Artigo recebido em 18/04/2022.
Publicado em 28/07/2023.

Revista SODEBRAS – Volume 18
N° 211 – JULHO/ 2023

DOI: http://doi.org/10.29367/issn.1809-3957.18.2023.211.21

VISÃO COMPUTACIONAL APLICADA À UM PROTÓTIPO DE MÃO ROBÓTICA UTILIZANDO IoT

COMPUTER VISION APPLIED TO A PROTOTYPE ROBOTIC HAND USING IOT

Kauê Martins de Souza[1]
Jadson de Souza Maciel[2]
Moisés Bastos[3]
Almir Kimura Junior[4]

Resumo - A aplicação de visão computacional é a ciência responsável no desenvolvimento de algoritmos que permitem máquinas a processar e analisar imagens ou vídeos digitalizados, uma das práticas está no reconhecimento de movimentos humanos. Este artigo propõe a analise do movimento de uma mão humana através de uma câmera, fazendo com que um protótipo de mão robótica replique os movimentos através das práticas de PDI. O desenvolvimento foi realizado utilizando a linguagem Python e as bibliotecas Mediapipe e OpenCV, especializadas em processamento digital de imagens. A comunicação entre o computador e o microcontrolador embarcado na mão robótica é realizada através do protocolo MQTT, permitindo uma resposta rápida da mão robótica aos movimentos dos dedos. Os resultados constataram eficiência na técnica de comunicação utilizada e eficácia nos processos de PDI.

Palavras-chave: Visão Computacional. Python. Mão Robótica. PDI (Processamento digital de imagem).

Abstract – The application of computer vision is the science responsible for the development of algorithms that allow machines to process and analyze digitalized images or videos, one of the practices being human motion recognition. This article proposes the analysis of the movement of a human hand through a camera, making a prototype of a robotic hand replicate the movements through digital image processing practices. The development was carried out using the Python language and the Mediapipe and OpenCV libraries, specialized in digital image processing. Communication between the computer and the microcontroller embedded in the robotic hand is carried out through the MQTT protocol, allowing for a quick response of the robotic hand to finger movements. The results found efficiency in the communication technique used and effectiveness in the digital image processing processes.

Keywords: Computer Vision. Python. Influent Robotic Hand. PDI (Processing Digital Images).

[1] Graduando em Engenharia de Controle e Automação (UEA). Contato: kmds.eai20@uea.edu.br.
[2] Graduando em Engenharia de Controle e Automação (UEA). Contato: sam.eai18@uea.edu.br.
[3] Professor do Departamento de Controle e automação (UEA). Contato: mpbastos@uea.edu.br.
[4] Professor do Departamento de Controle e automação (UEA). Contato: akimura@uea.edu.br.

I. INTRODUÇÃO

Segundo Backes (2016), podemos definir visão computacional como a área de estudo que tenta repassar para máquina a capacidade da visão humana. Quando falamos de visão não estamos nos referindo apenas ao ato de captar imagens. Apesar de essa capacidade ser importante, ela é apenas o início de um processo muito mais complexo. A visão computacional é a ciência responsável pela forma como um computador enxerga o meio à sua volta, obtendo informações por câmeras, sensores e outros dispositivos. Pode-se afirmar que esta ciência vem ganhando cada vez mais destaque devido sua grande versatilidade e por possuir técnicas variadas que servem para vários tipos de situações.

O Processamento Digital de Imagem (PDI) trata-se de um ramo da visão computacional onde são utilizados técnicas para a análise de dados multidimensionais onde a entrada e a saída sejam imagens. De acordo com Gonzales e Woods (2000), o interesse nos métodos de processamento de imagens digitais decorre de duas áreas principais de aplicação: melhoria de informação visual para interpretação humana e o processamento de dados de cenas para percepção automática através de máquinas". Pode-se observar uma aplicação análoga onde um braço robótico realiza movimentos semelhantes aos de uma mão real por meio de câmeras.

Em face ao exposto, esta pesquisa trata-se de um aprimoramento de um projeto inicial onde a mão robótica, Martins (2022), deveria reagir a botões, realizando assim, movimentos pré-definidos via *software*. Tem-se o objetivo de realizar o desenvolvimento de um método que analisa através de técnicas de PDI os movimentos de uma mão por meio de uma câmera, consiga definir as posições dos dedos das mãos, e replique estes movimentos em um protótipo de mão robótica. Para essa comunicação, usou-se o protocolo MQTT, segundo Jaffey (2014), a arquitetura do protocolo MQTT é baseada no modelo cliente/servidor, onde cada sensor é um cliente e se conecta a um servidor, conhecido como broker, via TCP. Este protocolo foi responsável por estabelecer uma comunicação IoT entre o computador e o microcontrolador ESP32 permitindo que este ajuste de forma rápida a angulação de cada servo motor da mão robótica.

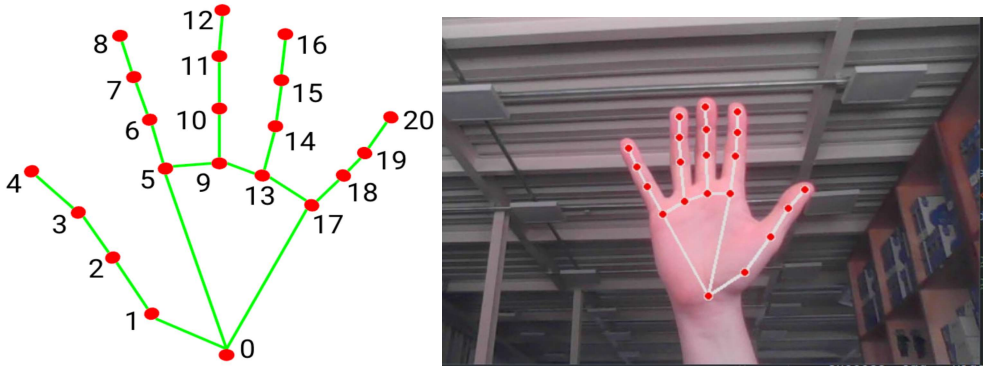
II. METODOLOGIA

2.1 – Desenvolvimento de aplicação em PDI

Para o desenvolvimento da aplicação, este projeto utilizou a linguagem de programação Python e as bibliotecas *Mediapipe* e *OPENCV* para realizar o processamento digital das imagens. O *MediaPipe* trata-se de uma plataforma *open source* que utiliza *Machine Learning* para as mais diversas aplicações como detecção 3D de objetos e detecção e segmentação de áreas de interesse que neste caso são os dedos. Segundo Lugaresi (2019), com o MediaPipe, um pipeline de percepção pode ser construído como um gráfico de componentes modulares, incluindo, por exemplo, modelos de inferência e funções de processamento de mídia. Detecção de objetos. Dados sensoriais, como fluxos de áudio e vídeo, entram no gráfico e descrições percebidas, como resultados de detecção de objetos ou anotações de pontos de referência faciais, saem do gráfico.

Para a aplicação em mãos, foi utilizado uma pipeline de *machine learning* com diversos modelos trabalhando em paralelo (mediapipe,2020). Na prática, existe um modelo de referência que opera nas imagens obtidas e retorna 21 pontos 3D em diferentes partes da mão como pode-se observar na Figura 1.

Figura 1 – Indicações das LandMarks (Pontos de referência) para cada articulação da mão



Fonte: GitHub, 2022.

Para a obtenção dos pontos, foram utilizadas cerca de 30 mil imagens reais com esses 21 pontos feitos manualmente. Estes foram os responsáveis pelo algoritmo encarregado de informar se um dedo está levantado ou abaixado.

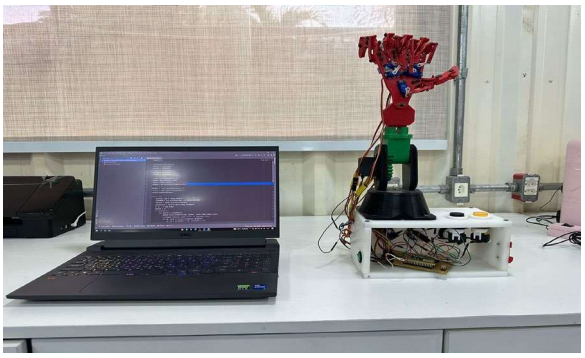
Inicialmente, a ideia seria embarcar o processamento digital em um *Raspberry PI*, porém, devido às limitações do projeto, foi utilizado outra alternativa. O microcontrolador ESP32 não possui poder de processamento suficiente e nem portabilidade para executar o *OPENCV*, sendo assim, optou-se por implementar o *OPENCV* no computador via *Python*. A sintaxe do Python é simples e fácil de usar. Ele enfatiza a legibilidade e usa palavras-chave padrão. OpenCV Python fornece uma infraestrutura comum para aplicações de visão computacional e para acelerar o uso de percepção da máquina nos produtos comerciais. O OpenCV Python se concentra principalmente no processamento de imagens em tempo real (Mustaffa, 2017).

2.2 – Comunicação "IoT"

Todo objeto se comunica e se associa de alguma forma. Por exemplo, a passagem do tempo é essa forma comunicativa de troca com outros objetos do ambiente e, por isso, ela vai envelhecendo (a relação entre o objeto sensorial e sua qualidade). Mas, com a IoT, trata-se agora de outra forma de comunicação das coisas: uma comunicação informacional em rede por protocolos de conexão seguindo a algoritmos e performances criando delegações, mediações, intermediações e estabilizações nas associações, (Lemos, 2012). Devido o computador não possuir nenhum tipo de conexão física com o protótipo, escolheu-se utilizar o protocolo MQTT (*Message Queuing Telemetry Transport*) por se tratar de uma via de comunicação entre dispositivos confiável e bastante funcional.

O MQTT foi criado em 1999 pela IBM, sua proposta é atuar em locais com baixa largura de banda e em dispositivos que possuem memória e processamento limitados (Piper, 2013). Este protocolo possui um funcionamento simples baseado em publicador, *broker* e assinante. O *broker* trata-se de um servidor na nuvem que irá ser responsável por permitir a comunicação ocorrer. No *Python*, foi utilizada a biblioteca *Paho.MQTT* que é a biblioteca do protocolo MQTT mais utilizada para esse sistema. Para o Arduino, foi utilizado a biblioteca *PubSubClient*. Na figura 2, é possível observar a ausência de cabos entre o computador e o protótipo.

Figura 2 – Mão Robótica realizando comunicação via protocolo MQTT



Fonte: Autores (2022).

2.3 – Algoritmo de Comparação no "Python"

O reconhecimento de gestos de mão pode ser usado para aprimorar interação com o computador sem depender da entrada tradicional de dispositivos como teclado e mouse (Jagdish, 2010). No Python, foi necessário criar um algoritmo para identificar se os dedos estavam levantados ou abaixados. Para fazer isso, devemos levar em conta as coordenadas X e Y do vídeo. Os principais pontos são os que estão nas pontas e na base dos dedos, pois estas serão as referências. Caso o valor da posição da ponta de um dedo esteja menor que o da base, o dedo estaria abaixado e vice-versa. Uma parte deste algoritmo pode ser observado na figura 3.

Figura 3 – Algoritmo de identificação da posição dos dedos

```
if pontos[8][1] > pontos[6][1]:
    # print('indicador abaixado')
     client.publish(motor2, 85)
if pontos[8][1] < pontos[6][1]:
    # print('indicador levantado')
     client.publish(motor2, 0)
-----------------------------------
if pontos[12][1] > pontos[10][1]:
    # print("meio abaixado")
     client.publish(motor3, 80)
if pontos[12][1] < pontos[10][1]:
     #print("meio levantado")
     client.publish(motor3, 200)
```

Fonte: Autores (2022).

Para o dedão foi desenvolvida uma lógica diferente pois este não se abaixa da mesma forma que os outros quatro dedos. Para que o dedão seja considerado levantado, o valor da componente Y da ponta do dedão deve ser menor que a componente Y da base do dedo do meio.

Quando uma mão é detectada e o algoritmo faz sua lógica de comparação, há um *delay* de 3 segundos que foi implementado com o intuito de evitar o envio constante de pacotes ao broker que consequentemente iria resultar em valores errados para os servos.

III. RESULTADOS

Após o término do projeto, foi realizado um teste de eficácia com 50 tentativas. Pode-se observar na tabela 1 que mostra os resultados obtidos com cada motor, nota-se que o dedo polegar teve uma taxa de erro maior que ocorreu devido sua lógica diferente. Em todos os casos onde houve algum tipo de erro, após o delay mencionado anteriormente, o dedo da mão robótica se ajustava rapidamente. Também foi publicado um vídeo no Youtube (https://youtu.be/N9kjUHI6KnY) que mostra a eficiência do projeto.

Tabela 1 – Valores obtidos durante os testes.

Dedos	Acertos Imediatos	Erros imediatos
Polegar	45	5
Indicador	49	1
Médio	50	0
Anelar	50	0
Mínimo	49	1

Fonte – Autoral (2022)

IV. CONCLUSÃO

Este artigo apresentou o desenvolvimento de uma aplicação de visão computacional utilizando PDI com o objetivo de movimentar dedos remotamente de um protótipo de mão robótica. O MQTT permite uma comunicação rápida e eficiente entre dispositivos e sistemas, enquanto o processamento digital de imagens fornece a capacidade de manipular e interpretar dados de imagem de maneira eficaz. Ao juntar essas duas tecnologias, é possível criar soluções robustas e escaláveis para aplicações em tempo real, como monitoramento de segurança, análise de tráfego, entre outras. A utilização de MQTT e processamento digital de imagens é um exemplo de como a tecnologia pode ser usada de forma inovadora para solucionar desafios complexos e transformar a forma como trabalhamos e nos relacionamos com o mundo ao nosso redor.

Aprimoramento do PDI para análise de rotações no braço completo e envio de informações para a mão robótica, bem como a utilização de um Raspberry para evitar a dependência de um computador externo, são oportunidades futuras de melhoria.

V. REFERÊNCIAS

AYACHE, Nicholas. Medical computer vision, virtual reality and robotics. **Image and vision computing**, v. 13, n. 4, p. 295-313, 1995.

BACKES, André Ricardo; JUNIOR, Sá; DE MESQUITA, Jarbas Joaci. **Introdução à visão computacional usando Matlab**. Alta Books Editora, 2016.

Google. (2021). MediaPipe Hands [Página de soluções]. GitHub. https://google.github.io/mediapipe/solutions/hands.html

GONZALES, Rafael C.; WOODS, Richards E. Processamento de Imagens Digitais. São Paulo. Edgard Blücher Ltda. 2000.

HOWSE, Joseph. **OpenCV computer vision with python**. Birmingham: Packt Publishing, 2013.

I. B. Mustaffa and S. F. B. M. Khairul, "Identification of fruit size and maturity through fruit images using OpenCV-Python and Rasberry Pi," *2017 International Conference on Robotics, Automation and Sciences (ICORAS)*, Melaka, Malaysia, 2017, pp. 1-3, doi: 10.1109/ICORAS.2017.8308068.

JAFFEY, Toby. MQTT and CoAP, IoT protocols. 2014. Disponível em: <http://www.eclipse.org/community/eclipse_newsletter/2014/february/article2.php>. Acesso em 08 de fevereiro. 2023.

LEMOS, André. A comunicação das coisas. Internet das coisas e teoria ator-rede. **SIMSOCIAL: CYBER-ARTE-CULTURA**, v. 2, 2012.

LUGARESI, Camillo et al. Mediapipe: A framework for perceiving and processing reality. In: **Third Workshop on Computer Vision for AR/VR at IEEE Computer Vision and Pattern Recognition (CVPR)**. 2019.

MARTINS, Ismael Rodrigues; ZEM, José Luís. Estudo dos protocolos de comunicação MQTT e COaP para aplicações machine-to-machine e Internet das coisas. 2015.

SOUZA, K. M. ; SILVA, E. T. S. ; MOTA, S. A. ; BASTOS, M. ; JUNIOR, A. K. ; COSTA, M. V. S. . DESENVOLVIMENTO DE UM PROTÓTIPO EDUCACIONAL DE BAIXO CUSTO (MÃO ROBÓTICA). SODEBRÁS, v. 17, p. 08-13, 2022.

PIPER, Andy. MQTT Wiki. 2 dez. 2013. Disponível em: <https://github.com/mqtt/mqtt.github.io/wiki/>. Acesso em 08 fevereiro. 2023.

RAHEJA, Jagdish Lal et al. Real-time robotic hand control using hand gestures. In: **2010 Second International Conference on Machine Learning and Computing**. IEEE, 2010. p. 12-16.

VI. AGRADECIMENTOS

O presente artigo é decorrente do projeto de Pesquisa e Desenvolvimento (P\&D) Projeto Samsung Ocean 2.0, que conta com financiamento da Samsung, usando recursos da Lei de Informática para a Amazônia Ocidental (Lei Federal nº 8.387/1991), estando sua divulgação de acordo com o previsto no artigo 39.º do Decreto nº 10.521/2020.

VII. COPYRIGHT