INTERFACE GRÁFICA DO TOMÓGRAFO DE IMPEDÂNCIA ELÉTRICA DA UDESC COM PYTHON E STM32CUBEIDE

Jerod William da Silva Monteiro, Pedro Bertemes Filho

INTRODUÇÃO

A tomografia por impedância elétrica (TIE) é uma técnica de imageamento que reconstrói a condutividade de objetos a partir de medições elétricas na superfície, oferecendo vantagens como ausência de radiação ionizante e monitoramento em tempo real [1]. O sistema de TIE em desenvolvimento integra hardware dedicado com STM32 para aquisição de dados e um software para processamento e reconstrução de imagens. A solução anterior apresentava limitações quanto à flexibilidade operacional, impossibilitando reconfiguração dinâmica do sistema sem interrupção, além de indisponibilidade durante a geração de imagens [2]. Este trabalho apresenta uma interface gráfica desenvolvida em Python que permite reconfiguração dinâmica em tempo de execução, visualização interativa de imagens tomográficas e controle intuitivo dos parâmetros operacionais. A aplicação também incorpora funcionalidade de playback para análise de dados previamente adquiridos, aumentando a usabilidade, acessibilidade e flexibilidade da plataforma.

DESENVOLVIMENTO

A leitura das tensões TIE utiliza *skip pattern*, excluindo eletrodos adjacentes ao par de injeção pois a impedância de contato entre eletrodo e pele gera ruídos e quedas de sinal nesses pontos, comprometendo a qualidade das medições [3]. A reconstrução de imagens é realizada através do software EIDORS [4], que estima a distribuição espacial de condutividade. O sistema é composto por hardware dedicado com microcontrolador STM32, fonte Howland para injeção estável de corrente e multiplexadores para chaveamento de eletrodos. O firmware em C controla a sequência de aquisição, realiza pré-processamento dos sinais com algoritmo DIBS e estabelece comunicação serial com o computador. A solução anterior exigia reinicialização do código para qualquer alteração operacional e apresentava travamentos durante o processamento de imagens [2]. Este trabalho desenvolveu uma interface gráfica em Python utilizando PyQt5 [5], *Matplotlib* para visualização de imagens, *NumPy* para processamento numérico e *PySerial* para comunicação com o STM32. A arquitetura foi estruturada em camadas separando interface, lógica e comunicação, com operação em threads separadas para processamento, comunicação e restante do software. A interface oferece painéis flutuantes redimensionáveis e *workspace* configurável para gerenciar múltiplos experimentos. As configurações do usuário são salvas em um arquivo JSON, enquanto a arquitetura *multithread* permite reconfiguração dinâmica em tempo de execução, visualização interativa contínua, playback de experimentos anteriores e operação responsiva sem travamentos.

RESULTADOS

A interface gráfica desenvolvida foi avaliada quanto à integração com o hardware, ao desempenho das funcionalidades implementadas e à experiência do usuário, com base em análise do código, testes de operação contínua e comparação com a solução anterior. Os testes foram realizados em computador x64 bits, processador Intel Core i5-10505 com 6 núcleos, 12 threads e 3,2 GHz, além de 8 GB de RAM, sob Windows 11 Enterprise.

A interface operou de forma estável em conjunto com o hardware existente, estabelecendo comunicação contínua e confiável. O painel de configuração opera em tempo real e transmite qualquer alteração de parâmetros imediatamente ao STM32, sem necessidade de interromper ou

reiniciar o software. Durante os testes, a interface manteve-se estável por períodos prolongados, sem quedas ou perda de dados.

O tempo total de um ciclo completo de aquisição, para 32 eletrodos e padrão adjacente, foi de 19 segundos. A comunicação serial manteve-se estável, sem perda de pacotes. A reconstrução das imagens, realizada pelo EIDORS, é atualizada a cada ciclo, mantendo fluidez mesmo em aquisições longas, com uso médio de CPU entre 5% e 10% e cerca de 220 MB de RAM. Aliada à técnica de *blitting* na renderização, essa arquitetura garante responsividade mesmo durante operações contínuas.

O módulo de playback mostrou-se eficiente para revisão retrospectiva de experimentos. Arquivos com até 60 quadros foram reproduzidos em aproximadamente 15 segundos, com controles de pausa, avanço frame a frame, retrocesso e barra de progresso, facilitando a identificação de eventos de interesse.

O salvamento automático dos dados brutos em arquivos .txt, com nomeação por *timestamp*, preserva todas as informações para reconstrução posterior e facilita a reprodutibilidade. Frente à versão anterior, os ganhos observados incluem maior responsividade durante operações demoradas, eliminando os travamentos anteriores, além de funcionalidades inéditas como o sistema de logs para visibilidade contínua do status de execução e o *workspace* configurável, que juntos ampliam as capacidades operacionais e a intuitividade do sistema.

CONSIDERAÇÕES FINAIS

Este trabalho desenvolveu uma interface gráfica para um tomógrafo de impedância elétrica utilizando *PyQt6*, transformando um script *Python* original sem interface em uma aplicação interativa que amplia a flexibilidade e a usabilidade do sistema. A reconfiguração de parâmetros sem interrupção do software e de forma simplificada ao usuário contribuiu diretamente para essa melhoria de praticidade operacional. O módulo *playback* representa um diferencial importante da solução. Por processar os dados exclusivamente no computador, ele alcançou velocidades superiores às da captura original, possibilitando revisão de experimentos. As funcionalidades implementadas atendem aos requisitos propostos no início da iniciação científica. Os testes realizados até o momento, no entanto, foram conduzidos apenas pelo próprio autor, o que limita a identificação de falhas ao que foi observado ao longo do desenvolvimento. Testes com diferentes usuários seriam importantes para revelar problemas não percebidos nesse processo e para orientar melhorias futuras de usabilidade e experiência geral da aplicação.

Palavras-chave: interface gráfica; Python; tomógrafo de impedância elétrica; EIDORS.

ANEXOS

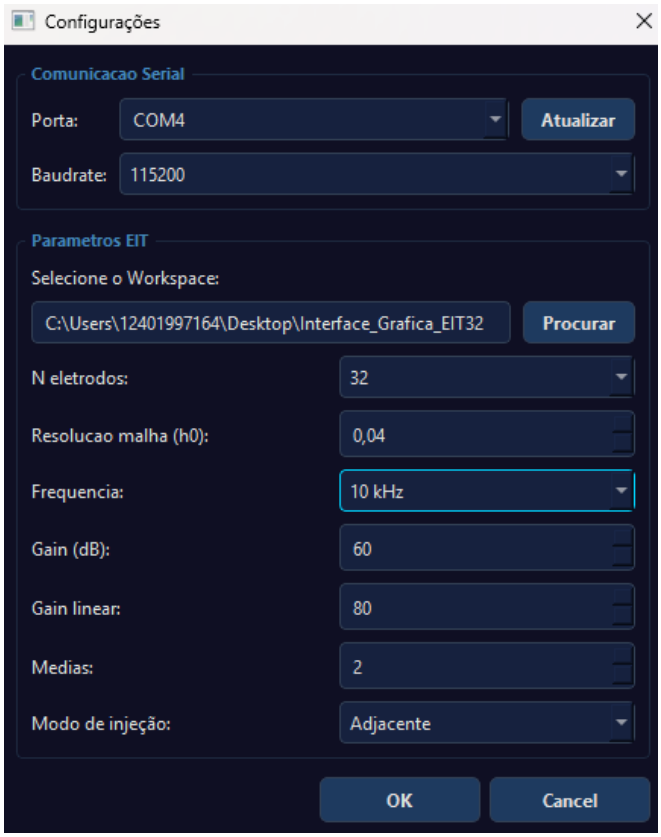


Figura 1. *Janela de configuração com parâmetros de hardware e aquisição. Fonte: Captura de tela do aplicativo desenvolvido pelo autor.*

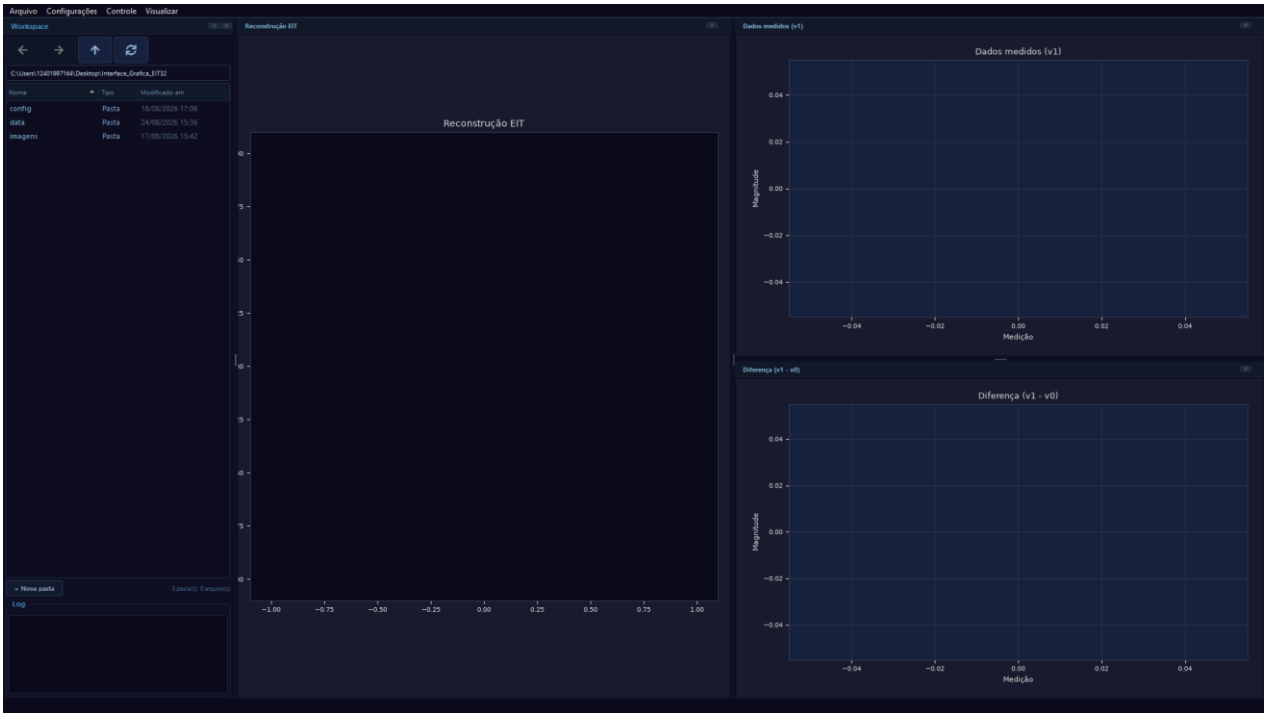


Figura 2. *Tela principal da interface gráfica, com visualização da imagem EIT e gráficos auxiliares. Fonte: Captura de tela do aplicativo desenvolvido pelo autor.*

REFERÊNCIAS BIBLIOGRÁFICAS

[1] G.J. Saulnier et al. “Electrical impedance tomography”. Em: IEEE Signal Processing Magazine 18.6 (2001), pp. 31–43. doi: 10.1109/79.962276.

[2] Kaue Felipe Morcelles. “Real-time monitoring device for 4d bioprinting based on electrical impedance tomography”. Diss. de mestr. Universidade do Estado de Santa Catarina (UDESC), 2025. url: https://repositorio.udesc.br/handle/UDESC/16034 (acesso em 06/09/2026).

[3] SHARMA, Vaishali; GOSWAMI, Mayank. Characterization of Electrical Impedance Tomography System.14 abr. 2023. Disponível em: <https://arxiv.org/abs/2304.07119v1>. (acesso em 06/09/2026).

[4] EIDORS: Electrical Impedance and Diffuse Optical Reconstruction Software. url: https://eidors3d.sourceforge.net/ (acesso em 06/09/2026).

[5] Martin Lewin. PyQt5 Tutorial. url: https://www.pythonguis.com/pyqt5-tutorial/ (acesso em 06/09/2026).

[6] Lin Yang et al. “Evaluation of adjacent and opposite current injection patterns for a wearable chest electrical impedance tomography system”. Em: Physiological Measurement 45.2 (fev. de 2024), p. 025004. doi: 10.1088/1361-657.