UNIVERSIDADE DO ESTADO DE SANTA CATARINA – UDESC
CENTRO DE CIÊNCIAS TECNOLÓGICAS - CCT
BACHARELADO EM ENGENHARIA ELÉTRICA

DEBORAH CRISTINA MAIA

DESENVOLVIMENTO DE INTERFACE GESTUAL PARA CONTROLE DA LOCOMOÇÃO DE ROBÔS MÓVEIS

JOINVILLE
2024

DEBORAH CRISTINA MAIA

DESENVOLVIMENTO DE INTERFACE GESTUAL PARA CONTROLE DA LOCOMOÇÃO DE ROBÔS MÓVEIS

Trabalho de Conclusão de Curso apresentado ao Departamento de Engenharia Elétrica do Centro de Ciências Tecnológicas, da Universidade do Estado de Santa Catarina, como requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. Douglas Wildgrube Bertol

JOINVILLE
2024

Para gerar a ficha catalográfica de teses e
dissertações acessar o link:
https://www.udesc.br/bu/manuais/ficha

Maia, Deborah Cristina
Desenvolvimento de interface gestual para controle da locomoção de robôs móveis / Deborah Cristina Maia. – Joinville, 2024.
38 p. : il.

Orientador: Prof. Dr. Douglas Wildgrube Bertol.
Trabalho de Conclusão de Curso – Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Dertamento de Engenharia Elétrica, Joinville, 2024.

1.Interface humano-robô. 2.Robótica móvel. 3.Visão computacional. 4.Aprendizagem de máquina. 5.Reconhecimento facial. I. , . II. , . III. Universidade do Estado de Santa Catarina, Centro de Ciências Tecnológicas, Dertamento de Engenharia Elétrica. IV. Título.

DEBORAH CRISTINA MAIA

DESENVOLVIMENTO DE INTERFACE GESTUAL PARA CONTROLE DA LOCOMOÇÃO DE ROBÔS MÓVEIS

Trabalho de Conclusão de Curso apresentado ao Departamento de Engenharia Elétrica do Centro de Ciências Tecnológicas, da Universidade do Estado de Santa Catarina, como requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. Douglas Wildgrube Bertol

BANCA EXAMINADORA:

Orientador:

Prof. Dr. Douglas Wildgrube Bertol
Universidade do Estado de Santa Catarina

Membros:

Prof. Me. Aureo Guilherme Dobrikopf
Universidade do Estado de Santa Catarina

Prof. Me. Gabriel Abatti
Universidade do Estado de Santa Catarina

Joinville, 02 de dezembro de 2024

Aos estudantes da Universidade do Estado de Santa Catarina, pela inspiração de sempre!

AGRADECIMENTOS

Agradeço primeiramente a Santíssima Trindade por ter me criado, me redimido e me dado a inteligência para aprender. A minha mãe Ana Cristina de Oliveira, a minha irmã Darah Cristina Maia, ao meu primo Darlan de Oliveira e ao meu cunhado Guilherme Luis Adams de Matos que sempre estiveram ao meu lado me apoiando ao longo de toda a minha trajetória na universidade. A todos os meus professores do curso de Engenharia Elétrica da Universidade do Estado de Santa Catarina – Udesc pela excelência da qualidade técnica de cada um. Deixo um agradecimento especial ao meu orientador pelo incentivo e pela dedicação do seu escasso tempo ao meu projeto de pesquisa. Como disse Santa Catarina de Sena: A providência divina jamais falta ao homem em nada, sob a condição de que ele a aceite.

"Porque o SENHOR dá a sabedoria, e da sua boca vem o conhecimento e a inteligência."
(Provérbios 2:6)

RESUMO

Este trabalho apresenta o desenvolvimento de uma interface humano-robô para controlar robôs móveis por movimentos das mãos. A pesquisa se insere na robótica interativa e busca solucionar a dificuldade de operadores não especializados em usar interfaces complexas, que priorizam aspectos técnicos em detrimento da usabilidade. Foi identificada a falta de interfaces intuitivas, levando à hipótese de que tecnologias como visão computacional e aprendizado de máquina podem viabilizar sistemas mais acessíveis. Assim foi proposto uma interface capaz de traduzir gestos em comandos, eliminando a necessidade de conhecimentos técnicos. A metodologia incluiu o uso de bibliotecas para captura e processamento de imagens e algoritmos de aprendizado supervisionado para reconhecer gestos. Os resultados confirmaram a viabilidade da solução, permitindo o controle fluido do robô por gestos. Conclui-se que a abordagem adotada contribui para democratizar o uso de sistemas robóticos.

Palavras-chave: Interface humano-robô. Robótica móvel. Visão computacional. Aprendizagem de máquina. Reconhecimento facial.

ABSTRACT

This work presents the development of a human-robot interface to control mobile robots through hand movements. The research falls within the field of interactive robotics and seeks to address the difficulty non-specialist operators face when using complex interfaces that prioritize technical aspects over usability. A lack of intuitive interfaces was identified, leading to the hypothesis that technologies such as computer vision and machine learning can enable more accessible systems. Thus, an interface capable of translating gestures into commands was proposed, eliminating the need for technical expertise. The methodology included the use of libraries for image capture and processing, as well as supervised learning algorithms to recognize gestures. The results confirmed the feasibility of the solution, allowing fluid robot control through gestures. It is concluded that the adopted approach contributes to democratizing the use of robotic systems.

Keywords: Human-robot interface. Mobile robotics. Computer vision. Machine learning. Facial recognition.

LISTA DE ILUSTRAÇÕES

Figura 1 – LEGO *Mindstorms* utilizado para teste. 18
Figura 2 – Ponto de referência da mão para o modelo. 24
Figura 3 – Implementação do identificador de mãos. 25
Figura 4 – Fluxograma de funcionamento do programa de reconhecimento facial. . . . 28
Figura 5 – Reconhecedor facial. 30
Figura 6 – Fluxograma de funcionamento da interface. 31
Figura 7 – Marcadores do centro das mãos e face. 31
Figura 8 – Mão esquerda para a direita. 33
Figura 9 – Mão esquerda para a lateral esquerda. 33
Figura 10 – Mão direita para baixo. 34
Figura 11 – Mão direita para cima. 34

LISTA DE ABREVIATURAS E SIGLAS

AI	*Artificial Intelligence*
AR	*Augmented Reality*
AMR	*Autonomous Mobile Robot*
BGR	*Blue, Green, Red*
CLP	Controlador Lógico Programável
DL	*Deep Learning*
FPS	*Frames Per Second*
ID	*Identificação*
HMM	*Hidden Markov Model*
IoT	*Internet of Things*
MIT	*Massachusetts Institute of Technology*
NUI	*Natural User Interface*
OpenCV	*Open Source Computer Vision Library*
ROS	*Robot Operating System*
RGB	*Red, Green, Blue*
VR	*Virtual Reality*

SUMÁRIO

1	**INTRODUÇÃO**	**12**
1.1	OBJETIVO PRINCIPAL	**14**
1.2	OBJETIVOS ESPECÍFICOS	**14**
1.3	ORGANIZAÇÃO DO TRABALHO	**14**
2	**REVISÃO BIBLIOGRÁFICA**	**15**
2.1	SISTEMAS ROBÔTICOS	**15**
2.2	INTERFACE DE CONTROLE ROBÓTICO	**18**
2.3	CONCLUSÃO DO CAPITULO	**21**
3	**VISÃO COMPUTACIONAL**	**23**
3.1	CAPTURA E TRATAMENTO DE IMAGENS	**23**
3.2	APRENDIZAGEM DE MAQUINA E RECONHECIMENTO FACIAL	**25**
4	**DESENVOLVIMENTO**	**29**
4.1	INTERFACE HUMANO-ROBÔ	**29**
4.2	RESULTADO	**32**
5	**CONCLUSÃO**	**35**
	REFERÊNCIAS	**36**

1 INTRODUÇÃO

Os robôs são máquinas autônomas ou semiautônomas que desempenham um papel crucial em várias áreas, adaptando-se às demandas específicas de cada setor. Eles são amplamente utilizados em fábricas e indústrias, onde assumem funções repetitivas e sistemáticas, frequentemente prejudiciais à saúde humana. Os progressos na robótica têm sido notáveis, transformando a maneira como realizamos diversas tarefas e aumentando a eficiência em muitos processos. Esses avanços abrangem desde a implementação de sensores autônomos e sistemas de navegação até o desenvolvimento da inteligência artificial. Essa evolução rápida ampliou o uso dos robôs, incluindo os móveis, empregados em diferentes aplicações, como sondas espaciais, drones de entrega e robôs utilizados na agricultura, que automatizam processos práticos e de proteção em grandes projetos. Também existem robôs especializados em transporte e distribuição de produtos em centros logísticos e armazéns, melhorando a logística e diminuindo a necessidade de intervenção humana em ambientes hostis.

No ambiente doméstico, surgiram aspiradores automáticos e sistemas de limpeza inteligentes, evidenciando a crescente integração da robótica em espaços cada vez mais dinâmicos. No contexto educacional, a aplicação dos robôs móveis LEGO *Mindstorms*, que são amplamente adotados em instituições de ensino para introduzir os estudantes a práticas de programação e controle robótico antes de seus estudos formais (Universal Robots, 2024). Esse tipo de tecnologia oferece aos alunos a chance de adquirir experiência prática ao realizar projetos, promovendo o aprendizado de lógica de programação, algoritmos e conceitos de eletrônica de maneira interativa e cativante.

Ademais, os robôs LEGOs *Mindstorms* servem como uma porta de entrada acessível a sistemas mais complexos, estimulando o interesse dos alunos em campos como robótica, controle e programação. Com o passar dos anos, o uso dessa tecnologia tem se expandido, proporcionando aos alunos a oportunidade de resolver desafios do mundo real e desenvolver habilidades valiosas para suas futuras carreiras.

Considerando o avanço dos robôs, novos desenvolvimentos inerentes ao sistema são as novas interfaces de controle robótico. Geralmente, estas são desenvolvidas pelos mesmos fabricantes. Estas interfaces são projetadas para garantir que o operador possa controlar e interagir com o robô de forma eficiente, segura. Dependendo do tipo de robô e das necessidades específicas da aplicação, a interface pode variar significativamente.

Essas interfaces podem incluir painéis de controle físico, telas sensíveis ao toque, sistemas de controle de voz ou até mesmo interfaces gráficas sofisticadas que permitem a programação e monitoramento do robô em tempo real. Com os avanços da tecnologia, muitas dessas interfaces estão cada vez mais integradas aos sistemas de software de computador, proporcionando funções como simulação, visualização 3D, controle remoto em rede e integração com outros sistemas de automação. Além disso, as interfaces de controle robótico também devem ser projetadas tendo em mente os aspectos de segurança, especialmente em ambientes industriais ou de alta

complexidade, onde o risco de avarias ou acidentes é maior. Isso inclui o desenvolvimento de sistemas que contam com monitoramento contínuo condição do robô, alarmes de falha e mecanismos de parada de emergência.

Embora as interfaces de controle robótico avançadas ofereçam grande funcionalidade, muitas delas enfrentam um desafio significativo: são projetadas com ênfase nas funcionalidades técnicas, sem considerar suficientemente a usabilidade para operadores sem formação especializada. Por exemplo, o sistema (ROS - do inglês Robot Operating System) é amplamente utilizado em ambientes de pesquisa e robótica, proporcionando aos desenvolvedores uma enorme gama de funcionalidades. No entanto, sua complexidade e a necessidade de conhecimentos avançados em programação tornam sua operação difícil para aqueles sem formação técnica. Os operadores precisam configurar pacotes e entender profundamente o funcionamento interno do sistema de navegação e controle, o que pode ser uma barreira significativa para usuários iniciantes (Siciliano; Khatib, 2008).

Outro exemplo é o uso de Controladores Lógicos Programáveis (PLC), que são comuns em robôs industriais. Esses sistemas oferecem flexibilidade e precisão, mas exigem um conhecimento profundo sobre programação de controle lógico e a interação entre hardware e software. Para operadores não especializados, a interface do PLC pode ser difícil de usar, pois não é projetada com foco na simplicidade e na usabilidade (Koren; Gu; Guo, 2018). Da mesma forma, as interfaces de controle por força e torque, utilizadas em robôs industriais para manipulação precisa, exigem que o operador tenha compreensão detalhada sobre os sensores de força e torque, bem como sobre os parâmetros de controle. Isso torna a operação dessas interfaces complexa e desafiadora para pessoas sem formação técnica, o que limita a acessibilidade desses sistemas (Siciliano; Khatib, 2008).

Esses exemplos demonstram como a ênfase excessiva nas funcionalidades técnicas das interfaces de controle pode resultar em sistemas difíceis de usar para aqueles sem treinamento específico. Isso resulta em sistemas que exigem amplo conhecimento técnico para operar, dificultando o controle humanizado do robô.

A premissa deste trabalho consiste em utilizar os movimentos das mãos como comandos para controlar um robô móvel. Para a implementação da interface, é vital o uso de modelos de ML capazes de interpretar os gestos realizados pelo usuário. Além disso, a visão computacional desempenha um papel fundamental ao processar os movimentos do operador, permitindo que o robô execute as ações correspondentes.

A combinação desses dois recursos possibilita a criação de um sistema interativo entre o robô e o operador, garantindo que a interface interprete de forma precisa os gestos manuais e os converta em movimentos equivalentes realizados pelo robô.

No decorrer deste trabalho, são abordados de maneira minuciosa os métodos e técnicas potenciais que permitem à interface de controle interpretar de forma eficaz os gestos humanos. Foram treinados algoritmos de aprendizado de máquina para facilitar o reconhecimento de padrões de movimento, realizando uma análise da precisão e da responsividade do sistema

em diversos cenários e com distintas modalidades de comando. Além disso, a utilização de inteligência artificial revela-se promissora para ajustar o comportamento do robô às preferências individuais dos usuários, resultando em um sistema de controle personalizável.

1.1 OBJETIVO PRINCIPAL

O objetivo principal deste trabalho é desenvolver uma interface humano-robô que permita comandar um robô móvel de uma forma mais natural.

1.2 OBJETIVOS ESPECÍFICOS

- Estudar bibliotecas capazes de mapear as mãos e reconhecer faces;
- Desenvolver uma aplicação para controlar os movimentos básicos do LEGO *Mindstorms*;
- Desenvolver uma aplicação de reconhecimento facial;
- Desenvolver uma solução de verificação de permissões de acesso, por usuário, para controle do robô.

1.3 ORGANIZAÇÃO DO TRABALHO

A estrutura do trabalho foi pensada para abordar de forma progressiva e integrada os principais elementos relacionados ao desenvolvimento da interface de controle do robô. Cada seção discute uma base de conhecimento e justifica as etapas executadas no projeto. Abaixo está uma descrição das partes que compõem este trabalho.

O captulo (2) apresenta as bases teóricas necessárias ao desenvolvimento do estudo. Reúne conceitos, técnicas e trabalhos anteriores relacionados com as áreas de controle robótico, dando o suporte necessário para justificar as escolhas do projeto. Detalha o conceito central do trabalho, o desenvolvimento de uma interface que permite o controle de um robô móvel via gestos manuais. São descritas as limitações das interfaces tradicionais e a necessidade de soluções mais acessíveis para operadores sem formação técnica avançada.

Na sequência, o texto (3) discute os fundamentos tecnológicos e algoritmos usados para interpretar os movimentos das mãos como comandos para o robô. As técnicas de tratamento de imagens são explicadas, além de como algoritmos de aprendizado supervisionado foram aplicados para treinar o sistema para reconhecer gestos. Também é abordada a aplicação de técnicas de reconhecimento facial, quando necessário, para distinguir usuários.

A terceira parte (4) discorre sobre o desenvolvimento e funcionamento da interface como um todo, integrando todos os conteúdos apresentados anteriormente, como a detecção das mãos e o reconhecimento facial.

Por fim, a última seção (5) resume o impacto do trabalho no contexto acadêmico e prático.

2 REVISÃO BIBLIOGRÁFICA

Neste capítulo, é apresentada a revisão bibliográfica sobre sistemas robóticos e desafios e inovações nas interfaces de controle. Na seção 2.1, são discutidos os avanços recentes da robótica e sua crescente presença em diversos setores. O uso de robôs móveis, em particular, tem transformado a automação ao viabilizar tarefas como a logística dentro de industrias, onde robôs autônomos estão otimizando processos em armazéns. A robótica colaborativa, representada pelos cobots, que interagem com humanos, também é apontada como um avanço relevante para o aumento da eficiência.

Na educação, destaca-se o papel essencial da robótica móvel no aprendizado prático. Plataformas como o LEGO Mindstorms proporcionam experiências que conectam teoria e prática, preparando os alunos para as demandas do mercado de trabalho.

A seção 2.2 aborda os desafios e inovações nas interfaces de controle, com ênfase em sistemas intuitivos, como as interfaces naturais de usuário (NUIs), que tornam a interação com robôs mais acessível. Entre os avanços destacados estão o uso de gestos, comandos de voz e sistemas baseados em sensores.

2.1 SISTEMAS ROBÔTICOS

Nas últimas décadas, o uso de robôs aumentou significativamente, impulsionado pelas inovações tecnológicas e pela crescente demanda por automação. Em 2020, o *stock* global de robôs industriais ultrapassou os três milhões de unidades, um aumento significativo desde o início da automação industrial na década de 1960.(IFR, 2021).

Na agricultura, a aplicação de robôs expandiu-se para tarefas como plantio, colheita e acompanhamento de colheitas. Na área da saúde, a utilização de robôs tem aumentado nas áreas de cirurgia assistida, diagnóstico e reabilitação, demonstrando um avanço na robótica colaborativa, com robôs concebidos para trabalhar com humanos em ambientes complexos. As previsões mostram que o mercado global de robótica continuará a crescer, atingindo um crescimento anual médio de dois dígitos até 2024, atingindo meio milhão de unidades instaladas anualmente (IFR, 2021).

A robótica móvel tem desempenhado um papel fundamental no avanço da automação, especialmente em setores que exigem flexibilidade, mobilidade e interação com ambientes dinâmicos. Diferentemente dos robôs estacionários, amplamente utilizados em linhas de produção, os robôs móveis são capazes de se deslocar e realizar tarefas em diferentes áreas, tornando-se essenciais em aplicações como logística, agricultura, exploração espacial. Esses robôs podem navegar autonomamente em ambientes não estruturados, adaptando-se às mudanças e interagindo com objetos e pessoas de maneira inteligente. Uma das principais vantagens da robótica móvel é a sua capacidade de aumentar a eficiência em processos logísticos. Em centros de distribuição e armazéns, robôs móveis autônomos (AMRs - do inglês *Autonomous Mobile Robots*) são capazes de transportar mercadorias de forma rápida e segura, reduzindo a necessidade de mão de obra

humana para tarefas repetitivas e perigosas. Um exemplo é o uso de AMRs pela *Amazon*, que implementou milhares de robôs em seus armazéns para otimizar o armazenamento e a entrega de produtos (Wurman; D'Andrea; Mountz, 2008). Esses robôs são equipados com sensores e sistemas de navegação avançados que permitem a movimentação eficiente em grandes áreas, mesmo quando compartilhadas com trabalhadores humanos.

A robótica móvel também apresenta desafios, especialmente em termos de segurança e confiabilidade. Para superar esses obstáculos, muitas empresas têm investido em pesquisas para melhorar a interação homem-robô, buscando tornar esses sistemas mais seguros e eficientes (Kelly, 2020). Um exemplo disso são os *cobots* móveis, que combinam a mobilidade dos robôs autônomos com a capacidade de trabalhar de maneira colaborativa com humanos, o que está transformando o ambiente industrial.

A robótica móvel tem se tornado uma ferramenta essencial no campo da engenharia, especialmente nas universidades, onde o aprendizado prático é fundamental para formar profissionais competentes. O uso de plataformas robóticas como o Robotino, desenvolvido pela Festo Didactic, é um exemplo claro dessa tendência. Robotino é um sistema mecatrônico móvel que permite aos estudantes explorar diversos conceitos de engenharia, programação e controle, promovendo um aprendizado ativo e envolvente (Weinert; Pensky, 2011).

A importância da robótica móvel na educação se reflete no modo como ela conecta teoria e prática. Por meio da construção e programação de robôs, os alunos desenvolvem habilidades em resolução de problemas e pensamento crítico. A inclusão de competições de robótica, como o *WorldSkills* e a *RoboCup*, tem sido uma estratégia eficaz para estimular o interesse dos estudantes em engenharia e tecnologia (Weinert; Pensky, 2011).

Essas experiências são essenciais para preparar os alunos para o mercado de trabalho, onde a habilidade de trabalhar em equipe e resolver problemas de maneira criativa é cada vez mais valorizada.Além disso, a robótica móvel contribui significativamente para o desenvolvimento de *soft skills*, como a comunicação e a colaboração. O trabalho em equipe é fundamental durante as competições, já que os alunos precisam dividir responsabilidades e combinar esforços para alcançar um objetivo comum (Weinert; Pensky, 2011). Essa dinâmica de grupo não só melhora a experiência de aprendizado, mas também reflete as condições de trabalho que os futuros engenheiros enfrentarão em suas carreiras.

A presença crescente de robótica nas universidades também está alinhada com as demandas do mercado de trabalho. A indústria está em constante evolução, e há uma necessidade crescente de profissionais que compreendam as tecnologias emergentes e saibam aplicá-las de maneira eficaz. Assim, as instituições de ensino superior que adotam a robótica móvel em seus currículos estão se posicionando para atender a essa demanda, formando graduados que são não apenas tecnicamente competentes, mas também adaptáveis e inovadores (Weinert; Pensky, 2011).

Um exemplo bastante atual da presença da robótica móvel em ambientes educacionais é LEGO *Mindstorms* uma plataforma de robótica educacional que revolucionou o ensino de ciên-

cias, tecnologia, engenharia e matemática ao permitir que alunos de diversas idades construam e programem seus próprios robôs. A história dessa plataforma remonta à década de 1960, quando Seymour Papert, em suas pesquisas no *Massachusetts Institute of Technology* (MIT), introduziu o conceito de *constructionism*, uma abordagem educacional que enfatiza a aprendizagem através da construção e criação de objetos significativos (Üçgül, 2013). Essa filosofia se tornou a base para o desenvolvimento dos kits de robótica LEGO, que foram projetados para proporcionar uma experiência prática e envolvente.

A versão do robô LEGO *Mindstorms* utilizado neste trabalho foi o EV3, lançado em setembro de 2013, sucedendo ao modelo NXT introduzido em 2006. A principal diferença entre os dois sistemas são as atualizações de hardware e software do EV3, que o tornaram mais poderoso e versátil. O EV3 possui processador ARM9 de 300 MHz, muito superior ao processador de 48 MHz do NXT, além de 64 MB de RAM e 16 MB de memória flash, com opção de expansão via cartão microSD. Também funciona em sistema baseado em *Linux*, o que amplia as possibilidades de programação e integração. O NXT, por outro lado, possuía um sistema mais simples e menos flexível. Outra inovação do EV3 é a introdução de novos sensores, como o giroscópio, que facilita a medição de orientação e rotação, mantendo a compatibilidade com os sensores NXT. Porém, os sensores do EV3 não são compatíveis com o bloco NXT, embora os motores sejam intercambiáveis. O EV3 também permite controle para dispositivos móveis (*Android* e iOS), enquanto o NXT era limitado ao *Android*. Além disso, o *software* EV3 é mais intuitivo, permitindo a configuração dos parâmetros diretamente nos blocos de programação, além de permitir a criação de programas básicos diretamente no tijolo. Isso contrasta com o software NXT, que exigia mais suporte em um computador. Essas melhorias tornam o EV3 uma escolha mais poderosa para educação e projetos avançados, ao mesmo tempo que mantém compatibilidade parcial com componentes NXT para facilitar a transição(Education, 2024a).

Podendo ser conectado por meio de Wi-Fi, *Bluetooth* e USB, permitindo uma ampla gama de integrações e controle remoto. O suporte a Wi-Fi exige o uso de um adaptador USB compatível, e a funcionalidade *Bluetooth* possibilita conexão com dispositivos móveis e outros computadores.

O EV3 suporta uma ampla variedade de linguagens de programação, permitindo que tanto iniciantes quanto usuários avançados explorem suas funcionalidades. A linguagem gráfica oficial, EV3-G, é baseada no *LabVIEW* e projetada para introduzir conceitos de programação de forma intuitiva. *Python*, especialmente através de ambientes como *MicroPython*, é uma alternativa mais flexível para desenvolvedores que desejam maior controle sobre os projetos. Além disso, o *Scratch*, amplamente usado em contextos educacionais, facilita o aprendizado de lógica com sua abordagem de programação em blocos. Para usuários mais experientes, há suporte a linguagens como C e C++, acessadas por bibliotecas específicas como a LeJOS, e *Java*, também disponível por meio de *frameworks* como LeJOS. MATLAB e *Simulink* são opções robustas voltadas para pesquisa e educação, oferecendo ferramentas avançadas para simulação e controle (Education, 2024b).

A robótica educacional proporciona um espaço onde os alunos podem experimentar, falhar e corrigir seus erros, promovendo um ciclo contínuo de aprendizagem. Essa abordagem prática é fundamental para engajar estudantes que podem achar o aprendizado de conceitos teóricos desafiador ou monótono (Üçgül, 2013).

Em suma, a LEGO *Mindstorms* representa uma abordagem eficaz e envolvente para o ensino de ciências e tecnologia, oferecendo aos alunos a oportunidade de se tornarem não apenas consumidores de tecnologia, mas também criadores. A experiência prática de construir e programar robôs proporciona um aprendizado significativo e motivador, alinhando-se com as melhores práticas educacionais e as necessidades do mundo moderno (Üçgül, 2013). Os robôs moveis LEGO *Mindstoms* podem ser programados em linguem nativa (NXT-G), ou em linguagens como C e *Python*, e este foi o robô móvel que se tinha disponível para ser controlado, portanto o escolhido para testar a interface, a versão utilizada pode ser observada na Figura 1.

Figura 1 – LEGO *Mindstorms* utilizado para teste.

Fonte: Elaborado pela autora (2024).

2.2 INTERFACE DE CONTROLE ROBÓTICO

Em termos de controle da locomoção desses robôs, há diversas formas de executá-lo, utilizando inúmeras e diferentes interfaces. A variedade de interfaces de controle robótico reflete a complexidade e as necessidades específicas que surgem com a evolução da robótica. Estas interfaces foram desenvolvidas para atender às diferentes aplicações e funções que os robôs desempenham em ambientes residenciais e em pesquisas acadêmicas. Com os robôs interagindo cada vez mais de maneira direta ou indireta com os seres humanos é necessário que a interação com estas interfaces seja natural.

A ABB fabricante renomada de robôs, lançou no ano de 2024 uma plataforma de controle robótico chamada *OmniCore* que é composta por diversas interfaces integradas, o usuário precisa de um conhecimento básico sobre sistemas de automação industrial e, idealmente, alguma experiência em controle robótico. É importante estar familiarizado com a interface de programação do controlador, ela exige compreensão dos conceitos de configuração de tarefas, parâmetros de

movimento e integração de periféricos. Também é necessário entender os princípios básicos da configuração de sensores, além de saber utilizar ferramentas de otimização, como o "*PickMaster Twin*", que simula processos e ajuda na programação eficiente (ABB, 2024).

Além disso, o usuário deve estar ciente das opções de conectividade disponíveis, como protocolos industriais padrão, e da configuração de redes industriais para integrar o sistema em ambientes de produção existentes. Treinamento nas soluções digitais complementares, como os recursos baseados em nuvem e a computação de borda, também é recomendável para aproveitar ao máximo a interface e implementar soluções escaláveis e inteligentes.

Diante desse cenário, a robótica tem direcionado esforços significativos em pesquisas que visam o desenvolvimento de interfaces mais intuitivas e acessíveis. O objetivo é permitir que usuários com diferentes níveis de conhecimento possam interagir com sistemas robóticos de maneira eficaz, sem barreiras técnicas. Isso não apenas democratiza o uso da robótica, mas também aumenta sua aplicação em novos setores, contribuindo para uma adoção mais ampla e inclusiva.

No campo da interação homem-robô, as interfaces naturais de usuário (NUIs - do inglês *Natural User Interfaces*) têm se destacado como uma forma intuitiva e eficiente de controlar robôs. As NUIs são projetadas para serem intuitivas, flexíveis e fluidas, permitindo que os usuários interajam com os sistemas computacionais de maneira mais natural, muitas vezes sem perceber que estão utilizando uma interface. As primeiras interfaces de computação envolviam a manipulação física dos circuitos, evoluindo para interfaces de linha de comando, mouse, ícones e *pads*. Na última década, houve avanços significativos para reduzir a distância entre as metáforas da experiência humana com o mundo físico e a operação do hardware dos sistemas computacionais.

Pode-se visualizar no trabalho de Palar e Oliveira (2021) é um estudo que busca desenvolver uma interface de controle para um robô escalador, no caso deste trabalho ele se utiliza de um *joystick* industrial, comumente utilizado para controlar pontes rolantes e um Bracelete de Eletromiografia para capturar e sensorear os movimentos do usuário afim de controlar o deslocamento do robô, tem-se aqui um exemplo de interface onde não é necessário um estudo técnico a respeito da interface para utilizá-la.

Pensando em interfaces de controle natural pode-se citar um artigo que tem por tema principal desenvolvimentos recentes na interface humano-robô baseada em linguagem falada do robô Carl de Ferreira et al. (2003), que discorre a respeito das interfaces de controle robótico por meio de voz, este tipo de interface usa uma linguagem natural ao humano, todos os que possuem a capacidade de se comunicar pela fala podem controlar robôs que possuem este tipo de interface, quebrando a barreira da necessidade de conhecimentos técnicos para utilizá-los.

No trabalho de Resende (2006) se discute uma nova abordagem para o desenvolvimento de interfaces visuais homem-robô baseadas em visão computacional. A proposta foca na utilização de uma linguagem composta por gestos simples, onde cada gesto isolado não tem significado, mas uma sequência ordenada de gestos forma "palavras" que desencadeiam respostas do robô.

Isto permite definir uma gramática associada a ações específicas, o que permite ao operador humano controlar o robô de forma intuitiva.

O sistema utiliza técnicas qualitativas de visão computacional, privilegiando a consistência em detrimento da precisão, o que contribui para sua eficácia mesmo em ambientes complexos ou com movimentação do operador. A detecção do comando é realizada de forma contínua, o reconhecimento do início e do fim dos gestos é identificado implicitamente pelo modelo. Dois tipos de sistemas baseados em eventos discretos foram implementados e testados para este reconhecimento: cadeias de Markov e modelos ocultos de Markov (HMMs - do inglês *Hidden Markov Vodels*). A construção dos modelos foi automatizada e os resultados mostram a superioridade dos HMMs em termos de robustez e baixa incidência de falsos positivos. Além disso, o estudo destaca a importância da escolha de comandos adequados para garantir altas taxas de reconhecimento, mesmo quando a complexidade do sistema aumenta.

As interfaces baseadas em gestos podem ser classificadas por diversos parâmetros. Os gestos a serem identificados podem ser estáticos, identificados pela posição e forma, ou dinâmicos, identificados por sua trajetória. A abordagem pode ser *bottom-up* baseada nas caraterísticas de baixo nível da imagem ou *top-down* como métodos de reconstrução geometria do corpo. O reconhecimento pode ainda ser realizado em três dimensões ou simplificadamente em duas dimensões (Resende, 2006).

Interfaces de reconhecimento de gestos tornando-se popular após o abandono do paradigma de "toque à distância" na década de 80 (Campos; Santos, 2019). Um exemplo notável de NUI é o sensor *Kinect* da *Microsof*, que foi utilizado inicialmente no console de videogame *Xbox One*. Devido ao seu baixo custo e disponibilidade de kits de desenvolvimento, o *Kinect* se tornou um sensor de desenvolvimento em engenharia, computação, arte e na cultura *maker*. O sensor *Kinect* permite o reconhecimento de gestos, varrendo a posição das juntas do esqueleto humano através de sua câmera de profundidade, estimando a posição das juntas no espaço e obtendo um modelo da posição desejada para controlar um braço robótico.

O uso de interfaces naturais de usuário NUIs com *serious games* é natural e sugere uma infinidade de aplicações em serviços e produtos. Aplicações atuais em terapia e reabilitação médica utilizando dispositivos como o *Microsoft Kinect*, processadores digitais de imagem, visão computacional e robótica assistiva são demonstradas em todo o mundo com alto potencial de geração de valor. Outras possíveis aplicações incluem a manipulação de materiais em ambientes não estruturados, como locais com presença ou risco de presença de agentes químicos, biológicos ou físicos perigosos (Campos; Santos, 2019).

Os sensores também são ferramentas utilizadas no desenvolvimento de interfaces inteligentes, no trabalho do (Cui et al., 2022) é abordado o desenvolvimento de um sistema de controle de cadeira de rodas inteligente baseado em *Internet of Things* (IoT), projetado para suportar múltiplos modos de operação, com foco na melhoria da mobilidade e acessibilidade para usuários com deficiência. Ele explora tecnologias de sensores, conectividade e algoritmos para viabilizar o controle remoto e autônomo da cadeira de rodas.

O estudo investiga como integrar diferentes módulos, como sistemas de navegação e interfaces homem-máquina, utilizando IoT para oferecer comandos personalizados e seguros. Além disso, o trabalho busca garantir eficiência energética, precisão na execução de comandos e acessibilidade, abordando desafios relacionados à interação usuário-máquina e ao ambiente dinâmico em que a cadeira de rodas opera. O sistema também prioriza a capacidade de adaptação a diferentes necessidades dos usuários, com modos de controle como comando de voz, *joystick* e outros métodos alternativos.

A pesquisa não apenas descreve os componentes técnicos e funcionais do sistema, mas também avalia seu desempenho em cenários experimentais, oferecendo resultados práticos que demonstram a viabilidade e eficácia da solução proposta.

O Volkswagen Golf R Touch é um protótipo apresentado pela marca alemã que incorpora um avançado sistema de controle por gestos, revelado inicialmente no *Consumer Electronics Show* 2015. Este modelo utiliza sensores de proximidade e uma câmera 3D no teto para detectar movimentos das mãos, permitindo aos usuários interagir com diversas funções do veículo sem tocar na tela. A tecnologia inclui gestos tridimensionais para operar sistemas como teto solar, ajuste de iluminação interna e controle de multimídia.

A interface central é composta por uma tela de 12,8 polegadas que suporta personalização semelhante a dispositivos móveis, como *smartphones*. Janelas de informações podem ser ajustadas em tamanho e posição, enquanto o design minimalista do console elimina a maioria dos botões físicos, substituindo-os por controles sensíveis ao toque e gestos. Além disso, o sistema diferencia comandos com base no número de dedos usados, permitindo ajustes intuitivos e precisos, como o controle de volumes separados para áudio e navegação (Volkswagen, 2015).

O Golf R Touch é um exemplo de como a Volkswagen explora interfaces intuitivas para melhorar a interação humano-veículo, integrando design futurista e funcionalidades inovadoras para maior conveniência e eficiência (Volkswagen, 2015).

2.3 CONCLUSÃO DO CAPITULO

Até aqui discutiu-se o avanço e a diversificação da robótica nas últimas décadas, com ênfase no aumento da utilização de robôs industriais e móveis em diversos setores, como indústria, agricultura, saúde e educação. Em 2020 o *stock* global de robôs industriais ultrapassou os três milhões de unidades, impulsionado pela crescente procura de automação e pela evolução de setores mais avançados, como a indústria eletrônica. Na robótica móvel, a sua aplicação é evidente em ambientes dinâmicos, onde oferece flexibilidade e eficiência, principalmente em contextos logísticos e educacionais. A utilização de plataformas como LEGO *Mindstorms* é vista como uma ferramenta prática e inovadora para a aprendizagem de ciência e tecnologia, que promove a aprendizagem ativa e o desenvolvimento de competências críticas e criativas.

Além disso, faz notar que as interfaces tradicionais muitas vezes exigem conhecimento técnico, criando barreiras para usuários sem treinamento especializado.

Pesquisas recentes têm se concentrado em interfaces naturais de usuário, como controles de gestos, dispositivos de captura de voz e movimento, que facilitam a interação humano-robô. Estudos e tecnologias, como modelos ocultos de Markov para reconhecimento de gestos e sistemas baseados em visão computacional, mostram avanços no desenvolvimento de sistemas mais poderosos e eficientes, capazes de interpretar os comandos.

Por isso o estudo e desenvolvimento de interfaces de controle por meio de voz, gesto e outros meios de comunicação humana são uma solução a ser estudada e desenvolvida em trabalhos acadêmicos e no meio industrial.

3 VISÃO COMPUTACIONAL

A interface humano-robô desenvolvida conta com controle via movimentos de mãos humanas, sem a utilização de sensores, mas por meio da tecnologia de visão computacional, tornando o controle de robôs mais natural.

3.1 CAPTURA E TRATAMENTO DE IMAGENS

A captura em vídeo dos movimentos de mãos humanos, primeiramente se pensou em utilizar a câmera de vídeo *RealSense* D435 da intel que possui um sensor RGB (*Red, Green, Blue*) e mede a profundidade de objetos ou cenas em ambientes 3D. Esses sensores são amplamente utilizados em diversas aplicações, desde câmeras digitais até sistemas de automação industrial e dispositivos de IoT. Porém, no desenvolvimento da interface percebeu-se que nenhuma das ferramentas tecnológicas presentes na câmera estava sendo utilizada, optou-se então pela utilização de uma câmera simples de webcam *Life*Cam HD-3000 da *Microsoft* que atende perfeitamente as necessidades do projeto, e a biblioteca *Open Source Computer Vision Library* (OpenCV) utilizada em projeto conta com o recurso RGB de tratamento de imagem.

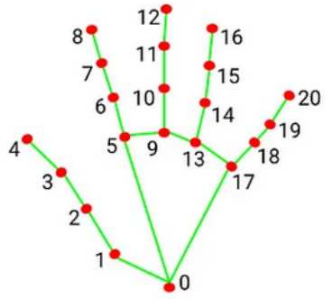
Existem pesquisas visando o estudo e desenvolvimento de ferramentas computacionais com a finalidade de realizar reconhecimento de partes do esqueleto humano, bem como a posição em que este se encontra, esta tecnologia é chamada de estimativa de pose. O *Google* desenvolveu o MediaPipe *Solutions*, uma poderosa ferramenta de aprendizado de máquina que oferece um conjunto abrangente de bibliotecas e ferramentas para a aplicação rápida de técnicas de IA e machine learning (ML) em diversos aplicativos. Essa plataforma pode ser personalizada para atender a diferentes necessidades e é compatível com várias plataformas de desenvolvimento, permitindo a criação de soluções escaláveis e interativas que integram IA diretamente em dispositivos móveis, navegadores e sistemas em tempo real.

O MediaPipe *Solutions* faz parte do projeto de código aberto do MediaPipe. Assim, é possível personalizar ainda mais o código das soluções para atender às necessidades do seu aplicativo (Google, 2024b). A tarefa MediaPipe Pose *Scoreer* permite detectar pontos de referência de corpos humanos em uma imagem ou vídeo. Essa tarefa permite identificar as principais localizações do corpo, analisar a postura e categorizar movimentos. Para isso, utiliza modelos de ML que operam com imagens ou vídeos únicos. A tarefa gera pontos de referência para as posições corporais na imagem e também em coordenadas tridimensionais no espaço (Google, 2024a).

O *framework* MediaPipe, desenvolvido pelo *Google*, inclui uma função chamada *Hand Landmarker*, que possibilita a detecção de pontos de referência das mãos em imagens. Esse recurso é útil para localizar pontos-chave das mãos e aplicar efeitos visuais sobre eles. A função opera em dados de imagem com um modelo de ML, podendo processar dados estáticos ou fluxos contínuos, e gera coordenadas de pontos de referência das mãos na imagem, marcos de posição em coordenadas do mundo, além de identificar múltiplas mãos (direita ou esquerda) detectadas.

Essa função atende à necessidade de rastreamento de mãos em imagens capturadas em tempo real, identificando 21 pontos de referência nas mãos, conforme ilustrado na Figura 2.

Figura 2 – Ponto de referência da mão para o modelo.



0.Pulso
1. Base do Polegar
2. Articulação do Polegar
3. Articulação Interfalângica do Polegar
4. Ponta do Polegar
5. Articulação do Indicador
6. Articulação Proximal do Indicador
7. Articulação Distal do Indicador
8. Ponta do Indicador
9. Articulação do Médio
10. Articulação Proximal do Médio
11. Articulação Distal do Médio
12. Ponta do Médio
13. Articulação do Anelar
14. Articulação Proximal do Anelar
15. Articulação Distal do Anelar
16. Ponta do Anelar
17. Articulação do Mínimo
18. Articulação Proximal do Mínimo
19. Articulação Distal do Mínimo
20. Ponta do Mínimo

Fonte: Modificado de RIGOR(2021).

Pode-se observar na Figura3 os FPS (*Frames Per Second*), ou quadros por segundo, uma medida de quantas imagens individuais são exibidas em um segundo em um vídeo ou animação. Pense em um vídeo como uma série de fotos que são mostradas rapidamente uma após a outra. Cada uma dessas "fotos" é chamada de *frames*. Quanto mais quadros forem exibidos a cada segundo, mais suave será o movimento que você vê. Um vídeo com 30 FPS mostra 30 imagens diferentes por segundo, 60 FPS mostra 60 imagens por segundo, o que geralmente faz o movimento parecer ainda mais fluido.

Visando o controle exclusivo do robô móvel por meio da interface, optou-se por realizar o reconhecimento facial do controlador (humano), para isso utilizou-se uma biblioteca desenvolvida pela Intel em 1999 chamada de OpenCV, uma poderosa ferramenta de código aberto para processamento de imagens e visão computacional. Afim de fornecer uma infraestrutura eficiente que permita a implementação de algoritmos complexos em tempo real, amplamente utilizados em áreas como robótica, detecção de objetos, reconhecimento facial, entre outras.

A *OpenCV* suporta muitas linguagens de programação, como *Python*, C++, *Java* e *MATLAB*, e é integrado a bibliotecas como NumPy, que permitem manipulação eficiente de *arrays* e integração com outros *frameworks*. Ele também possui recursos de manipulação de vídeo, processamento em tempo real e aprendizado de máquina. Além de ser amplamente utilizado para detecção de rostos e reconhecimento de formas, o *OpenCV* também suporta diversas operações de processamento de imagens, como filtragem, transformação geométrica, segmentação, reconhecimento de objetos, etc (Bradski; Kaehler, 2008).

Figura 3 – Implementação do identificador de mãos.

Fonte: Elaborado pela autora (2024).

3.2 APRENDIZAGEM DE MAQUINA E RECONHECIMENTO FACIAL

Diversos estudos têm sido conduzidos nas últimas décadas sobre ML, com aplicações em diversas áreas do conhecimento. O aprendizado de máquina é um subcampo da IA que, em termos gerais, refere-se à capacidade de uma máquina executar funções que os humanos associam a processos mentais, como “aprendizagem” e “resolução de problemas” (Shinde; Shah, 2018). Esse aspecto de aprendizado é fundamental para a operação e desenvolvimento de sistemas de IA permitindo que máquinas aprimorem seu desempenho e adaptação a partir de dados e experiências acumuladas.

Dentro do campo da ML existe a aprendizagem de máquina supervisionada, uma tarefa de aprendizado de máquina de aprender uma função que mapeia uma entrada para uma saída com base em pares de entrada-saída de exemplo. Ele infere uma função de dados de treinamento rotulados que consistem em um conjunto de exemplos de treinamento. Os algoritmos de aprendizado de máquina supervisionado são aqueles algoritmos que precisam de assistência externa. O conjunto de dados de entrada é dividido em conjunto de dados de treinamento e teste. O conjunto de dados de treinamento tem variável de saída que precisa ser prevista ou classificada (Mahesh, 2020).

A biblioteca *OpenCV* é capaz de realizar o reconhecimento facial utilizando *Deep Learning* (DL), Modelo *Deep Neural Network* (DNN), pode ser realizado também por meio de classificadores do tipo *Haarcascade*, que são algoritmos utilizados para identificar e classificar objetos ou padrões em imagens ou vídeos. O Haarcascade é uma técnica de detecção de objetos desenvolvida por Viola e Jones em 2001, sendo uma das abordagens mais comuns para a

detecção de rostos. Ele usa características baseadas em retângulos, chamadas de *Haar features*, para identificar padrões visuais em imagens. O classificador é treinado usando um conjunto de imagens positivas (com o objeto de interesse, como um rosto) e negativas (sem o objeto), permitindo que ele aprenda a distinguir entre os dois. Envolve um processo supervisionado de aprendizado de máquina baseado em um grande conjunto de dados rotulados. Este conjunto é composto por duas categorias principais: imagens positivas, que contêm o objeto alvo (como o rosto humano), e imagens negativas, que não apresentam o objeto (como paisagens e objetos diversos). Essa abordagem permite que o algoritmo aprenda a distinguir entre as duas classes, extraindo padrões e características visuais comuns às imagens positivas (Viola; Jones, 2001).

Os recursos usados pelo classificador *Haarcascade* são chamados de recursos *Haar*, que descrevem padrões de contraste claro-escuro em regiões específicas da imagem. Esses recursos são baseados em diferenças na intensidade de *pixels* entre áreas adjacentes e são eficazes na captura de padrões simples, como bordas horizontais, verticais e diagonais. Ao aplicar esses recursos a uma imagem, o classificador consegue identificar regiões que possuem as propriedades desejadas, facilitando a detecção de objetos. Para treinamento de classificadores, é comum usar o algoritmo *AdaBoost*, que combina vários classificadores fracos em um único classificador poderoso. *AdaBoost* é responsável por selecionar as características mais importantes do *Haar* durante o processo de treinamento, atribuindo pesos a cada uma com base em sua capacidade de distinguir exemplos positivos de negativos. Este ajuste contínuo dos pesos ajuda a aumentar a eficiência do classificador.

Uma característica importante do classificador *Haarcascade* é sua estrutura em cascata. Esta abordagem organiza os classificadores em etapas sequenciais, onde cada etapa consiste em um pequeno número de funções simples. Caso uma determinada área da imagem não seja detectada numa primeira fase como contendo o objeto, ela é removida rapidamente, evitando assim transformações desnecessárias nas etapas subsequentes. Isto torna o processo de detecção muito mais eficiente, permitindo ao classificador rejeitar rapidamente áreas que não contenham o objeto de interesse.

Após o treinamento, o classificador é testado em um conjunto de dados de validação, o que é essencial para melhorar sua precisão e reduzir o número de falsos positivos e negativos. O resultado desse processo é um classificador poderoso que pode ser utilizado para detecção em tempo real, como visto nos detectores faciais implementados na biblioteca *OpenCV. Haarcascade* é aplicado a uma imagem em diferentes escalas, permitindo ao classificador identificar objetos em diferentes tamanhos e posições. Essa capacidade de detectar rostos e olhos com eficácia em tempo real torna o classificador *Haarcascade* uma ferramenta valiosa em várias aplicações de visão computacional, incluindo segurança, interação homem-máquina e reconhecimento facial (Lienhart; Maydt, 2002).

O programa em *Python* desenvolvido para realizar o reconhecimento facial foi dividido em três partes; captura de imagem de faces, treinamento e, por fim, reconhecimento de faces em tempo real. Na primeira etapa, o código começa importando as bibliotecas necessárias e

define os caminhos do arquivo XML com *Haarcascades*, que são classificadores usados para detectar rostos e olhos em imagens. Esses classificadores são carregados e configurados para realizar detecções, e o programa solicita ao usuário um ID, que será utilizado para nomear os arquivos de imagem. As dimensões padrão da imagem capturada são definidas como 220 x 220 *pixels*. O código também cria uma pasta chamada "fotos", onde as imagens serão armazenadas, caso esta ainda não exista. Em seguida, a câmera é inicializada e os parâmetros de captura de vídeo são ajustados, como o formato de codificação *Motion* JPEG. Um *loop* inicia a gravação de imagens da câmera em tempo real e, a cada quadro, a imagem é convertida em tons de cinza para facilitar a detecção de rostos. Com o classificador de faces, o código identifica as possíveis faces da imagem, retornando suas coordenadas, e para cada rosto detectado, um retângulo vermelho é desenhado ao redor do rosto.

A detecção dos olhos é realizada na área do rosto, e novos retângulos vermelhos são desenhados nos olhos detectados. Existe uma lógica de temporização controlada por uma função chamada *snapshot*, que impede a captura contínua de imagens. Quando a variável de captura está habilitada e exatamente um rosto e dois olhos são detectados, o programa salva a imagem original com um nome que inclui o ID do usuário e a quantidade de amostras já capturadas. Após cada disparo, a função *shot_timer* reinicia o disparo após um intervalo de meio segundo, garantindo que as imagens não sejam gravadas em sequência rápida demais.

As imagens são exibidas em uma janela em tempo real. O *loop* de captura continua até que o número de amostras atinja o limite especificado (num*Shows*). Quando esse número é atingido, o *loop* é encerrado e todas as janelas do OpenCV são fechadas. Por fim, uma mensagem é exibida ao usuário informando que as imagens foram capturadas com sucesso.

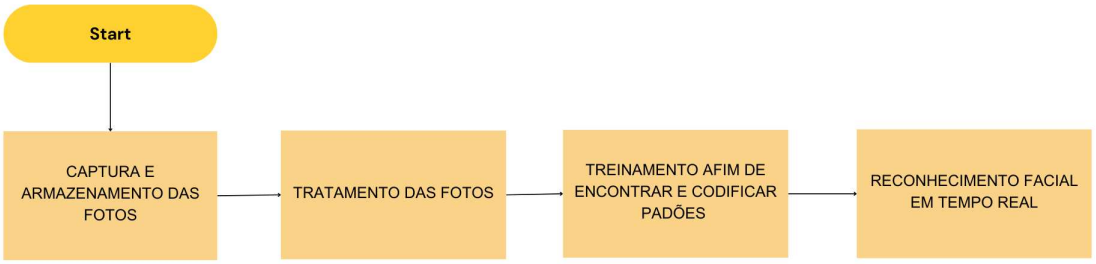
A segunda etapa é o treinamento. Após a captura e armazenamento das imagens, o treinamento é realizado para que o programa aprenda a encontrar e codificar padrões. O código utiliza a função *list_images* do pacote *imutils* para obter os caminhos das imagens contidas na pasta fotos, resultando em uma lista chamada *imagePaths* que armazena os caminhos completos das imagens a serem processadas. Paralelamente, duas listas vazias, codificações conhecidas e nomes_conhecidos, são inicializadas para armazenar os vetores de codificação facial e os nomes correspondentes a cada face detectada. O código entra em um *loop* que percorre cada caminho de imagem presente em *imagePaths*. Para cada imagem, o índice e o caminho são recuperados, e uma mensagem de progresso é impressa informando que a imagem está sendo processada. O nome da pessoa é extraído do caminho da imagem, principalmente da parte entre a última barra e o ponto antes da extensão do arquivo. A imagem é carregada usando a função cv2.*imread* e convertida de BGR (usado pelo OpenCV) para RGB (usado pelo dlib) usando a função cv2.cvt*Color*.

Após essa conversão, a detecção de rosto é realizada usando a função *face_recognition.face_locations*, que retorna as coordenadas da caixa delimitadora de cada rosto encontrado na imagem. O modelo de detecção utilizado é as redes neurais convolucionais (CNNs - do inglês *Convolution Nerural Networks*), mais preciso e robusto. Após identificar as

caixas delimitadoras, o código calcula as codificações faciais correspondentes com a função *face_recognition.face_encodings*, que gera um vetor de características único para cada face detectada. Em um novo ciclo, para cada vetor de codificação, o código adiciona essa codificação à lista de codificações_conhecidas e o nome correspondente à lista de nomes_conhecidos.

Finalmente, após processar todas as imagens, o código serializa os dados de codificação e os nomes em um arquivo usando o módulo *pickle*. Uma mensagem informativa é exibida indicando que a serialização da codificação está em andamento. Os dados são armazenados em um dicionário com as chaves “codificações” e “nomes”, que são então gravados em um arquivo chamado *data.db*. Esse arquivo é aberto no modo de gravação binária, os dados são gravados e o arquivo é fechado. Esse processo permite o armazenamento de dados de reconhecimento facial para uso posterior, facilitando a identificação de rostos em novas imagens. A Figura 4 apresenta um fluxograma que ilustra, de forma simplificada, o funcionamento do programa de reconhecimento facial.

Figura 4 – Fluxograma de funcionamento do programa de reconhecimento facial.



Fonte: Elaborado pela autora (2024).

4 DESENVOLVIMENTO

Neste capítulo, é descrito o desenvolvimento de uma interface em Python que integra visão computacional e controle robõtico, utilizando uma câmera para realizar a detecção de rostos e mãos. A interface é composta por diversas bibliotecas, como *OpenCV*, *Mediapipe* e *Face Recognition*, que permitem a identificação de marcadores nas mãos e rostos, além de calcular os deslocamentos das mãos em relação ao rosto do operador.

Com esses dados, é possível controlar o robô LEGO *Mindstorms*, realizando comandos como avanço, recuo e rotação, baseados em gestos manuais. A interface utiliza *threads* para garantir o processamento simultâneo da captura de vídeo e da detecção, além de exibir *feedback* visual, como a posição das mãos e o FPS, em tempo real.

O código também permite o reconhecimento facial para verificar a identidade do usuário e restringir o controle do robô ao administrador.

4.1 INTERFACE HUMANO-ROBÔ

A interface desenvolvida em linguagem *Python* é um sistema que combina visão computacional e controle de robô, utilizando uma câmera para detecção de faces e mãos. O código começa importando várias bibliotecas necessárias, como *threading* para executar múltiplas tarefas simultaneamente, *socket* para conexões de rede, e cv2 (OpenCV) junto com MediaPipe e *face_recognition* para a detecção de rostos e mãos. Os dados previamente armazenados de faces conhecidas são carregados utilizando o módulo *pickle* para manipulação de objetos serializados. Em seguida, é estabelecida uma conexão com um dispositivo remoto baseado no sistema EV3, que é uma plataforma de robótica, permitindo o controle dos motores e sensores do robô LEGO. O código inicializa variáveis e modelos de detecção, configurando um *mutex* para garantir o acesso seguro a variáveis compartilhadas entre diferentes *threads*.

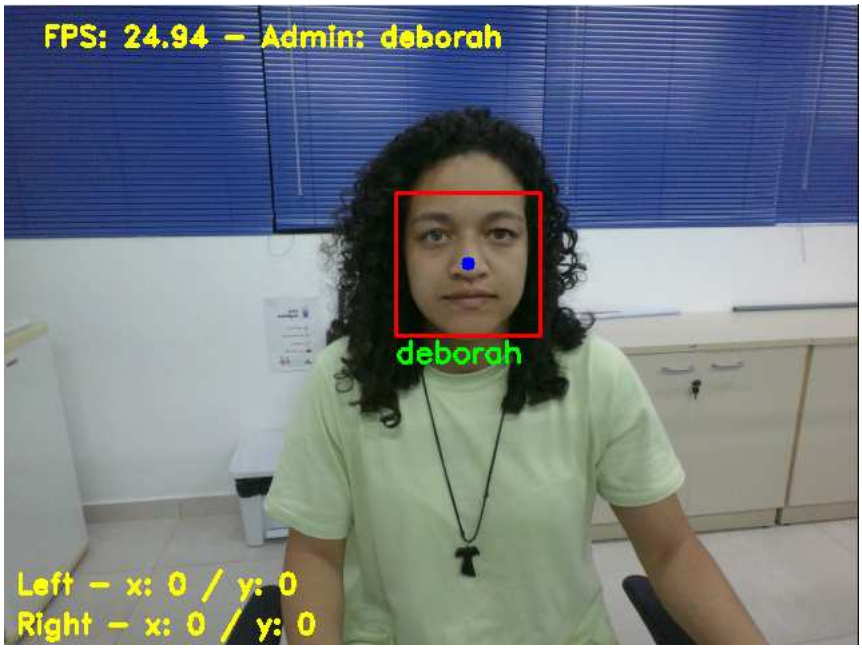
No início da função *detect()*, o modelo é configurado para detectar até duas mãos simultaneamente, com um nível mínimo de confiança de 0,5. A cada iteração do *loop*, a imagem capturada pela câmera é convertida de (BGR - do inglês Blue, Green, Red) para (RGB - do inglês Red Gree, Blue) e processada pelo modelo para detectar as *landmarks* (marcadores) das mãos. Quando as mãos são detectadas, o código verifica quantas mãos estão presentes. Se duas mãos forem detectadas, a variável *hands_detected* é atualizada para *True*.

O processamento da imagem também envolve a identificação dos centros das mãos. Para isso, o código percorre as *landmarks* das mãos detectadas e calcula a posição média dos pontos que correspondem aos dedos, especificamente os pontos 0, 1, 2, 5, 9, 13 e 17 de cada mão. Essa média é utilizada para determinar o centro de cada mão, armazenando os resultados nas variáveis *left_hand_center* e *right_hand_center*. O cálculo do deslocamento das mãos em relação ao rosto do administrador é realizado na função *control()*. Aqui, é necessário determinar a posição do rosto, que é representada pelo centro da face. Após a detecção da face o programa verifica se este é o administrador por meio do programa de reconhecimento facial, se for o

administrador, o código calcula a média das coordenadas (x1, y1) e (x2, y2) que delimitam o retângulo ao redor da face como pode ser visto na Figura 5, resultando em um ponto central chamado *enter_admin_face*.

Os deslocamentos das mãos são então calculados em relação a esse ponto central da face. Para a mão esquerda, o deslocamento em x é determinado subtraindo a coordenada x do centro da mão esquerda (*leftvhand_center[0]*) da coordenada x do centro da face (*center_admin_face[0]*). Similarmente, o deslocamento em y é calculado subtraindo a coordenada y da mão esquerda da coordenada y do centro da face. O mesmo processo é realizado para a mão direita, onde as coordenadas de deslocamento são armazenadas em *displacement_x_right* e *displacement_y_right*. Ao observar a Figura 6, que apresenta um fluxograma simplificado do funcionamento da interface, é possível compreender seu funcionamento de maneira clara e objetiva.

Figura 5 – Reconhecedor facial.

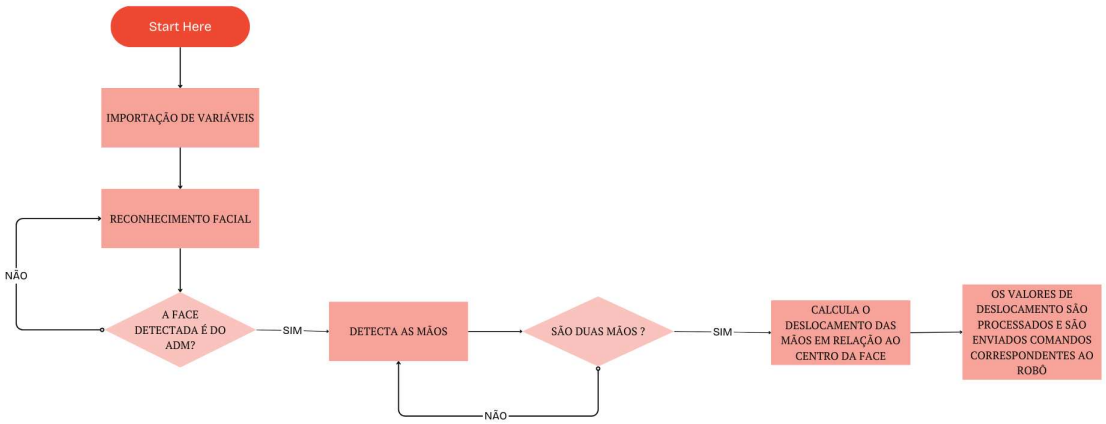


Fonte: Elaborado pela autora (2024).

Esses deslocamentos são utilizados para controlar o movimento do robô. O sistema analisa os valores de deslocamento e decide se deve mover o robô para frente, para trás ou girar, dependendo da posição relativa das mãos em relação ao rosto do administrador e adiciona marcadores do centro da mão e da face como pode ser verificado na Figura 7, e amostra da janela a posição X Y da mão direita e esquerda. Além disso, a função *capture()* é responsável por capturar os quadros da câmera e processá-los para garantir que o sistema esteja continuamente recebendo dados atualizados.

A parte principal do código cria e inicia *threads* para captura de vídeo e detecção, enquanto um *loop* principal processa as informações recebidas, desenhando retângulos em torno das faces detectadas e exibindo informações sobre o estado atual, como o FPS e a posição das mãos. Se o administrador não estiver presente por um tempo determinado, o sistema atualiza seu

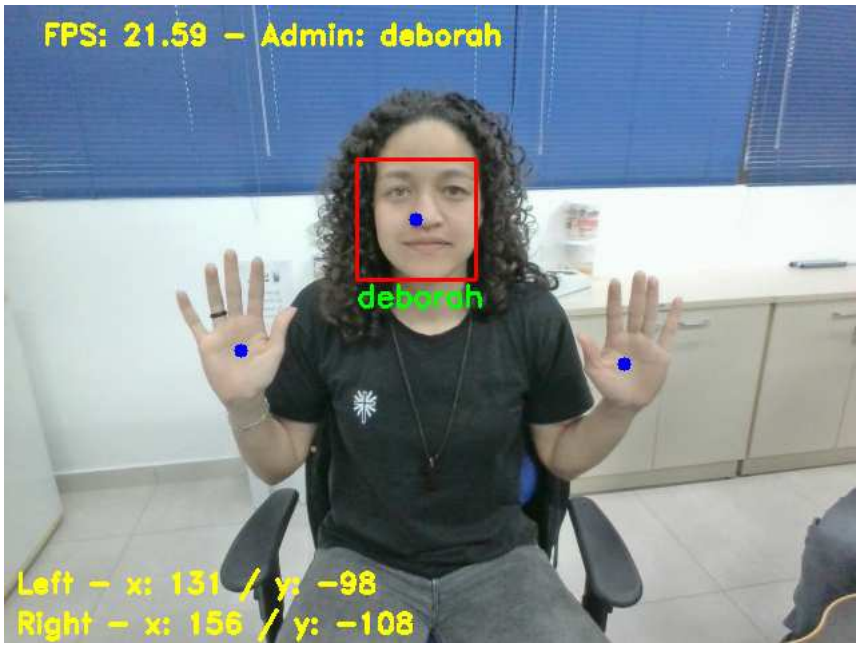
Figura 6 – Fluxograma de funcionamento da interface.



Fonte: Elaborado pela autora (2024).

estado para refletir isso. A visualização é realizada através de uma janela que mostra a imagem capturada, juntamente com as informações processadas. Essa interação entre a detecção de mãos e rostos, juntamente com o cálculo de deslocamentos, permite que o robô responda de maneira dinâmica aos gestos do usuário, criando um sistema interativo e adaptável ao ambiente em que opera.

Figura 7 – Marcadores do centro das mãos e face.



Fonte: Elaborado pela autora (2024).

O sistema utiliza *threads* para processamento simultâneo de vídeo e detecção. A função de captura mantém os dados atualizados, enquanto um *loop* principal processa as informações e exibe *feedback* visual em tempo real, como posições das mãos e FPS. Caso o operador se ausente, o sistema altera seu estado automaticamente. A interface cria uma interação dinâmica e

intuitiva entre o operador e o robô, possibilitando comandos precisos sem interação direta com o código. O código fonte da interface está disponível em Maia (2024a).

Para fazer a conexão entre o computador e o EV3 foi utilizada a biblioteca rpyc, que permite um programa *python* em um dispositivo, controlar outro. Isso é mais comumente conhecido como "encadeamento de margaridas". Isto pode ser útil se o seu robô requer código intensivo da unidade central de processamento que seria lento para ser executado no EV3. É preciso conectividade IP entre os dispositivos, onde o código *python* é executado (laptop, um dispositivo ev3dev, etc) e os dispositivos ev3dev remotos. Alguns cenários comuns pode ser: Múltiplos EV3s na mesma rede Wi-Fi Um laptop e um EV3 na mesma rede Wi-Fi (lang, 2024a). O passo a passo a respeito da instalação da biblioteca pode ser encontrado em (lang, 2024b).

4.2 RESULTADO

O sistema foi projetado para operar de forma eficiente, utilizando *threads* para processar simultaneamente os dados de vídeo e detecção. A função de captura assegura a atualização contínua das informações, enquanto um *loop* principal processa e exibe o *feedback* visual em tempo real, incluindo a posição das mãos, a detecção do rosto e o FPS. Além disso, o sistema reage automaticamente à ausência do operador, ajustando seu estado conforme necessário.

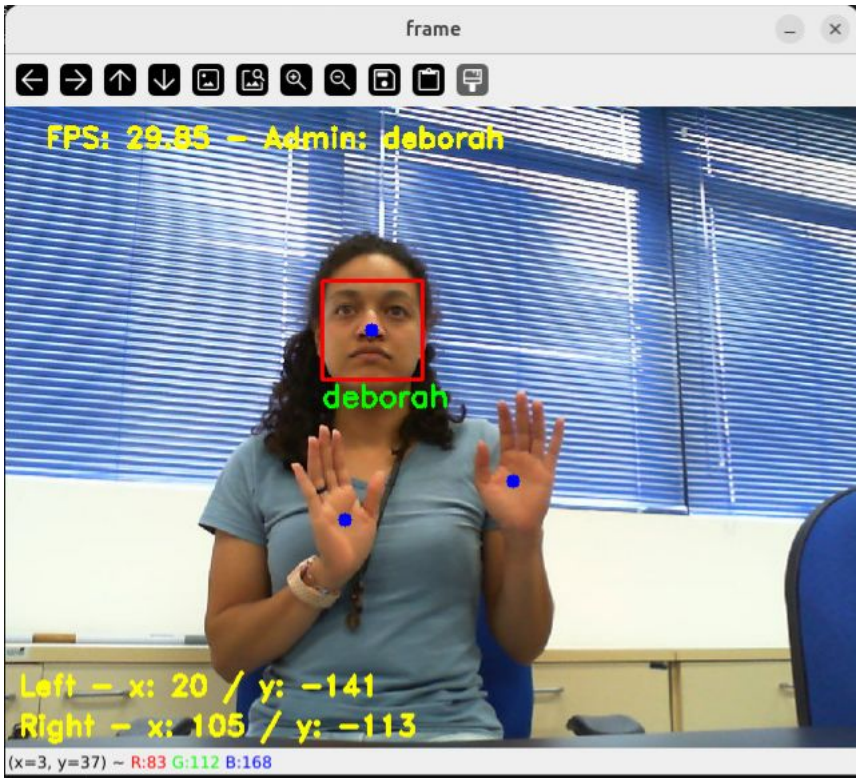
Essa solução apresenta uma interação entre o operador e o robô, eliminando a necessidade de comandos manuais no código, ao mesmo tempo em que proporciona um controle preciso e responsivo por meio de gestos. A combinação de tecnologias e a abordagem estruturada demonstram a viabilidade do sistema desenvolvido.

A interface desenvolvida em *Python* que integra visão computacional e controle robótico, permitindo a detecção de rostos e mãos através de uma câmera para controlar um robô LEGO *Mindstorms*. A interface utiliza bibliotecas como *OpenCV*, *Mediapipe* e *Face Recognition* para identificar marcadores nas mãos e calcular deslocamentos em relação ao rosto do operador.

Os comandos responsáveis pelos movimentos do robô são definidos com base nos deslocamentos descritos.

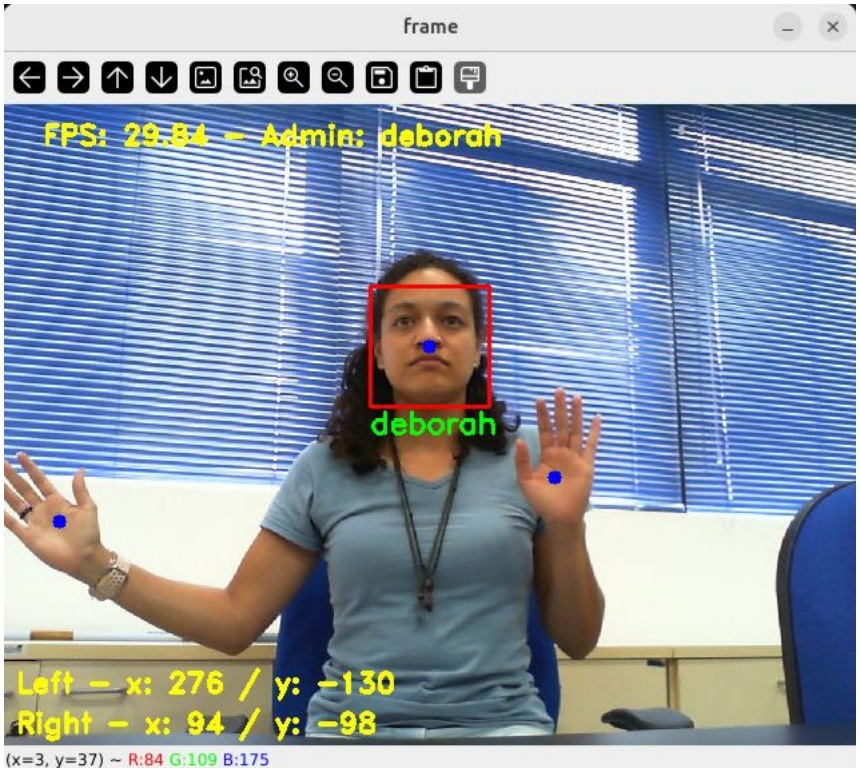
- O avanço, demonstrado no vídeo Maia (2024b), ocorre ao mover a mão esquerda para a lateral esquerda, como ilustrado na Figura 9.
- O recuo, mostrado no vídeo Maia (2024c), é realizado ao deslocar a mão esquerda para a direita, conforme apresentado na Figura 8.
- A rotação no sentido horário, exibida no vídeo Maia (2024d), é acionada pelo movimento da mão direita para baixo, como representado na Figura 10.
- A rotação no sentido anti-horário, evidenciada no vídeo Maia (2024e), é ativada ao deslocar a mão direita para cima, conforme mostrado na Figura 11.

Figura 8 – Mão esquerda para a direita.



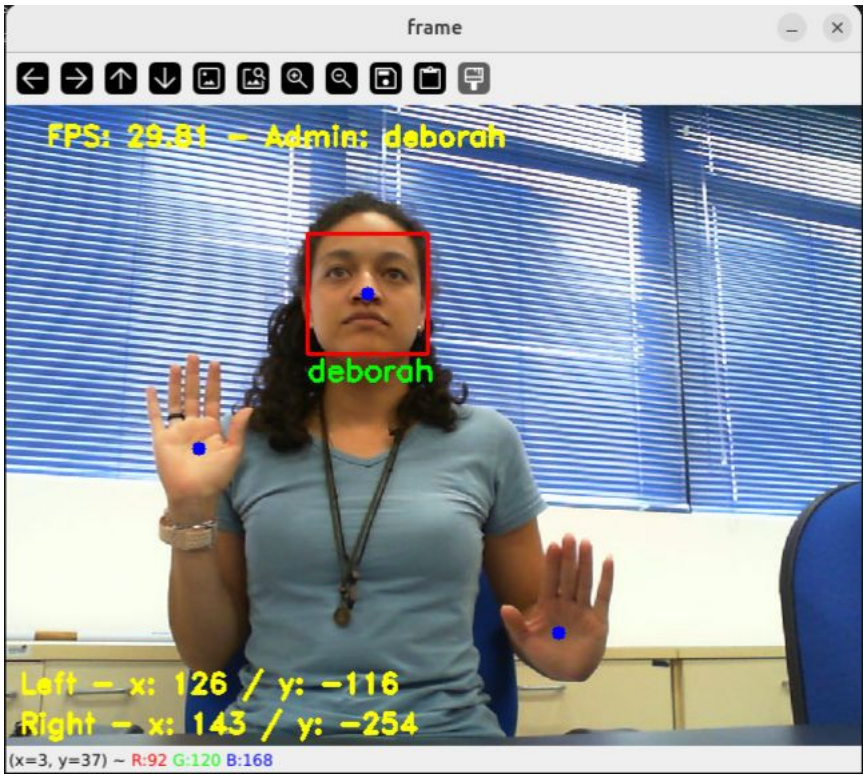
Fonte: Elaborado pela autora (2024).

Figura 9 – Mão esquerda para a lateral esquerda.



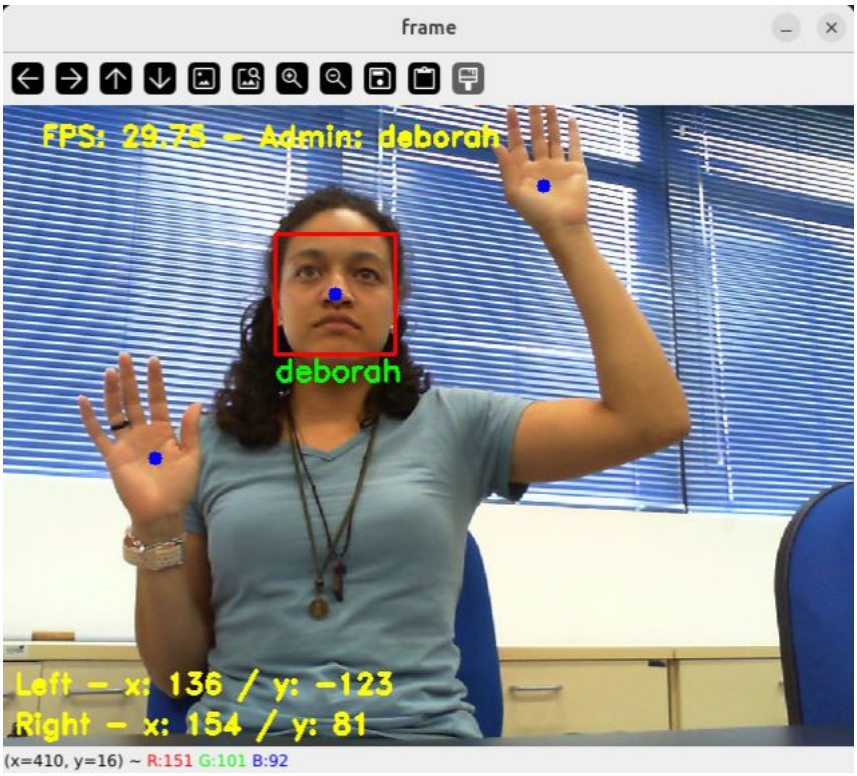
Fonte: Elaborado pela autora (2024).

Figura 10 – Mão direita para baixo.



Fonte: Elaborado pela autora (2024).

Figura 11 – Mão direita para cima.



Fonte: Elaborado pela autora (2024).

5 CONCLUSÃO

Os robôs têm sido amplamente utilizados em diversas áreas, com destaque para os robôs móveis, empregados em aplicações como logística industrial, onde transportam mercadorias de maneira ágil, e na entrega de produtos. Com o aumento das demandas, diversos estudos e pesquisas vêm sendo desenvolvidos para criar novas interfaces de controle. Contudo, as interfaces tradicionais de controle robótico exige conhecimento técnico prévio para sua utilização. Diante da crescente interação, direta e indireta, entre humanos e robôs, torna-se essencial a criação de interfaces que permitam um controle mais humanizado, sem a necessidade de formação técnica especializada.

Nesse contexto, foi desenvolvida uma interface humano-robô que possibilita controlar os movimentos básicos do LEGO Mindstorms de maneira natural para o ser humano, permitindo que o usuário interaja com o sistema de forma compatível com seus próprios movimentos e capacidades, utilizando gestos manuais. Os objetivos estabelecidos para o desenvolvimento deste sistema foram alcançados com sucesso, incluindo o estudo de bibliotecas capazes de mapear mãos e reconhecer faces que se deu por meio das bibliotecas de código aberto como *OpenCV* e *MediaPipe*. O desenvolvimento de um sistema de reconhecimento facial e a implementação de uma solução para verificação de permissões de acesso por usuário foi possível com as tecnologias de visão computacional e ML. Os resultados obtidos evidenciam o potencial desta interface para o avanço das tecnologias de interação humano-robô, especialmente em contextos educacionais e industriais, ao permitir métodos de controle humanizados. A integração de técnicas de aprendizado supervisionado resultou em um sistema adaptável a diferentes dispositivos.

Este estudo, portanto, oferece uma base para futuros desenvolvimentos que possam expandir as funcionalidades e aplicações das interfaces humano-robô, incorporando tecnologias de inteligência artificial mais avançadas e integração com sensores. Em síntese, a proposta apresentada contribui significativamente para tornar as interações entre humanos e robôs mais naturais e acessíveis.

REFERÊNCIAS

ABB. **ABB lança a plataforma de controle de robótica de última geração OmniCore**. 2024. Acesso em: 25 nov. 2024. Disponível em: https://new.abb.com/news/pt-br/detail/116279/abb-lanca-a-plataforma-de-controle-de-robotica-de-ultima-geracao-omnicore. Citado na página 19.

BRADSKI, Gary; KAEHLER, Adrian. **Learning OpenCV: Computer Vision with the OpenCV Library**. 1st. ed. [S.l.]: O'Reilly Media, 2008. Citado na página 24.

CAMPOS, Rafael; SANTOS, Eduardo. Controle de braço robótico via interface natural de usuário com sensor Kinect. In: **Anais do 14º Simpósio Brasileiro de Automação Inteligente**. Ouro Preto: Galoá, 2019. Citado na página 20.

CUI, Jianwei et al. Iot wheelchair control system based on multi-mode sensing and human-machine interaction. **Micromachines**, v. 13, n. 7, 2022. Disponível em: https://www.mdpi.com/2072-666X/13/7/1108. Citado na página 20.

EDUCATION, Hack. **LEGO Mindstorms History**. 2024. Disponível em: https://hackeducation.com. Acesso em: 04 dez. 2024. Citado na página 17.

EDUCATION, LEGO. **LEGO Mindstorms EV3 Overview**. 2024. Disponível em: https://education.lego.com. Acesso em: 04 dez. 2024. Citado na página 17.

FERREIRA, Liliana et al. Desenvolvimentos recentes na interface humano-robô baseada em linguagem falada do robô Carl. **Eletrónica e Telecomunicações**, v. 4, n. 1, p. 51–63, 2003. Citado na página 19.

GOOGLE. **Guia de detecção de pontos de referência de poses**. 2024. https://ai.google.dev/edge/mediapipe/solutions/vision/pose_landmarker?hl=pt-br. Acesso em: 29 out. 2024. Citado na página 23.

GOOGLE. **Guia de soluções do MediaPipe**. 2024. https://ai.google.dev/edge/mediapipe/solutions/guide?hl=pt-br. Acesso em: 29 out. 2024. Citado na página 23.

IFR, International Federation of Robotics. **World Robotics 2021: Industrial Robots**. 2021. Acesso em: 6 nov. 2024. Disponível em: https://ifr.org/. Citado na página 15.

KELLY, A. **Mobile Robotics: Mathematics, Models, and Methods**. Cambridge: Cambridge University Press, 2020. Citado na página 16.

KOREN, Yoram; GU, Xi; GUO, Weihong. Reconfigurable manufacturing systems: Principles, design, and future trends. **Frontiers of Mechanical Engineering**, Springer, v. 13, p. 121–136, 2018. Citado na página 13.

LANG ev3dev. **Networking with RPyC**. 2024. Disponível em: https://ev3dev-lang.readthedocs.io/projects/python-ev3dev/en/latest/rpyc.html#networking. Acesso em: 04 dez. 2024. Citado na página 32.

LANG ev3dev. **RPyC - Remote Python Call**. 2024. https://ev3dev-lang.readthedocs.io/projects/python-ev3dev/en/ev3dev-jessie/rpyc.html. Acesso em: 04 dez. 2024. Citado na página 32.

LIENHART, Rainer; MAYDT, Jürgen. An extended set of Haar-like features for rapid object detection. In: **Proceedings of the International Conference on Image Processing**. [S.l.: s.n.], 2002. Citado na página 26.

MAHESH, Batta. Machine learning algorithms - a review. **International Journal of Science and Research (IJSR).[Internet]**, v. 9, n. 1, p. 381–386, 2020. Citado na página 25.

MAIA, Deborah Christina. **Gesture Locomotion Controller for Mobile Robots**. 2024. Disponível em: https://github.com/MOBILAB-UDESC/gesture-control.git. Citado na página 32.

MAIA, Deborah Cristina. **LEGO Mindstorms - Controle de movimento para frente LEGO Mindstorms - Controle de movimento para frente**. 2024. https://youtube.com/shorts/40em4yMQIQA. Acesso em: 3 dez. 2024. Citado na página 32.

MAIA, Deborah Cristina. **LEGO Mindstorms - controle de movimento para trás**. 2024. https://youtube.com/shorts/vorxjWKxYuo?feature=share. Acesso em: 3 dez. 2024. Citado na página 32.

MAIA, Deborah Cristina. **LEGO Mindstorms - Controle de rotação no sentido anti-horário**. 2024. https://youtube.com/shorts/uuvSyV9K6T8. Acesso em: 3 dez. 2024. Citado na página 32.

MAIA, Deborah Cristina. **LEGO Mindstorms - Controle de rotação no sentido horário**. 2024. https://youtube.com/shorts/w2z4BSqPH5Q?feature=share. Acesso em: 3 dez. 2024. Citado na página 32.

PALAR, Piatan; OLIVEIRA, André. Interface de controle por métodos de autonomia adaptável deslizante para robôs de inspeção. In: **Anais Estendidos do XIII Simpósio Brasileiro de Robótica e XVIII Simpósio Latino Americano de Robótica**. Porto Alegre, RS, Brasil: SBC, 2021. p. 10–21. Citado na página 19.

RESENDE, Raoni Maira. **Desenvolvimento de uma interface humano-robô utilizando visão computacional e sistemas a eventos discretos**. Dissertação (Mestrado) — Universidade Federal de Minas Gerais, 2006. Acesso em: 23 nov. 2024. Disponível em: http://hdl.handle.net/1843/RVMR-6WDNLV. Citado 2 vezes nas páginas 19 e 20.

SHINDE, Pramila P.; SHAH, Seema. A review of machine learning and deep learning applications. In: **2018 Fourth International Conference on Computing Communication Control and Automation (ICCUBEA)**. [S.l.: s.n.], 2018. p. 1–6. Citado na página 25.

SICILIANO, B.; KHATIB, O. **Springer Handbook of Robotics**. [S.l.]: Springer, 2008. Citado na página 13.

UNIVERSAL ROBOTS. **O uso dos robôs de Lego como ferramenta de incentivo à robótica e engenharia**. 2024. Online. Acesso em: 21 nov. 2024. Disponível em: https://www.universal-robots.com/br/blog/o-uso-dos-rob%C3%B4s-de-lego-como-ferramenta-de-incentivo-%C3%A0-rob%C3%B3tica-e-engenharia/. Citado na página 12.

VIOLA, P.; JONES, M. Rapid object detection using a boosted cascade of simple features. In: **IEEE Conference on Computer Vision and Pattern Recognition**. [S.l.]: IEEE, 2001. p. 511–518. Disponível em: https://ieeexplore.ieee.org/document/990517. Citado na página 26.

VOLKSWAGEN. **Golf-R Touch: Volkswagen muestra su sistema de control por gestos para mejorar la interacción y el manejo del automóvil**. 2015. Online. Acesso em: 23 nov. 2024. Disponível em: https://www.volkswagen.es/comunicacion/golf-r-touch-volkswagen-muestra-su-sistema-de-control-por-gestos-para-mejorar-la-interaccion-y-el-manejo-del-automovil/. Citado na página 21.

WEINERT, Horst; PENSKY, Dirk. Mobile robotics in education and student engineering competitions. In: **IEEE Africon 2011**. [S.l.]: IEEE, 2011. p. 1–5. Citado na página 16.

WURMAN, Peter; D'ANDREA, Raffaello; MOUNTZ, Mick. Coordinating hundreds of cooperative, autonomous vehicles in warehouses. **AI Magazine**, v. 29, n. 1, p. 9–20, 2008. Citado na página 16.

ÜÇGÜL, M. History and educational potential of lego mindstorms NXT. **Mersin Üniversitesi Eğitim Fakültesi Dergisi**, v. 9, n. 2, p. 127–137, 2013. Citado 2 vezes nas páginas 17 e 18.