Desenvolvimento de ferramenta de contagem de pessoas com esp32-cam e yolov3-tiny para controle inteligente de ar-condicionado

Isaac Barros Silva, Valdez Aragão de Almeida Filho, Diego de Azevedo Gomes e Diorge de Souza Lima.

***Resumo*— Este artigo apresenta o desenvolvimento de um algoritmo baseado em YOLOv3-tiny para detecção de pessoas em imagens capturadas por ESP32-CAM. O pipeline engloba pré-processamento, inferência e pós-processamento para gerar o campo people_count em JSON. Esse dado aciona, de forma reativa, centrais de ar-condicionado via ESP32, sem polling. Testado em diversas condições, o sistema mostrou baixa latência, operação estável e viabilidade para otimização energética em salas de aula.**

Palavras-Chave— ESP32-CAM; YOLOv3-tiny; Flask; Visão Computacional; IoT; Eficiência Energética.

***Abstract*— This paper presents the development of an algorithm based on YOLOv3-tiny for person detection in images captured by an ESP32-CAM. The pipeline includes preprocessing, inference, and post-processing to output a people_count field in JSON. This data reactively triggers air-conditioning units via an ESP32, eliminating polling. Tested under varied conditions, the system demonstrated low latency, stable operation, and feasibility for energy optimization in classroom environments.**

Keywords— ESP32-CAM; YOLOv3-tiny; Flask; Computer Vision; IoT; Energy Efficiency.

I. INTRODUÇÃO

Os sistemas de climatização são responsáveis por parcela significativa do consumo energético em edificações comerciais e educacionais. Estudos indicam que tais sistemas podem representar até 40% da demanda elétrica total em salas de aula e escritórios no Brasil, gerando impactos econômicos e ambientais consideráveis [1]. Nesse contexto, estratégias capazes de ajustar dinamicamente o funcionamento do ar-condicionado conforme a ocupação real dos ambientes surgem como alternativas promissoras para reduzir custos e emissões de gases de efeito estufa.

Tradicionalmente, detectores de presença baseados em sensores PIR (Passive Infrared) são utilizados para acionar automaticamente os sistemas de climatização. Embora apresentem baixo custo e consumo reduzido, esses sensores identificam apenas variações pontuais de radiação infravermelha, falhando na detecção de ocupantes estáticos ou posicionados longe do dispositivo. No trabalho de [2], foi desenvolvido um protótipo com Arduino, sensor PIR e DHT11, que alcançou até 92% de economia diária, mas permaneceu limitado à simples presença ou ausência de indivíduos, sem quantificar o número de ocupantes.

Com o crescimento exponencial de conteúdos visuais — imagens estáticas, vídeos e transmissões em tempo real —, técnicas de visão computacional tornaram-se essenciais para a contagem de pessoas em cena [3]. Arquiteturas como o YOLOv3-tiny integram detecção e classificação de objetos em um único estágio, oferecendo baixa latência e *footprint* enxuto, o que as torna adequadas para dispositivos embarcados como o módulo ESP32-CAM [4]. Assim, supera-se a limitação dos sensores PIR ao estimar a quantidade real de ocupantes por meio de processamento de imagem.

Neste artigo, propomos um pipeline integrado que combina: captura de imagens pelo ESP32-CAM; inferência de um modelo YOLOv3-tiny em um servidor Flask; e acionamento reativo de unidades de ar-condicionado por meio de um segundo ESP32, sem necessidade de *polling*. Apresentamos detalhes da implementação no sistema embarcado e no backend em Python; avaliamos o desempenho em termos de latência e acurácia sob diferentes condições; e demonstramos um controle inteligente que ajusta o climatizador conforme a contagem de pessoas, evidenciando a viabilidade para otimização energética em salas de aula.

II. REFERENCIAL TEÓRICO

A. *Arquiteturas de Detecção de Objetos e a Evolução do YOLO*

O YOLO (You Only Look Once) unifica localização e classificação em uma única etapa, ajustando caixas-âncora pré-definidas às regiões de interesse e atribuindo classes com baixa latência [4]. A versão v3-tiny empregada neste trabalho opera em três escalas: em uma escala mais grossa, o modelo detecta objetos maiores; em escalas mais finas, percebe objetos menores. Além disso, a utilização de âncoras pré-definidas aprimora a detecção de indivíduos próximos ou distantes e mantém alta acurácia em cenários densos, como salas de aula [5].

B. *Python, Flask e a Implantação de Modelos de Visão Computacional*

Optou-se pelo Flask devido à sua leveza e flexibilidade para criar *endpoints* HTTP que recebem imagens do ESP32-CAM e retornam um objeto *JSON* com a contagem de pessoas. A inferência é realizada em um único processo Python, usando o módulo DNN do OpenCV, TensorFlow ou PyTorch, o que reduz latência e *overhead* de processamento [6].

C. *ESP32: Microcontrolador para IoT Embarcado*

O ESP32 é um microcontrolador compacto e de baixo custo, com Wi-Fi integrado e poder de processamento [7] suficiente para capturar e enviar imagens, além de controlar periféricos como LEDs ou relés. Ao reunir tudo em um único módulo, ele simplifica o circuito, reduzindo componentes extras e facilitando a prototipagem. Além disso, o suporte a 4 MB de PSRAM (no módulo ESP32-CAM) permite o buffer de imagens e o controle de sensores e atuadores. Foi escolhido o ESP32 como cérebro do sistema por sua capacidade de executar pilhas de rede, gerenciar captura e transmissão de imagens e, simultaneamente, controlar pinos de I/O (por exemplo, para simular o acionamento

do ar-condicionado), garantindo um dispositivo único, flexível e de rápida prototipagem para ambientes acadêmicos.

D. Módulo de Câmera OV2640

Optou-se pelo sensor OV2640 de 2 MP (Full HD 1080p) com lente de 160° e capacidade de visão noturna infravermelha (850 nm) porque ele oferece cobertura ampla do ambiente e mantém a operação mesmo em baixa luminosidade. Além disso, sua integração direta com o ESP32-CAM simplifica o circuito, reduz custos e garante imagens de qualidade suficiente para a contagem de pessoas em tempo quase real, atendendo às necessidades do projeto sem complexidade desnecessária.

III. METODOLOGIA

O sistema inicial, descrito por [2], apoiava-se unicamente em um sensor PIR para detectar presença em sala de aula, conforme ilustrado na [2, Fig. 1]. Como detector de movimento passivo, o PIR identifica apenas variações de radiação infravermelha associadas ao deslocamento dos ocupantes, falhando quando estes permanecem estáticos ou distantes do sensor. Nesses casos, era necessário um gesto amplo para que o PIR "visse" alguém, levando a leituras equivocadas de sala vazia, mesmo quando havia pessoas presentes.

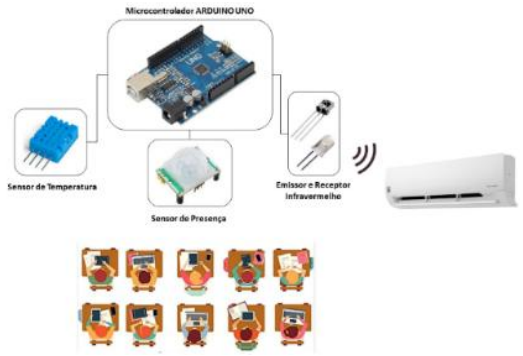

Fig. 1. Proposta do primeiro protótipo [2]

Para superar essas limitações, substituímos o Arduino Uno por um módulo ESP32-CAM equipado com a câmera OV2640. As imagens capturadas são enviadas via Wi-Fi a um servidor Python implementado em Flask, que cria o diretório de armazenamento quando necessário, valida os formatos de arquivo e disponibiliza rotas para upload de imagens, inferência e visualização de resultados.

o servidor, o modelo YOLOv3-tiny é carregado na inicialização. A cada imagem recebida, realiza-se filtragem de detecções de baixa confiança e supressão de sobreposições, obtendo-se a contagem de ocupantes. Os dados são encapsulados em um *JSON* no campo *people_count* e retornados ao cliente. Em paralelo, um segundo ESP32 atua como servidor HTTP leve, monitorando esse *JSON* e, ao detectar variações no número de pessoas, acionando imediatamente seu pino de saída para ligar ou desligar o ar-condicionado (simulado por LEDs), dispensando consultas periódicas (*polling*) e garantindo resposta instantânea

Ao separar as funções de visão computacional e de acionamento, evita-se sobrecarga no ESP32-CAM e assegura-se desempenho máximo de cada dispositivo. O ESP32 de controle pode ser posicionado próximo às unidades de ar-condicionado, melhorando a qualidade de comunicação com emissores infravermelhos. Essa arquitetura permite ajustar a temperatura apenas em resposta a alterações significativas na ocupação, evitando oscilações frequentes que poderiam causar desconforto ou desperdício energético.

A validação do algoritmo envolveu testes em ambientes controlados e com imagens de referência, cobrindo diferentes níveis de iluminação e variações de posição dos ocupantes e sua respectiva contagem tudo isso guardado em uma "galeria". Esse histórico guardado permitiu a calibração do *software* nos limiares de confiança e nos parâmetros de pré-processamento, aumentando a robustez do sistema para uso real em salas de aula. Todo o fluxo de operação está detalhado no fluxograma da [Fig. 2].

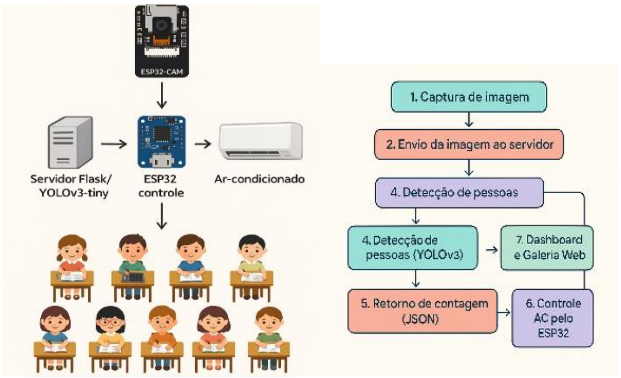

Fig. 2. Fluxograma do sistema. Figura do autor

IV. RESULTADOS

O módulo ESP32-CAM demonstrou-se robusto e eficiente na captura e transmissão de imagens. Durante todos os testes, o firmware executou continuamente o ciclo de inicialização da câmera, aquisição do frame JPG e envio ao servidor Flask via HTTP POST em aproximadamente 150 ms por iteração. Esse desempenho consistente garantiu que a funcionalidade de capturar imagens e enviar ao servidor estivesse plenamente implementada, sem perda de pacotes, mesmo em conexões Wi-Fi de sinal mediano.

sistema encontra-se em fase de testes laboratoriais e, embora ainda não tenha sido implantado em sala de aula, já apresenta um fluxo completo de captura, envio, processamento e retorno de informações. Ao submeter imagens pela página de envio manual, conforme ilustrado na [Fig. 3], o servidor Flask responde, em média, em 200 ms com um JSON contendo o número estimado de pessoas. As Figuras 4 e 5 apresentam, respectivamente, o primeiro e o segundo retornos do servidor, mostrando a imagem enviada e o *JSON* recebido. Esse desempenho, ainda que inferior ao desejável em ambiente de produção, demonstra a viabilidade de operação quase em tempo real e permite iterações rápidas durante a fase de ajuste.

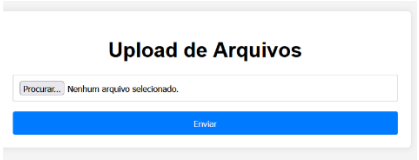

Fig. 3. Pagina de envio manual. Figura do autor

Isaac Barros SIlva, Estudante, Unifesspa, Marabá-PA, e-mail: Isaac.barros@unifesspa.edu.br; Valdez Aragão de Almeida Filho, Faculdade de Engenharia Elétrica/Instituto de Geociências e Engenharias, Unifesspa, Marabá-PA, e-mail valdez.filho@unifesspa.edu.br; Diego de Azevedo Gomes, Faculdade de Engenharia Elétrica/Instituto de Geociências e Engenharias, Unifesspa, Marabá-PA, e-mail: diagomes@unifesspa.edu.br; Diorge de Souza Lima, Faculdade de Engenharia Elétrica/Instituto de Geociências e Engenharias, Unifesspa, Marabá-PA, e-mail: diorgelima@unifesspa.edu.br. Este trabalho foi parcialmente financiado pela FAPESPA (09/2024).

Fig. 4. Primeira resposta do servidor ao envio via ESP32-CAM. Figura do autor

Fig. 5. Segunda resposta do servidor. Figura do autor

Nos testes iniciais, que incluíram fotografias tiradas em diferentes salas de aula da Unifesspa tiradas com o ESP32-CAM e imagens coletadas de bancos públicos na internet, observou-se que o modelo tem um erro considerável e duas situações: situações, quando a pessoas muito distantes da câmera e indivíduos parcialmente sobrepostos. Na [Fig. 4] por exemplo, há três pessoas na cena, mas o modelo detectou quatro; já na [Fig. 5], das 47 pessoas presentes, foram detectadas apenas onze. Esse desvio deve-se a fatores como variações de iluminação, ângulos de captura e limitações do modelo YOLOv3-tiny em distinguir indivíduos parcialmente ocluídos.

A galeria de imagens foi fundamental para estruturar a análise de erros, pois permitiu comparar lado a lado cada foto enviada e as delimitações geradas pelo algoritmo, identificando padrões de falha, por exemplo: turmas próximas ao fundo ou iluminação lateral intensa.

Além da detecção, validou-se a comunicação com o segundo ESP32, responsável pelo controle do ar-condicionado e inicialmente representado por LEDs cujo estado (aceso/apagado) é alternado via painel de controle. Essa etapa comprovou a confiabilidade do enlace HTTP e a capacidade do firmware embarcado de interpretar corretamente o *JSON* retornado, precursora para o futuro envio de sinais infravermelhos às unidades de ar-condicionado.

Ressalta-se que, apesar das imprecisões atuais, o sistema já cumpre sua função básica de detecção e controle remoto, servindo como protótipo educacional para estudantes de engenharia elétrica. A existência de páginas dedicadas ao upload de teste e à visualização histórica de imagens enriquece a experiência de depuração e oferece um repositório de dados para estudos posteriores, por exemplo, para re-treinamento de modelos ou ajuste de parâmetros de confiança e supressão de detecções..

Em síntese, o fluxo completo de captura, envio, processamento e resposta está funcional e estabilizado em ambiente de laboratório, ainda que faltem validações em tempo real e a implementação física do atuador. Próximas etapas incluem refinar o pré-processamento (equalização de histograma, correção de exposição), realizar o ajuste fino do detector com imagens do próprio campus e integrar o controle de temperatura real, finalizando assim um ciclo de IoT inteligente para otimização energética em salas de aula.

V. CONCLUSÕES

Os resultados obtidos demonstram que o uso do módulo ESP32-CAM, em conjunto com um servidor Flask que executa inferência via YOLOv3-tiny, forma um sistema viável para a contagem de ocupantes e o acionamento inteligente de unidades de climatização. A latência média de 150 ms na captura e envio dos frames, aliada aos 200 ms de resposta do servidor, comprova a capacidade de operação quase em tempo real, mesmo em redes Wi-Fi com qualidade variável. Dessa forma, valida-se a eficiência do pipeline integrado em ambiente laboratorial.

Apesar das limitações observadas, principalmente nas detecções de indivíduos distantes ou parcialmente ocluídos, o protótipo já cumpre com sua função educacional e tecnológica, oferecendo uma plataforma flexível para estudos em IoT e visão computacional. A galeria de imagens e o histórico de detecções permitiram identificar padrões de erro e calibrar parâmetros, fator essencial para o aprimoramento contínuo do sistema.

Como trabalhos futuros, propõe-se o ajuste fino do modelo com imagens coletadas no próprio campus, a implementação de técnicas avançadas de pré-processamento de imagem e a integração física de atuadores infravermelhos para controle direto das centrais de ar-condicionado. Tais evoluções poderão transformar o protótipo atual em uma solução completa e escalável para a otimização energética em salas de aula e outros ambientes de uso coletivo.

REFERENCIAS

[1] *Empresa de Pesquisa Energética (EPE), *Anuário estatístico de energia elétrica 2016: ano base 2015*, Rio de Janeiro, 2016.*

[2] *I. Barros, V. A. de Almeida Filho, D. A. Gomes, and D. de S. Lima, "Dispositivo autônomo de baixo custo para controle eficiente de equipamentos condicionadores de ar em salas de aula," 2025. [Online]. Available: https://drive.google.com/file/d/17DT1S3cKgMifmkGgESn-w07k3DRF_Bzp/view. [Accessed: 29-Apr-2025].*

[3] *G. Juraszek, "Reconhecimento de produtos por imagem utilizando palavras visuais e redes neurais convolucionais," ResearchGate, 2014, pp. 21–23..*

[4] *J. Redmon and A. Farhadi, "You Only Look Once: Unified, real-time object detection," CV-Foundation, 2016, pp. 780–781.*

[5] *H. Muhammad, "Yolo-v1 to YOLO-v8: the rise of YOLO and its complementary nature toward digital manufacturing and industrial defect detection," *Machines*, MDPI, pp. 3–4, 2023.JURASZEK, G. Reconhecimento de produtos por imagem utilizando palavras visuais e redes neurais convolucionais. ResearchGate, researchgate.net, p. 21–23, 2014.*

[6] *M. Grinberg, *Flask Web Development: Developing Web Applications with Python*, Sebastopol, CA: O'Reilly Media, 2018.*

[7] *[1] M. Makiyama, "Placa ESP32: O que é, para que serve e uso!", Victor Vision, 6 Nov. 2023. [Online]. Available: https://victorvision.com.br/blog/placa-esp32/. [Accessed: 29-Apr-2025].*