo sistema.

Após a classificação realizada pelo modelo Random Forest, o módulo GestureModel também realiza uma validação baseada no nível de confiança da predição. Para cada classificação, o modelo fornece uma probabilidade associada à classe prevista, utilizada como medida de confiança da decisão. Foi estabelecido um limiar mínimo de 60% de confiança para que o gesto reconhecido seja considerado válido e tenha seu respectivo comando executado. Caso a confiança da classificação seja inferior a esse valor, a predição é desconsiderada e nenhuma ação é realizada no sistema operacional. Essa estratégia foi adotada com o objetivo de reduzir a ocorrência de falsos positivos e evitar que classificações de baixa confiabilidade resultem na execução involuntária de comandos.

6.6 Módulo de Execução de Comandos

O módulo de execução de comandos é responsável por converter os gestos classificados em ações efetivas sobre o sistema operacional.

Para facilitar a manutenção da aplicação, foi adotado um mecanismo de mapeamento entre classes de gestos e funções de execução. Dessa forma, cada gesto reconhecido é automaticamente associado ao método responsável pela ação correspondente.

Essa estratégia reduz o acoplamento entre os módulos e facilita a inclusão de novos comandos em futuras versões do sistema.

6.6.1 Controle do Cursor

Quando o gesto de movimentação do cursor é identificado, o sistema utiliza a posição do landmark correspondente à ponta do dedo indicador para controlar o cursor do mouse.

As coordenadas capturadas pela câmera são convertidas para a resolução da tela utilizando interpolação linear. Esse procedimento permite mapear adequadamente o espaço da webcam para o espaço disponível no monitor.

Além disso, foram implementados mecanismos adicionais para melhorar a experiência de utilização:

- Margem de segurança para impedir que o cursor ultrapasse os limites da tela;
- Fator de sensibilidade para permitir um melhor controle da velocidade do cursor;
- Suavização temporal para reduzir oscilações indesejadas.

6.6.2 Suavização dos Movimentos e Redução de Jitter

Um dos principais desafios de sistemas baseados em rastreamento das mãos é o fenômeno conhecido como *jitter*, caracterizado por pequenas oscilações involuntárias causadas por tremores naturais da mão ou pequenas imprecisões na detecção dos landmarks.

Para minimizar esse problema, foi implementada uma estratégia de suavização baseada na utilização da posição anterior do cursor. Em vez de mover o cursor diretamente para a posição desejada, o sistema calcula uma posição intermediária utilizando um fator de suavização configurável.

Esse procedimento reduz movimentos bruscos e proporciona uma navegação mais estável e confortável para o usuário.

6.6.3 Processamento Assíncrono dos Comandos

A movimentação do cursor é executada utilizando processamento assíncrono por meio de threads. Essa abordagem evita que operações relacionadas ao controle do cursor bloqueiem o processamento dos frames capturados pela webcam, contribuindo para manter a fluidez da interação em tempo real.

O uso de processamento concorrente reduz a latência percebida pelo usuário e melhora a responsividade geral do sistema.

6.6.4 Demonstração do Protótipo

Para melhor visualização do funcionamento do sistema em tempo real, foram gravados vídeos durante o desenvolvimento do sistema:

- **Vídeo 1 - Teste dos Gestos:** apresenta teste de todos os gestos do sistema. `https://youtu.be/PxB8VZglBjQ`
- **Vídeo 2 - Teste do Gesto de Scroll:** apresenta teste do gesto de scroll, passando por uma situação real de uso do sistema. `https://youtu.be/JclQ76i8T8U`

7 Avaliação do Modelo

7.1 Metodologia de Avaliação

A avaliação do modelo foi realizada utilizando um dataset de dados independentes daquele utilizado durante o treinamento, permitindo mensurar a capacidade de generalização do classificador frente a amostras não vistas anteriormente. Para a construção desse conjunto de teste, foram coletadas aproximadamente 1.800 imagens para cada um dos seis gestos definidos no sistema: Clique Esquerdo, Clique Direito, Mover Cursor, Scroll para cima, Scroll para baixo e Standby.

Entretanto, as imagens coletadas não são utilizadas diretamente pelo modelo. Inicialmente, cada imagem é processada pelo MediaPipe Hands [5], responsável por detectar a mão presente na cena e extrair os 21 landmarks que representam sua estrutura, passando pelo mesmo processo de normalização descrito previamente no treinamento do modelo. Apenas as imagens nas quais o MediaPipe conseguiu detectar corretamente a mão e retornar os landmarks foram incorporadas ao dataset de avaliação.

Como consequência, a quantidade final de amostras disponíveis para cada classe não é exatamente igual ao número de imagens originalmente coletadas. A variação ocorre devido às limitações no processo de detecção das mãos, uma vez que determinadas poses apresentam maior dificuldade para o MediaPipe Hands localizar a mão e retornar os landmarks da mesma.

No conjunto de avaliação gerado, foram obtidas 10.537 amostras distribuídas entre as seis classes, conforme apresentado na Tabela 3.

Classe	**Número de amostras**
Clique Esquerdo	1.800
Clique Direito	1.798
Mover cursor	1.789
Scroll para Baixo	1.635
Scroll para Cima	1.788
Standby	1.727
Total	**10.537**

Tabela 3 – Números de amostras por gesto

Observa-se que os gestos **Scroll para Baixo** e **Standby** apresentaram menor quantidade de amostras válidas, indicando que, durante a etapa de extração dos landmarks, o MediaPipe [5] encontrou maior dificuldade para detectar corretamente a mão nessas configurações. Esse comportamento evidencia que o desempenho global da solução depende não apenas do modelo de classificação baseado em Random Forest, mas também da qualidade da etapa anterior de detecção dos landmarks.

Após a geração do dataset de avaliação, foram calculadas as métricas de acurácia, precisão, recall e F1-score, tanto de forma global quanto individualmente para cada classe. Também foram analisadas a matriz de confusão, a importância das features utilizadas pelo modelo, a distribuição da confiança das predições e os resultados da validação cruzada com cinco partições (5-fold cross-validation), permitindo uma avaliação abrangente do desempenho do classificador.

Além das métricas de classificação, também foi avaliado o atendimento aos requisitos não funcionais definidos para o sistema. A Tabela 4 apresenta uma comparação entre os requisitos estabelecidos e os resultados obtidos durante os testes. Os resultados indicaram uma latência média de 61,88 ms por frame, com desvio padrão de 11,68 ms, correspondente a aproximadamente 16,16 quadros por segundo (FPS). Esses resultados demonstram que o sistema atende ao requisito não funcional RNF01, que estabelece uma latência máxima de 100 ms por frame, mantendo desempenho adequado para aplicações de interação em tempo real.

Requisito	**Meta**	**Resultado**	**Status**
RNF01 - Latência	$\leq$ 100 ms/frame	61,88 ms/frame	Atendido
RNF02 - Precisão	$\geq$ 90%	99,32%	Atendido

Tabela 4 – Verificação dos requisitos não funcionais avaliados

Observa-se que ambos os requisitos não funcionais definidos para o sistema foram atendidos. A precisão obtida foi superior à meta estabelecida de 90%, enquanto a latência permaneceu significativamente abaixo do limite máximo de 100 ms por frame, indicando que o sistema é capaz de executar o reconhecimento gestual em tempo compatível com aplicações interativas.

7.2 Avaliação Geral

A Tabela 5 apresenta os principais resultados obtidos durante a avaliação do modelo.

Métrica	**Valor**
Acurácia	99,32%
Precisão Macro	99,31%
Recall Macro	99,32%
F1-score Macro	99,32%
Taxa de Erro	0,68%
Número de classes	6
Número de amostras	10.537

Tabela 5 – Métricas gerais

Observa-se que todas as métricas apresentam valores superiores a 99%, indicando elevada capacidade de classificação dos gestos utilizados pelo sistema.

A proximidade entre precisão, recall e F1-score demonstra que o modelo apresenta comportamento equilibrado entre identificação correta dos gestos e redução de falsos positivos.

7.3 Avaliação por Classe

A Tabela 6 apresenta o desempenho individual para cada gesto reconhecido.

Classe	**Precisão**	**Recall**	**F1**
Clique Esquerdo	99,83%	99,67%	99,75%
Clique Direito	100%	99,61%	99,80%
Mover Cursor	99,55%	99,50%	99,52%
Scroll para Baixo	99,14%	99,14%	99,14%
Scroll para Cima	99,55%	97,99%	98,76%
Standby	97,79%	100%	98,88%

Tabela 6 – Métricas por classe

Observa-se que todas as classes apresentaram F1-score superior a 98%, evidenciando elevado desempenho em todo o conjunto de gestos.

O gesto **Clique Direito** obteve precisão de 100%, indicando que todas as amostras classificadas como clique Direito realmente pertenciam a essa classe. Esse resultado pode ter relação com o fato do gesto Clique Direito ser o mais distinto entre os gestos, com uma elevação do dedo mínimo que o difere dos outros gestos.

Por outro lado, o gesto **Scroll para Cima** apresentou o menor valor de recall (97,99%), indicando que parte das amostras dessa classe foi confundida com outros gestos durante a classificação.

Já a classe **Standby** apresentou recall de 100%, significando que todas as amostras pertencentes a essa classe foram corretamente identificadas pelo modelo. Esse resultado é particularmente importante, pois o gesto Standby representa o estado neutro do sistema e sua correta identificação reduz significativamente a ocorrência de comandos involuntários.

7.4 Matriz de Confusão

A Figura 10 apresenta a matriz de confusão obtida durante a avaliação.

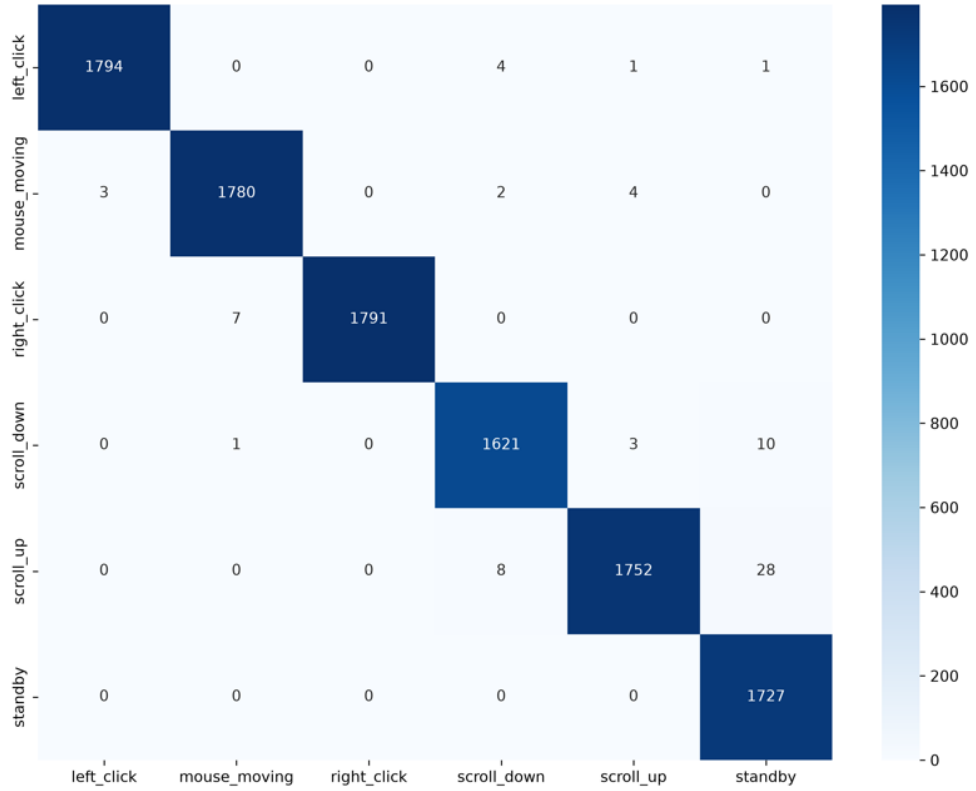

Figura 10 – Matriz de confusão

A matriz evidencia que a maior parte das classificações encontra-se concentrada na diagonal principal, indicando elevado número de classificações corretas.

As poucas classificações incorretas ocorreram principalmente entre os gestos de rolagem. Observa-se que:

- Scroll para Cima foi confundido com Standby em 28 ocasiões;
- Scroll para Baixo foi confundido com Standby em 10 ocasiões;
- Clique Direito foi confundido com Mover Cursor em apenas 7 amostras;
- Clique Esquerdo apresentou apenas 6 classificações incorretas.

A maior concentração de erros ocorreu na distinção entre os gestos de Scroll e Standby. Esse comportamento sugere que determinadas variações na orientação ou configuração da mão durante os gestos de Scroll podem produzir landmarks semelhantes à configuração utilizada para Standby. Como não foi realizada uma análise específica das amostras classificadas incorretamente, não é possível determinar com precisão a causa desse comportamento. Essa limitação representa uma oportunidade para investigação futura, incluindo ajustes na definição dos gestos, coleta de amostras adicionais e análise das características geométricas associadas aos erros.

Mesmo nesses casos, a quantidade de erros representa uma pequena fração do conjunto total de dados, indicando boa capacidade de generalização do modelo no dataset em que foi feita a análise.

7.5 Validação Cruzada

Com o objetivo de verificar a capacidade de generalização do modelo e reduzir possíveis efeitos de particionamentos específicos do dataset, foi empregada validação cruzada com cinco partições (5-Fold Cross Validation).

Os resultados obtidos são apresentados na Tabela 7.

Métrica	**Média**	**Desvio Padrão**
Acurácia	99,86%	0,11%
Precisão	99,86%	0,11%
Recall	99,86%	0,11%
F1-score	99,86%	0,11%

Tabela 7 – Métricas gerais da validação cruzada

Os resultados demonstram elevada estabilidade entre as diferentes partições do dataset.

A pequena variação observada entre os folds (desvio padrão de aproximadamente 0,11%) indica que o modelo mantém desempenho consistente mesmo quando treinado e avaliado sobre subconjuntos distintos dos dados. Essa estabilidade reduz a probabilidade de overfitting e reforça a robustez e a capacidade de generalização do classificador.

Entretanto, como os dados foram coletados pelo mesmo usuário dos dados de treinamento e em condições de captura semelhantes, os resultados não permitem afirmar de forma conclusiva a generalização do modelo para usuários e ambientes não representados no dataset.

7.6 Importância das Features

Uma vantagem do algoritmo Random Forest é a possibilidade de estimar a contribuição individual de cada atributo utilizado durante o treinamento.

A Figura 11 apresenta as vinte features mais relevantes identificadas pelo modelo.

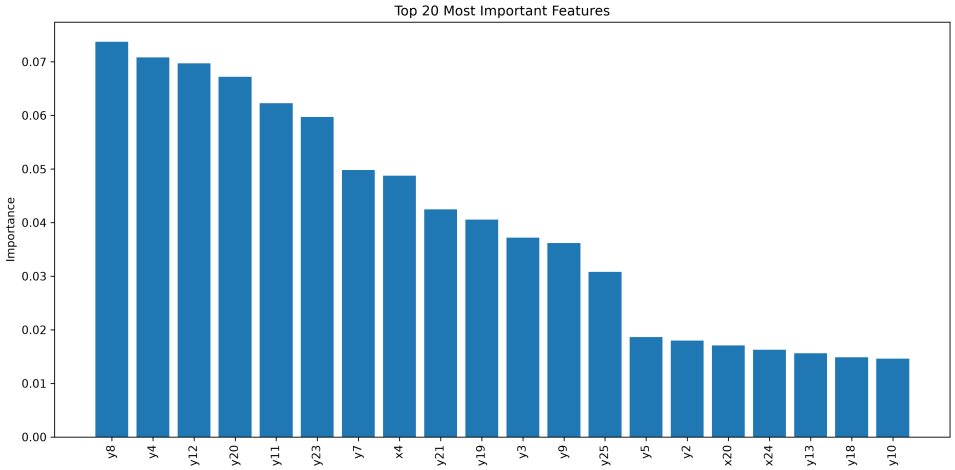

Figura 11 – As 20 features mais importantes

Observa-se que a maior parte das features mais importantes corresponde às coordenadas Y dos landmarks.

As features y8, y4, y12, y20 e y11 foram as que apresentaram maior contribuição para o processo de classificação.

Esse comportamento é coerente com o problema abordado, uma vez que os gestos escolhidos diferenciam-se principalmente pela posição vertical dos dedos durante sua execução.

Também é possível observar que algumas coordenadas x apresentam menor relevância, reforçando que a orientação vertical da mão exerce maior influência na distinção entre os gestos utilizados neste trabalho.

7.7 Distribuição da Confiança das Predições

Além da avaliação das métricas tradicionais, foi analisada a confiança das classificações produzidas pelo modelo. Os resultados obtidos foram:

Métrica	**Valor**
Confiança média	94,96%
Desvio padrão	10,37%
Confiança mínima	23%
Confiança máxima	100%

Tabela 8 – Confiança das predições

A confiança das predições também foi considerada durante o desenvolvimento do sistema, sendo estabelecido um limiar mínimo de 60% para a execução dos comandos. Dessa forma, embora o modelo possa produzir uma classificação com baixa confiança, essa predição não é encaminhada para a etapa de execução da ação correspondente. Esse mecanismo tem como

objetivo reduzir a possibilidade de que classificações incertas provoquem comandos involuntários no sistema operacional. É importante destacar que o limiar de 60% atua como uma regra de decisão da aplicação e não representa uma alteração na capacidade de classificação do modelo. Assim, as métricas apresentadas para avaliação do classificador representam o desempenho das predições realizadas pelo modelo, enquanto o limiar de confiança representa um mecanismo adicional de controle aplicado durante a execução do sistema. O valor de 60% foi definido empiricamente durante os testes de utilização da aplicação, considerando o equilíbrio entre a fluidez da interação e a redução de falsos positivos. Observou-se que os momentos de transição entre diferentes gestos apresentavam maior ocorrência de classificações incorretas e, consequentemente, maior risco de execução involuntária de comandos. Dessa forma, o limiar estabelecido buscou proporcionar maior segurança durante essas transições, sem comprometer significativamente a fluidez da interação com o sistema.

A Figura 12 apresenta a distribuição da confiança das predições.

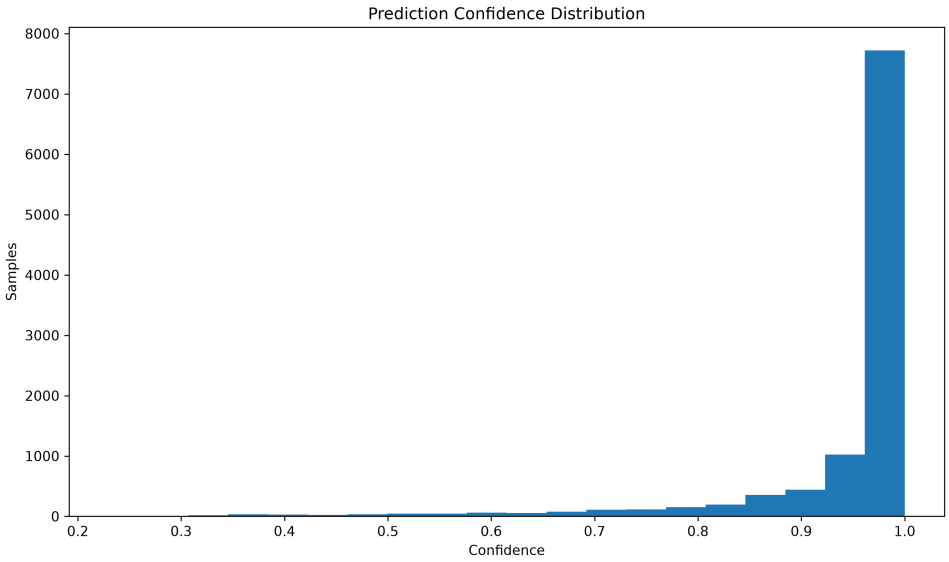

Figura 12 – Gráfico da confiança das predições

Observa-se que a maior parte das classificações concentra-se próxima ao valor máximo de confiança, com predominância de predições superiores a 95%.

As poucas classificações com baixa confiança correspondem principalmente aos casos próximos às fronteiras de decisão entre classes semelhantes, comportamento esperado em problemas de classificação multiclasse.

7.8 Discussão dos Resultados

Os resultados obtidos demonstram que o modelo Random Forest apresentou excelente desempenho para o reconhecimento dos seis gestos definidos neste trabalho.

A acurácia superior a 99%, aliada aos elevados valores de precisão, recall e F1-score, indica que o modelo foi capaz de aprender de forma consistente os padrões presentes no dataset.

A análise da matriz de confusão evidencia que os poucos erros concentram-se entre gestos semanticamente semelhantes, especialmente aqueles relacionados às ações de rolagem, enquanto gestos como Clique Esquerdo, Clique Direito e Mover Cursor apresentaram índices de acerto bastante elevados.

A validação cruzada reforça a robustez do modelo ao demonstrar desempenho consistente entre diferentes subconjuntos do dataset, reduzindo a possibilidade de que os resultados observados sejam decorrentes de uma divisão específica dos dados.

Por fim, a análise da importância das características mostra que o classificador baseou suas decisões principalmente nas coordenadas verticais dos landmarks, indicando que essas informações desempenham papel fundamental na diferenciação dos gestos utilizados. Em conjunto, esses resultados demonstram que a estratégia adotada para construção do dataset, normalização dos landmarks e treinamento do modelo foi adequada para o problema proposto.

7.9 Limitações da Solução

Embora os resultados obtidos sejam satisfatórios para o escopo proposto, algumas limitações permanecem presentes.

O desempenho do sistema depende diretamente da capacidade do MediaPipe [5] em detectar corretamente a mão do usuário. Situações envolvendo iluminação inadequada, oclusões parciais ou posicionamentos extremos podem dificultar a extração dos landmarks e comprometer a classificação dos gestos.

Além disso, a utilização contínua de gestos para controle do computador pode gerar fadiga física em sessões prolongadas de uso, característica comum em interfaces baseadas em movimentos corporais.

Por fim, o sistema foi desenvolvido para rastrear apenas uma mão por vez, restringindo a utilização de gestos que dependam da interação simultânea entre ambas as mãos.

7.10 Diagramas do Sistema

7.10.1 Diagrama de Sequência

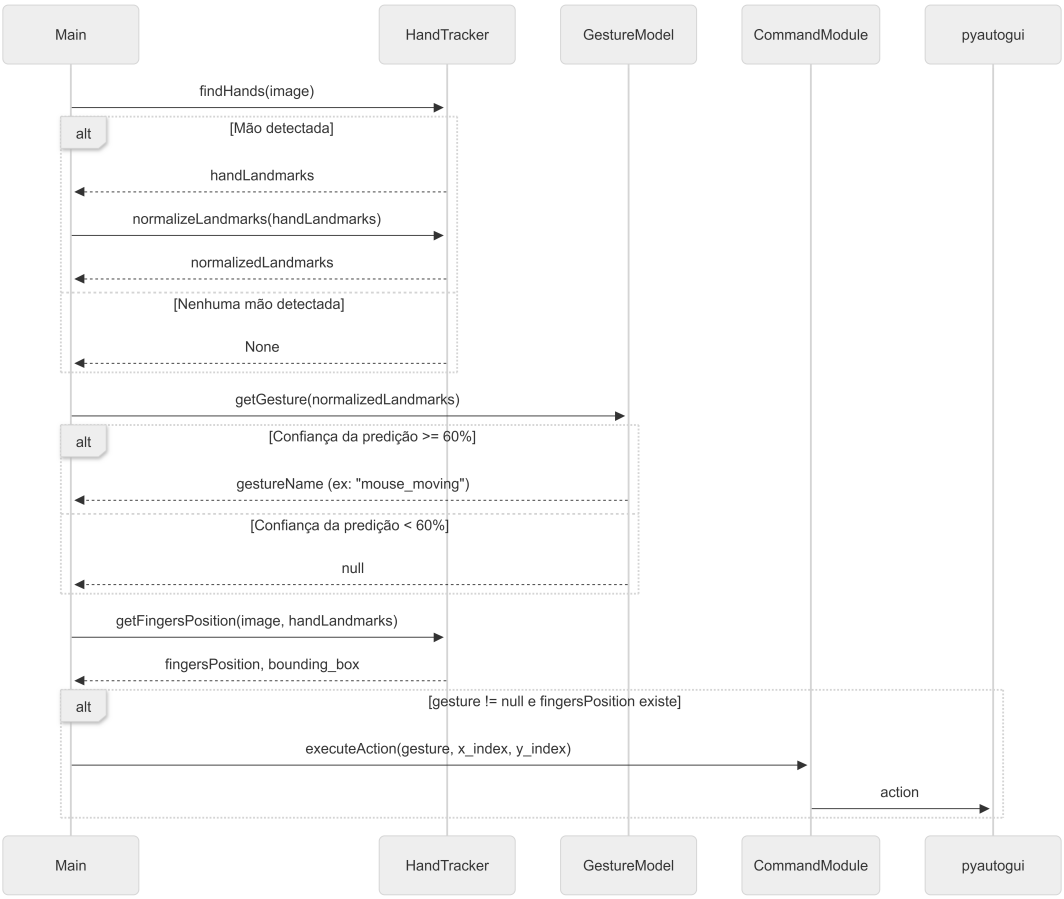

Figura 13 – Diagrama de Sequência

7.10.2 Diagrama de Fluxo de Dados

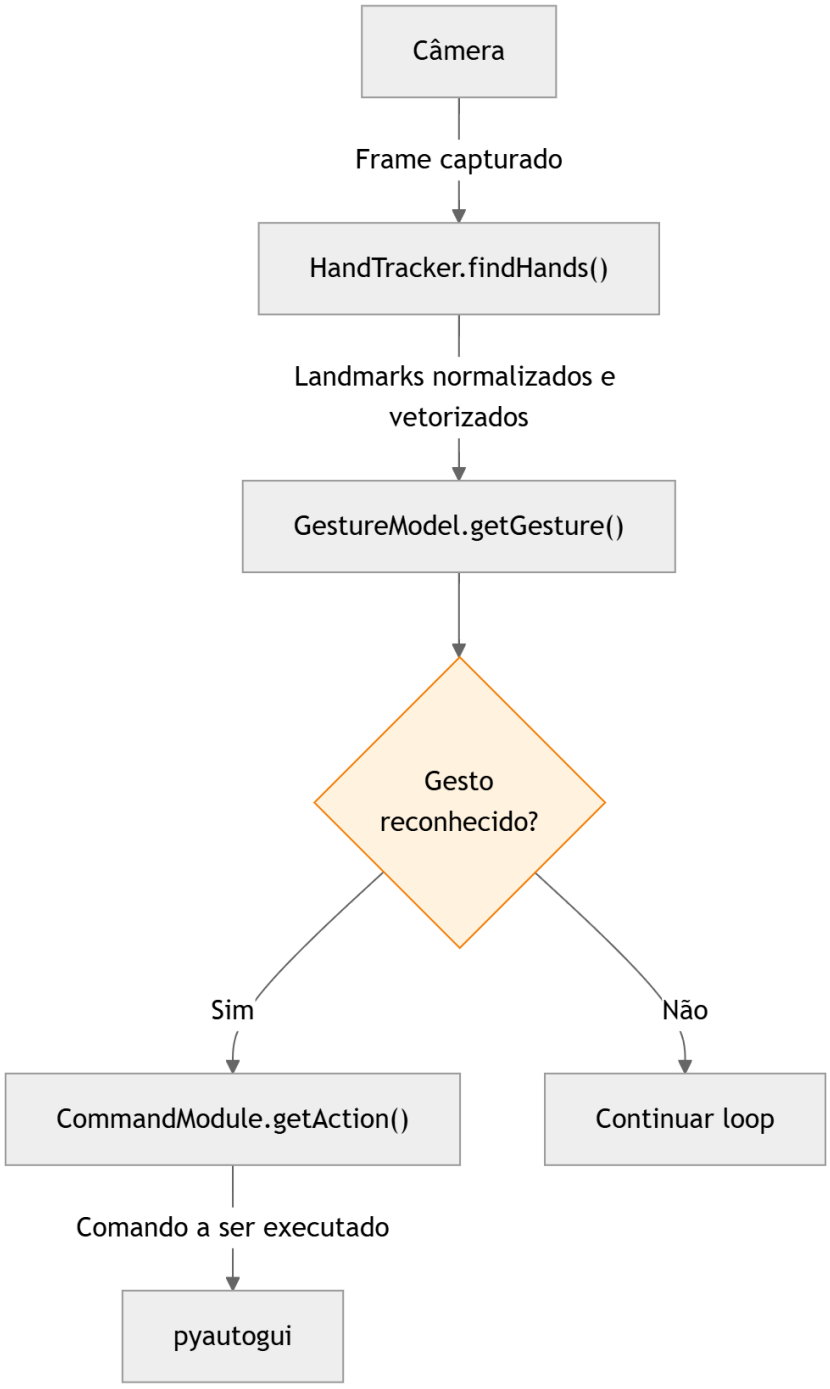

Figura 14 – Diagrama de Fluxo de Dados

8 Conclusão

8.1 Considerações Finais

Este trabalho teve como objetivo desenvolver um sistema de reconhecimento de gestos manuais para controle da interface de um computador utilizando apenas uma webcam convencional como método de captura e bibliotecas de código aberto, oferecendo uma alternativa de baixo custo para aplicações de interação humano-computador *touchless*. Para atingir esse objetivo, foi desenvolvido um pipeline completo composto pela captura das imagens, detecção das mãos utilizando o MediaPipe Hands [5], extração e normalização dos landmarks, classificação dos gestos por meio do algoritmo Random Forest e execução dos comandos correspondentes no sistema operacional.

Ao longo do desenvolvimento foram avaliadas diferentes estratégias para o reconhecimento dos gestos. Os experimentos realizados demonstraram que a utilização dos landmarks extraídos pelo MediaPipe como entrada para um modelo Random Forest apresentou um ótimo desempenho, mostrando-se uma boa opção entre as abordagens inicialmente consideradas, e evidenciando que a representação geométrica da mão é suficiente para diferenciar os gestos propostos, tornando o modelo menos sensível a fatores como fundo, iluminação e demais elementos presentes na imagem.

8.2 Resultados Obtidos

Os resultados experimentais demonstraram que a solução proposta alcançou elevado desempenho na tarefa de reconhecimento dos gestos. O modelo de classificação apresentou acurácia superior a 99%, além de elevados valores de precisão, recall e F1-score para todas as classes avaliadas. A validação cruzada indicou boa capacidade de generalização, enquanto a análise da matriz de confusão evidenciou reduzida ocorrência de erros de classificação.

Além da avaliação quantitativa, o sistema foi avaliado por cinco usuários voluntários, que tiveram acesso apenas à documentação dos gestos, a um vídeo demonstrando a aplicação em funcionamento e ao executável do sistema. Após os testes, os participantes relataram facilidade de aprendizado, boa precisão no reconhecimento dos gestos e rápida adaptação ao método de interação proposto. De modo geral, consideraram a solução intuitiva e funcional para tarefas cotidianas de controle do computador.

Durante os testes também foram identificadas oportunidades de melhoria, principalmente relacionadas à sensibilidade da movimentação do cursor, ao suporte para ambientes com múltiplos monitores e à inclusão de novos comandos, como duplo clique e seleção de texto. Embora o sistema já disponha de parâmetros configuráveis, como a sensibilidade horizontal e vertical

do cursor, a ausência de uma interface de configuração impede que o usuário personalize esse e outros parâmetros. Esses apontamentos representam importantes contribuições para a evolução da solução desenvolvida.

Os resultados experimentais demonstraram que o sistema atingiu os requisitos não funcionais inicialmente definidos. A precisão de classificação alcançou 99,32%, superando a meta mínima de 90%, enquanto a latência média foi de 61,88 ms por frame, permanecendo abaixo do limite de 100 ms estabelecido para operação em tempo real. Esses resultados evidenciam que a solução proposta apresenta desempenho adequado tanto sob o aspecto da qualidade da classificação quanto da responsividade durante a interação com o usuário.

8.3 Limitações do Trabalho

Apesar dos resultados obtidos, algumas limitações devem ser consideradas. O funcionamento do sistema depende diretamente da etapa de detecção das mãos realizada pelo MediaPipe Hands [5], podendo ocorrer falhas quando a mão está parcialmente ocluída, posicionada em ângulos extremos ou submetida a condições inadequadas de iluminação.

Os testes realizados indicam que o sistema foi capaz de funcionar com os usuários avaliados. Entretanto, como o conjunto de participantes foi reduzido e não houve uma avaliação controlada envolvendo diferentes perfis de usuários, não é possível afirmar que o desempenho será equivalente para qualquer pessoa.

A normalização dos landmarks reduz a influência da posição e escala da mão na imagem, enquanto o uso do MediaPipe reduz a dependência do classificador em relação à aparência visual da mão. No entanto, características geométricas individuais e diferentes formas de execução dos gestos podem influenciar a classificação, sendo necessária uma avaliação com maior diversidade de usuários para confirmar a capacidade de generalização.

Além disso, o sistema foi desenvolvido para reconhecer apenas gestos estáticos realizados com uma única mão, não contemplando gestos dinâmicos ou combinações entre ambas as mãos. Outra limitação refere-se ao número reduzido de participantes envolvidos na avaliação prática, o que restringe análises mais abrangentes sobre usabilidade em diferentes perfis de usuários.

8.4 Trabalhos Futuros

Como continuidade deste trabalho, pretende-se ampliar o conjunto de gestos reconhecidos pelo sistema, incorporando novos comandos e funcionalidades capazes de tornar a interação mais completa. Entre as possibilidades estão a implementação de ações como duplo clique, seleção de texto, arrastar e soltar objetos e atalhos personalizados.

Outra possibilidade consiste na inclusão de gestos dinâmicos, permitindo reconhecer movimentos realizados ao longo do tempo. Para esse cenário, poderão ser investigadas abordagens

baseadas em modelos capazes de explorar informações temporais, ampliando as possibilidades de interação oferecidas pelo sistema.

Também se pretende realizar estudos de usabilidade com um número maior e mais diversificado de participantes, além de investigar técnicas de suavização mais avançadas para o controle do cursor e estratégias que aumentem a robustez da solução em diferentes condições ambientais.

Por fim, conclui-se que os objetivos propostos foram alcançados, demonstrando que a combinação entre Visão Computacional, MediaPipe Hands [5] e Random Forest constitui uma solução tecnicamente viável para o desenvolvimento de interfaces naturais de baixo custo, contribuindo para a disseminação de aplicações de interação humano-computador baseadas em reconhecimento de gestos em tempo real.

Referências

[1] MITRA, S.; ACHARYA, T. Gesture Recognition: A Survey. IEEE Transactions on Systems, Man, and Cybernetics – Part C: Applications and Reviews, v. 37, n. 3, p. 311–324, 2007.

[2] MITCHELL, T. M. Machine Learning. New York: McGraw-Hill, 1997.

[3] MACEDO, Anne Livia da F.; GOMES, Igor Ruiz. Sistema de interação computacional baseado na detecção das mãos utilizando classificadores em cascata. Universidade Federal do Pará, 2021.

[4] BENTES, Yan Wiverson Lavor. MusicalGestures: Reconhecimento Gestual utilizando Visão Computacional e Inteligência Artificial aplicado à Composição Musical. Trabalho de Conclusão de Curso (Graduação em Engenharia de Computação) – Universidade Federal de Santa Catarina, Araranguá, 2024.

[5] GOOGLE. MediaPipe Hands. Disponível em: `https://ai.google.dev/edge/mediapipe`. Acesso em: 04 jun. 2026.

[6] INTERACTIVE CHAOS. Random Forest. Disponível em: `https://interactivechaos.com/en/wiki/random-forest`. Acesso em: 28 jun. 2026.

[7] MACHINE LEARNING TUTORIAL. K-Nearest Neighbors (KNN). Disponível em: `https://machine-learning-tutorial-abi.readthedocs.io/en/latest/content/supervised/knn.html`. Acesso em: 28 jun. 2026.