Universidade dos Açores

Pós-Graduação em Programação, Robótica e Inteligência Artificial (PRIA)

Relatório de Projeto Experimental

OpenBalance – Plataforma de Equilíbrio com Visão Computacional e Controlo PID

Autor: João Pavão

Docente: Prof. José Cascalho

Ponta Delgada, 14 de junho de 2025

Resumo

Este relatório apresenta o desenvolvimento do projeto OpenBalance, uma plataforma robótica capaz de equilibrar uma bola através do controlo da inclinação da sua superfície. O sistema recorre a visão computacional para localizar a bola em tempo real e aplica algoritmos de controlo PID para manter a sua posição centrada. Desenvolvido no contexto da unidade curricular de Laboratório de Aplicações em Robótica e Aprendizagem (PRIA), este projeto combina conceitos de automação, controlo e inteligência artificial aplicada, com uma forte vertente pedagógica e open-source. Utiliza componentes como Arduino, servomotores, driver PCA9685 e uma webcam USB, integrados por uma interface gráfica construída em Python com CustomTkinter.

Palavras-chave: Visão Computacional, Controlo PID, Arduino, Robótica Educacional, OpenCV, CustomTkinter

Índice

1. Introdução

O presente relatório documenta o desenvolvimento do projeto **OpenBalance**, uma plataforma robótica experimental concebida no âmbito da unidade curricular de **Laboratório de Aplicações em Robótica e Aprendizagem (PRIA)** da Universidade dos Açores. O sistema tem como principal objetivo demonstrar, de forma visual e interativa, os fundamentos do controlo automático aplicados a um problema clássico: o equilíbrio dinâmico de uma bola sobre uma superfície inclinável.

A ideia central consiste em permitir que uma bola se mantenha centrada numa plataforma bidimensional, cuja inclinação é controlada por dois servomotores, um em cada eixo (X e Y). A posição da bola é monitorizada em tempo real através de uma **webcam USB**, com processamento de imagem realizado em **Python** utilizando a biblioteca **OpenCV**. A decisão sobre os ângulos de inclinação é determinada por algoritmos de **controlo PID (Proporcional–Integral–Derivativo)**, ajustáveis através de uma **interface gráfica interativa** desenvolvida com **CustomTkinter**.

O OpenBalance não é apenas uma implementação técnica; constitui também uma ferramenta pedagógica e de código aberto, pensada para facilitar o ensino e a aprendizagem de conceitos fundamentais de **automação, controlo, robótica e visão computacional**. A modularidade do sistema permite que seja facilmente replicado, adaptado e expandido, por exemplo para sistemas de maior complexidade como plataformas de Stewart (6 graus de liberdade).

Ao longo deste relatório serão apresentados os objetivos do projeto, os componentes utilizados, a estrutura de hardware e software desenvolvida, bem como as estratégias de calibração e os resultados obtidos. Será ainda feita uma reflexão sobre as potencialidades educativas do projeto e as perspetivas futuras para a sua evolução.

2. Objetivos

O projeto **OpenBalance** tem como objetivo principal o desenvolvimento de uma plataforma robótica educativa capaz de demonstrar o funcionamento de sistemas de controlo em tempo real, com particular enfoque no **controlo PID** e na **visão computacional**. Esta plataforma serve como um modelo físico interativo para o estudo de sistemas dinâmicos, permitindo a aplicação prática de conceitos abordados na unidade curricular de **Laboratório de Aplicações em Robótica e Aprendizagem**.

Objetivos específicos:

- **Construir uma plataforma mecânica funcional** que permita a inclinação controlada de uma superfície em dois eixos (X e Y), com servomotores de precisão;
- **Integrar um sistema de aquisição de imagem em tempo real**, utilizando uma webcam USB e a biblioteca OpenCV para deteção da posição de uma bola sobre a plataforma;
- **Implementar algoritmos de controlo PID** para estabilizar a posição da bola no centro da superfície, ajustando dinamicamente os ângulos dos servos;
- **Desenvolver uma interface gráfica interativa (Dashboard)** em Python com CustomTkinter, permitindo ao utilizador ajustar os parâmetros do controlo PID e da deteção de cor (HSV), bem como visualizar o vídeo e interagir com o sistema;
- **Testar e calibrar** o sistema físico e os algoritmos de controlo, otimizando o desempenho para diferentes condições;
- **Documentar o processo de desenvolvimento**, desde a conceção mecânica até à implementação do software, promovendo uma abordagem pedagógica, acessível e reprodutível;
- **Explorar possibilidades de expansão futura**, com destaque para:
    - A **evolução da plataforma para sistemas com três ou mais braços de controlo**, permitindo não só o equilíbrio passivo da bola mas também ações proativas e direcionadas sobre o movimento da mesma;

- A conceção de um **sistema tipo Stewart** com seis graus de liberdade, oferecendo um campo de controlo tridimensional e potencial para simulações avançadas;
- A incorporação de **comportamentos mais interativos**, como a **capacidade de "bater" ou impulsionar a bola intencionalmente** para zonas-alvo, baseando-se em inputs visuais e algoritmos de decisão;
- A **integração com algoritmos de inteligência artificial**, nomeadamente aprendizagem por reforço, para que o sistema aprenda estratégias ótimas de controlo ou interação a partir da experiência acumulada.

Estas perspetivas futuras posicionam o **OpenBalance** como uma plataforma **aberta, escalável e reprodutível**, com potencial para projetos de investigação, ensino avançado e demonstrações interativas em contextos científicos ou educativos. Todo o código-fonte, esquemas eletrónicos, ficheiros de impressão 3D e documentação estão a ser organizados e publicados num **repositório público no GitHub**, promovendo a colaboração, a partilha de boas práticas em robótica educativa e a reutilização do projeto em contextos escolares, académicos ou pessoais. O projeto será disponibilizado sob **licença open source**, incentivando a sua adaptação, evolução e integração em futuras iniciativas.

3. Materiais e Montagem Física

3.1 Materiais Gerais

Bola leve (ex: esférica de pingue-pongue ou borracha macia)

Contraplacado fino (base da plataforma móvel)

Parafusos, porcas e anilhas M3 e M4

Cabos Dupont, terminais e fita dupla face

Tinta preta fosca (para contraste visual da plataforma)

Calha técnica em PVC (para suporte da webcam)

3.2 Eletrónica

Microcontrolador: Arduino UNO (ou compatível)

Módulo controlador de servos: PCA9685 (16 canais, I^2C)

Servomotores: 2x DM996 (15 kg·cm, compatíveis com 6 V)

Fonte de alimentação: Fonte regulável 6 V / 72 W (com bornes)

Webcam USB: Câmera genérica HD (fixada em estrutura superior)

Cablagem adicional: fios, conectores macho-fêmea, ficha de alimentação

3.3 Estrutura Física e Impressão 3D

Braços articulados (x2): Impressos em PLA, com suporte para servos e encaixes de fixação à base

Suportes laterais: Estrutura rígida para montagem vertical dos servos

Pivô central: Elemento circular ou cónico, também em PLA, que permite rotação e estabilidade da plataforma

Encaixes tipo cavilha: Integrados nos braços para evitar deslizamento lateral

Plataforma móvel: Contraplacado pintado de preto com cruz central branca e borda natural visível (~4 mm) para facilitar a deteção por visão computacional

3.4 Montagem e Infraestrutura

Os braços 3D foram aparafusados diretamente aos servos, que por sua vez foram fixados às paredes laterais da base.

A plataforma repousa livremente sobre os braços, encaixando nos pinos e no pivô central para estabilidade e liberdade de movimento.

A webcam foi montada sobre uma calha técnica superior, garantindo uma visão perpendicular e estável de toda a área da plataforma.

O circuito foi montado numa breadboard lateral com ligações ao Arduino e à alimentação externa.

A fonte de alimentação foi cuidadosamente ajustada para fornecer 6 V estáveis e proteger os servos de sobrecargas.

4. Arquitetura do Sistema

O funcionamento do projeto OpenBalance assenta na integração de diversos subsistemas — deteção visual, controlo PID e interface gráfica — que comunicam entre si de forma coordenada. A arquitetura do sistema foi concebida para funcionar em tempo real, garantindo a resposta rápida necessária à estabilização da bola na plataforma.

4.1 Deteção Visual com OpenCV

A deteção da posição da bola é feita com recurso a uma **webcam USB** fixada verticalmente sobre a plataforma. A imagem captada é processada em **Python**, usando a biblioteca **OpenCV**, com o objetivo de localizar a bola em cada frame.

O processo é realizado da seguinte forma:

- **Captura de imagem**: Utiliza-se o cv2.VideoCapture() para capturar imagens da webcam em tempo real.
- **Conversão de cores**: A imagem é convertida do espaço BGR para HSV (cv2.cvtColor) para permitir uma segmentação mais robusta da cor da bola, mesmo com variações de luz.
- **Filtragem por intervalo HSV**: Aplicam-se máscaras (cv2.inRange) com base em valores ajustáveis via interface, permitindo identificar apenas a cor da bola.
- **Tratamento morfológico**: Usa-se erosão e dilatação para remover ruído e obter uma forma limpa.
- **Localização do centroide**: Com cv2.findContours e cv2.moments, calcula-se o centro (x, y) da bola, que representa a sua posição relativa na imagem.
- **Escalonamento para coordenadas da plataforma**: A posição em píxeis é convertida para uma escala proporcional à plataforma, usada no controlo PID.

A plataforma é pintada de preto com uma cruz branca no centro e uma borda natural de madeira visível (~4 mm), o que favorece o contraste visual e melhora a fiabilidade da deteção. Este sistema oferece uma taxa de atualização suficiente para aplicações de controlo em tempo real (≥ 20 FPS em máquinas comuns).

4.2 Controlo PID e Comando da Plataforma

A posição da bola em relação ao centro da plataforma é usada como erro de entrada para dois **controladores PID**, um para cada eixo (X e Y).

O algoritmo calcula a correção necessária para devolver a bola ao centro, ajustando os ângulos dos servos.

- **PID em Python**: O cálculo do PID é realizado diretamente em Python, com fórmulas discretas para as componentes proporcional, integral e derivativa:

$$u(t) = K_p \cdot e(t) + K_i \cdot \sum e(t) \cdot dt + K_d \cdot \frac{de(t)}{dt}$$

  onde e(t) é a distância da bola ao centro nos eixos X ou Y.
- **Conversão para ângulo de servo**: O sinal de controlo resultante do PID é mapeado para um intervalo de ângulos seguros (por exemplo, 80º a 120º para X e 70º a 110º para Y).
- **Envio de comandos ao Arduino**: Os valores de ângulo são enviados via **comunicação serial** para o Arduino, que aciona os servos através do **driver PCA9685**.
- **Arduino como interface de controlo**:
  - Recebe comandos via porta serial (ex: "X105Y95").
  - Usa a biblioteca Adafruit PWM Servo Driver para controlar os servos em PWM de 16 canais.
  - Aplica os ângulos com limites de segurança definidos durante a fase de calibração.

Esta separação de responsabilidades permite que o **controlo lógico e visual seja feito no PC**, enquanto o **controlo físico é delegado ao microcontrolador**, aumentando a robustez e modularidade do sistema.

4.3 Interface Gráfica em Python

A interface gráfica foi desenvolvida com a biblioteca **CustomTkinter**, que permite criar GUIs modernas com tema escuro, sliders estilizados e separação por painéis.
A interface está dividida em três áreas principais:

- **Painel HSV (deteção da bola)**:
  - Sliders para ajuste em tempo real dos limites de Hue, Saturation e Value;
  - Vista prévia da máscara HSV gerada, permitindo afinar a segmentação;
  - Botão para guardar os valores de HSV em ficheiro CSV.
- **Painel PID (controlo da plataforma)**:
  - Sliders para ajuste dos parâmetros Kp, Ki e Kd para cada eixo;
  - Visualização numérica dos valores;
  - Possibilidade de reiniciar os acumuladores do PID;
  - Gravação de presets em CSV para teste posterior.
- **Painel de vídeo e controlo**:
  - Visualização em tempo real do vídeo processado com sobreposição da posição da bola;
  - Botões para iniciar/pausar seguimento, ligar motores e selecionar a porta serial do Arduino;
  - Deteção automática das portas COM disponíveis.

Toda a lógica de controlo, comunicação e processamento está integrada na interface, o que facilita a interação com o sistema e a experimentação com diferentes configurações. A interface foi projetada para ser pedagógica, intuitiva e extensível.

5. Calibração dos Serrvomotores

A calibração dos servomotores é um passo crítico para garantir o bom funcionamento da plataforma móvel do **OpenBalance**, pois define os limites físicos de inclinação e assegura que os movimentos da plataforma correspondem corretamente aos comandos enviados pelo sistema de controlo PID.

Procedimento Inicial de Calibração

Antes de iniciar o controlo automático, foi necessário determinar os **ângulos mínimos e máximos seguros** para cada servo (eixo X e Y), de modo a evitar esforços excessivos, vibrações ou colisões com a estrutura.

1. **Posição neutra (90°)**:
    - Cada servo foi inicialmente posicionado a 90°, correspondente à **posição horizontal da plataforma**.
    - A montagem dos braços 3D foi feita com a plataforma nivelada nesta posição, garantindo simetria.
2. **Limites físicos**:
    - Através de testes manuais e observação, foram definidos intervalos seguros para cada servo:
        - **Eixo X**: aproximadamente **80° a 120°**
        - **Eixo Y**: aproximadamente **70° a 110°**
    - Estes valores dependem da montagem física, da folga mecânica e da rigidez dos materiais.
3. **Estabilização da estrutura**:
    - Para garantir precisão, foram adicionados **encaixes tipo cavilha** nos braços 3D e uma **base côncava** no pivô central, evitando folgas laterais.
    - A plataforma foi construída de modo a não ser muito lisa, aumentando a aderência sem prejudicar o deslizamento suave da bola.

Implementação da Calibração no Código

A calibração foi incorporada diretamente no código Arduino, responsável por acionar os servos via o módulo PCA9685:

- Foram definidos **limites máximos e mínimos** para cada servo, de modo que os valores recebidos do computador fossem automaticamente restringidos dentro da faixa segura;
- Foi criado um modo de teste manual na dashboard, permitindo enviar ângulos específicos e observar o movimento da plataforma em tempo real;
- A lógica de controlo no Arduino converte os ângulos em pulsos PWM compatíveis com os servos DM996, respeitando o mapeamento definido na calibração.

Exemplo de pseudocódigo para mapeamento:

```
if (anguloX > 120) anguloX = 120;
if (anguloX < 80)  anguloX = 80;
```

Validação com Feedback Visual

Durante a calibração, a posição da plataforma foi monitorizada com a webcam instalada na parte superior. Esta abordagem permitiu:

- Ajustar a inclinação máxima sem sair do campo de visão da câmara;
- Garantir que a plataforma se mantinha estável em repouso (90°);
- Verificar que os movimentos em X e Y não causavam oscilações indesejadas ou contacto com os limites físicos.

A calibração adequada dos servos garante não só a **segurança mecânica** do sistema, mas também o **bom desempenho do controlo PID**, pois permite respostas consistentes e previsíveis aos comandos de correção enviados pelo algoritmo.

6. Testes e Resultados Preliminares

Após a montagem física e implementação do software, foi realizada uma bateria completa de testes para validar o comportamento do sistema **OpenBalance** em condições reais de operação. Os testes envolveram a deteção visual da bola, o controlo PID em tempo real e a resposta física da plataforma acionada por servomotores, com resultados globalmente positivos.

Deteção Visual com OpenCV

A deteção da bola revelou-se estável e precisa após o ajuste adequado dos parâmetros HSV na interface gráfica:

- Foram testadas bolas de várias cores e materiais, sendo as de cor **laranja mate** e **amarela** as que ofereceram melhor contraste sobre a plataforma preta;
- A máscara HSV gerada por OpenCV foi eficaz na maioria das condições de iluminação, especialmente sob luz difusa e estável;
- A presença de uma **cruz branca central** e uma **borda natural visível (~4 mm)** permitiu localizar claramente o centro e os limites da plataforma, melhorando a calibração espacial;
- A taxa de atualização de vídeo situou-se entre **20 e 25 FPS**, suficiente para aplicações em controlo dinâmico.

Desempenho do Controlo PID

Com os valores de PID ajustados via sliders, o sistema demonstrou boa capacidade de manter a bola próxima do centro da plataforma:

- A componente proporcional K_p teve maior influência no retorno rápido da bola ao centro;
- O termo derivativo K_d suavizou significativamente os movimentos, reduzindo oscilações;
- O uso controlado da componente integral K_i ajudou a corrigir desvios persistentes sem causar instabilidade;

- A bola foi capaz de recuperar de pequenas perturbações e manteve-se estável em repouso, validando a eficácia do controlo PID implementado.

Comportamento Físico da Plataforma

A estrutura da plataforma e os servomotores apresentaram desempenho satisfatório:

- Os **limites de inclinação previamente definidos** (X: 80°–120°, Y: 70°–110°) mostraram-se adequados, evitando impactos ou movimentos bruscos;
- Os **servos DM996** responderam de forma consistente, com precisão e força suficiente para inclinar suavemente a superfície;
- A **plataforma foi nivelada** com sucesso, e a bola manteve-se centrada durante longos períodos de teste;
- A **webcam montada na calha superior** permitiu uma visão clara e estável de toda a superfície da plataforma.

Registo e Otimização

Todos os testes foram devidamente registados:

- Foram guardados **presets de valores PID e HSV** em ficheiros CSV para reutilização;
- A interface gráfica demonstrou ser um apoio essencial para afinação em tempo real;
- O código e dados obtidos foram integrados no repositório GitHub, garantindo transparência e partilha do conhecimento;
- Vídeos e capturas de ecrã documentaram o desempenho do sistema e serão utilizados na apresentação final.

Com os testes realizados, conclui-se que o **OpenBalance** cumpre com sucesso os objetivos definidos: demonstrar o funcionamento de um sistema de controlo em tempo real com feedback visual, utilizando uma abordagem acessível, modular e educativa.

7. Conclusão e Perspetivas Futuras

O desenvolvimento do projeto **OpenBalance** permitiu integrar, de forma prática e aplicada, os conhecimentos adquiridos ao longo da unidade curricular de **Laboratório de Aplicações em Robótica e Aprendizagem (PRIA)**. Através da construção de uma plataforma física capaz de equilibrar uma bola com base em visão computacional e controlo PID, foi possível explorar conceitos fundamentais de automação, eletrónica, programação e inteligência artificial aplicada.

A abordagem adotada foi orientada para a **simplicidade, modularidade e reprodutibilidade**, utilizando componentes acessíveis e software open-source. A integração de uma **interface gráfica interativa** permitiu não só afinar os parâmetros de controlo, mas também tornar o sistema mais pedagógico e visualmente compreensível. A utilização de **Python com OpenCV e CustomTkinter**, aliada à **comunicação serial com o Arduino**, mostrou ser uma solução eficaz e flexível para sistemas robóticos educativos.

Do ponto de vista pedagógico, o projeto destacou-se como uma ferramenta de elevada aplicabilidade no ensino de controlo automático, visão computacional e robótica, podendo ser facilmente replicado em contexto escolar ou universitário. A capacidade de visualizar em tempo real a resposta de um sistema PID perante estímulos externos reforça a ligação entre teoria e prática, promovendo a aprendizagem ativa.

Dito pelo ChatGPT:

Aqui está a **Secção 7 — Conclusão e Perspetivas Futuras**, com foco na reflexão crítica, contributos pedagógicos e caminhos de evolução do projeto **OpenBalance**. Está estruturada para fechar o relatório de forma forte e articulada com os objetivos da pós-graduação PRIA.

---

7. Conclusão e Perspetivas Futuras

O desenvolvimento do projeto **OpenBalance** permitiu integrar, de forma prática e aplicada, os conhecimentos adquiridos ao longo da unidade curricular de **Laboratório de Aplicações em Robótica e Aprendizagem (PRIA)**. Através da

construção de uma plataforma física capaz de equilibrar uma bola com base em visão computacional e controlo PID, foi possível explorar conceitos fundamentais de automação, eletrónica, programação e inteligência artificial aplicada.

A abordagem adotada foi orientada para a **simplicidade, modularidade e reprodutibilidade**, utilizando componentes acessíveis e software open-source. A integração de uma **interface gráfica interativa** permitiu não só afinar os parâmetros de controlo, mas também tornar o sistema mais pedagógico e visualmente compreensível. A utilização de **Python com OpenCV e CustomTkinter**, aliada à **comunicação serial com o Arduino**, mostrou ser uma solução eficaz e flexível para sistemas robóticos educativos.

Do ponto de vista pedagógico, o projeto destacou-se como uma ferramenta de elevada aplicabilidade no ensino de controlo automático, visão computacional e robótica, podendo ser facilmente replicado em contexto escolar ou universitário. A capacidade de visualizar em tempo real a resposta de um sistema PID perante estímulos externos reforça a ligação entre teoria e prática, promovendo a aprendizagem ativa.

Perspetivas Futuras

Tendo por base a experiência adquirida, identificam-se várias possibilidades de evolução do projeto:

- **Expansão para uma plataforma com mais graus de liberdade**, como uma estrutura tipo **Stewart** (6 DOF), permitindo simulações complexas em 3D;
- **Adição de braços articulados com capacidade de interação ativa com a bola**, incluindo movimentos como empurrar ou redirecionar a trajetória — abrindo portas a sistemas com objetivos dinâmicos, como "jogar" com a bola;
- **Integração de algoritmos de inteligência artificial**, como **aprendizagem por reforço**, permitindo que o sistema aprenda estratégias de controlo através de tentativa e erro, otimizando automaticamente os ganhos PID ou desenvolvendo comportamentos mais sofisticados;
- **Uso de câmaras de maior resolução** e **computação embarcada com Raspberry Pi**, permitindo processamento autónomo e maior portabilidade;

- **Aplicação em contextos competitivos ou demonstrativos**, como feiras de ciência ou o **AzoresBot**, onde o sistema pode ser apresentado de forma interativa ao público.

Em suma, o **OpenBalance** posiciona-se como um projeto com **elevado valor educativo**, passível de expansão em múltiplas direções — desde aplicações pedagógicas básicas até desafios de investigação avançada. O seu caráter open-source e a documentação completa disponibilizada no GitHub garantem a continuidade, partilha e evolução por parte de outros alunos, docentes ou entusiastas da robótica.

8. Bibliografia e Recursos

8.1 Bibliografia

Controlo e Automação

- Ogata, K. (2010). *Modern Control Engineering* (5th ed.). Prentice Hall.
- Dorf, R. C., & Bishop, R. H. (2017). *Modern Control Systems* (13th ed.). Pearson.
- Nise, N. S. (2020). *Control Systems Engineering* (8th ed.). Wiley.

Robótica Educacional e Arduino

- Blum, J. (2013). *Exploring Arduino: Tools and Techniques for Engineering Wizardry*. Wiley.
- Monk, S. (2016). *Programming Arduino: Getting Started with Sketches* (2nd ed.). McGraw-Hill Education.

Python, OpenCV e Interfaces Gráficas

- Beyeler, M. (2017). *Machine Learning for OpenCV*. Packt Publishing.
- Laganière, R. (2014). *OpenCV 3 Computer Vision Application Programming Cookbook*. Packt Publishing.
- Chen, J. (2023). *Learn OpenCV with Python by Examples*.
- Harrison, M. (2017). *Tkinter GUI Programming by Example*. Packt Publishing.

8.2 Webgrafia

GitHub — JohanLink:

https://github.com/JohanLink/Ball-Balancing-PID-System

GitHub — giusenso:

https://github.com/giusenso/Ball-Balancing-PID-System

Thingiverse – Ball Balancing PID Robot:

https://www.thingiverse.com/thing:4457405

Documentação OpenCV:

https://docs.opencv.org

Biblioteca Adafruit PWM Servo Driver (PCA9685):

https://learn.adafruit.com/16-channel-pwm-servo-driver

CustomTkinter – GUI Moderna para Python:

https://github.com/TomSchimansky/CustomTkinter

Introdução ao Controlo PID (PID Explained):

https://www.controleng.com/articles/pid-control-explained

Curso de Controlo PID com Arduino (Maker):

https://www.makerguides.com/arduino-pid-controller

8.3 Prompts utilizados (ChatGPT/OpenAI)

Durante o processo de desenvolvimento e documentação, foram utilizados vários prompts para obter apoio técnico, estrutural e criativo, incluindo:

- "Cria uma estrutura de relatório técnico para um projeto de controlo PID com visão computacional."
- "Explica como calibrar servos com Arduino usando PCA9685."
- "Como implementar controlo PID em Python com comunicação serial para Arduino?"
- "Sugere uma interface gráfica com CustomTkinter para controlo PID e HSV em tempo real."
- "Descreve os passos de testes e validação de um sistema que equilibra uma bola com feedback visual."
- "Adiciona perspetivas futuras para uma plataforma de controlo com múltiplos braços e IA."
- "Faz uma reflexão final para relatório técnico com foco pedagógico e open-source."