

Quando o “Hello World” não faz sentido: uma proposta de tutor conversacional para aprender programação

Victor Braga¹, Jeniffer Macena¹, David Fernandes², João Bernardo¹,
Marcela Pessoa¹, Fernanda Pires¹

¹ Escola Superior de Tecnologia – Universidade do Estado do Amazonas (EST-UEA)
ThinkTED Lab – Pesquisa, Desenvolvimento e Inovação em Tecnologias Emergentes

²Programa de Pós-Graduação em Informática (PPGI)
Instituto de Computação – Universidade Federal do Amazonas (IComp – UFAM)

{vab.snf21, mspessoa, fpires}@uea.edu.br

{jeniffer.souza, david, jrbs}@icomp.ufam.edu.br

Abstract. *The abstract nature of computing concepts can make learning more difficult. This study presents TutorIA, a virtual environment featuring a conversational agent based on a Large Language Model (LLM) that integrates text, voice, and image generation to support the learning of programming. The system was guided by a computing reference curriculum and developed iteratively, with prompt engineering guidelines aimed at reducing out-of-domain responses. Five experts evaluated the robustness of the responses, the quality of the illustrations, and aspects of the interface. The results indicated positive perceptions and the tool’s potential to support the study of logic and algorithms, without measuring learning gains.*

Resumo. *A natureza abstrata dos conteúdos de computação pode dificultar a aprendizagem. Este estudo apresenta o TutorIA, um ambiente virtual com agente conversacional baseado em Large Language Model (LLM), que integra texto, voz e geração de imagens para apoiar a aprendizagem de programação. O sistema foi orientado por currículo de referência da computação e desenvolvido de forma iterativa, com diretrizes de engenharia de prompt para reduzir respostas fora do domínio. Cinco especialistas avaliaram a robustez das respostas, a qualidade das ilustrações e aspectos da interface. Os resultados indicaram percepções positivas e potencial da ferramenta para apoiar o estudo de lógica e algoritmos, sem mensurar ganhos de aprendizagem.*

1. Introdução

Disciplinas introdutórias de programação apresentam elevadas taxas de reprovação e evasão [Silva et al. 2023], atingindo até 40% na modalidade presencial [França 2024]. A natureza abstrata das linguagens exige que os estudantes interpretem problemas, estructurem soluções lógicas e compreendam processos sem representação concreta imediata [Lahtinen et al. 2005, Robins 2019]. Além disso, fatores como a sobrecarga cognitiva [Paas et al. 2010] podem tornar a aprendizagem frustrante para iniciantes, gerando dificuldades persistentes ao longo da formação [Morais et al. 2020,

Bennedsen and Caspersen 2007]. Tal cenário demanda ferramentas que auxiliem na visualização de conceitos complexos por meio de variadas formas de interação e representação.

Como alternativa, ambientes virtuais 3D e agentes conversacionais destacam-se como recursos promissores [Chiu et al. 2023, Stracke et al. 2025]. Os agentes conversacionais podem proporcionar experiências interativas, permitindo representar conceitos abstratos de forma mais concreta, estimulando a experimentação e o engajamento do estudante [da Silva et al. 2025]. Modelos baseados em *Large Language Models* (LLMs) ampliam esse potencial ao gerarem respostas personalizadas em linguagem natural, tornando a comunicação flexível e dinâmica [Dam et al. 2024]. Apesar disso, ainda há desafios técnicos para integrar de forma coesa comunicação textual, captação de voz e representações visuais voltadas ao ensino de computação [Kiran et al. 2025].

Diante desse cenário, este trabalho busca responder à seguinte questão de pesquisa: como a combinação de recursos textuais, visuais e sonoros em um tutor conversacional baseado em IA pode apoiar o processo de aprendizagem de lógica e algoritmos, e qual o potencial de aplicação dessa ferramenta segundo uma avaliação técnica preliminar?

Considerando a complexidade desses conteúdos e o potencial das LLMs, este trabalho apresenta o TutorIA, um ambiente educacional 3D composto por um tutor conversacional inteligente. Projetada para auxiliar na compreensão de conceitos abstratos, a ferramenta articula três recursos multimodais: (i) um avatar 3D dinâmico que combina respostas em texto, síntese de voz, expressões faciais e gestuais; (ii) entrada por voz convertida em *prompt*, como alternativa à digitação; e (iii) geração de representações visuais para ilustrar fluxos de execução, favorecendo a construção de modelos mentais concretos. Por fim, este estudo avalia a percepção de cinco especialistas em computação quanto à robustez técnica e ao potencial pedagógico do TutorIA no apoio ao ensino de algoritmos, sem, contudo, mensurar diretamente o impacto da ferramenta na aprendizagem dos estudantes.

2. Fundamentação Teórica e Trabalhos Relacionados

A aprendizagem de computação exige habilidades como pensamento abstrato, raciocínio lógico e decomposição de problemas para traduzir processos mentais em algoritmos [Robins 2019].

A Teoria da Carga Cognitiva de Sweller [Sweller 1988] sustenta que a aprendizagem é prejudicada quando a capacidade limitada da memória de trabalho é sobrecarregada, situação frequente na programação introdutória. Complementarmente, a Teoria da Aprendizagem Multimídia de Mayer [Mayer 2001] e a Teoria da Codificação Dual de Paivio [Clark and Paivio 1991] indicam que a integração entre texto, imagem e áudio favorece a construção de modelos mentais mais robustos. Essas perspectivas fundamentam o uso de recursos multimodais no TutorIA para reduzir a carga cognitiva extrínseca.

Ambientes virtuais 3D e agentes conversacionais podem tornar conceitos abstratos em representações mais concretas [Campos et al. 2022], oferecendo mediação pedagógica e *feedback* imediato [Paiva and Tori 2017, Baylor and Kim 2005]. Sob a perspectiva da Teoria da Presença Social, a incorporação de avatares favorece a percepção do agente como uma entidade social, podendo ampliar o engajamento, a motivação e a

eficácia da aprendizagem [Kreijns et al. 2021]. Em conjunto, essas evidências sugerem que a combinação entre presença social e recursos multimodais constitui uma abordagem promissora para apoiar a aprendizagem em contextos de elevada carga cognitiva.

A integração de LLMs pode potencializar esses agentes ao permitir respostas contextualizadas e adaptadas ao ritmo do aprendiz [Dam et al. 2024, Kamalov et al. 2026]. Para que essas respostas sejam pedagogicamente adequadas, diferentes estratégias de Engenharia de *Prompt* (*Prompt Engineering*) podem ser utilizadas para definir restrições de resposta, papéis instrucionais e progressão do conteúdo apresentado ao estudante [Chen et al. 2023, Sahoo et al. 2024], garantindo que agentes conversacionais mantenham interações mais coerentes com os objetivos educacionais e com o nível de conhecimento do usuário [Olea et al. 2024, Liu et al. 2024, Wu et al. 2022].

O CosmoBot [Estrela et al. 2024] oferece suporte conversacional baseado em questões e dicas, enquanto o CoderBot 2.0 [Mendes et al. 2025] utiliza LLMs e engenharia de *prompt* para gerar explicações, *feedbacks* e exemplos sob a perspectiva da Teoria da Carga Cognitiva. Ambas as soluções, contudo, utilizam interfaces predominantemente textuais, sem integrar voz, avatar e geração de imagens. O RoboEdu [Fernando 2025], por sua vez, combina LLMs e recomendação de mídias externas para apoiar o ensino de Estruturas de Dados, mas não dispõe de tutor virtual personificado nem produz ilustrações personalizadas durante o diálogo. Assim, o principal diferencial do TutorIA está na integração entre LLM, avatar 3D, voz e geração contextual de imagens para apoiar a aprendizagem de programação introdutória.

3. Abordagem e Passos Metodológicos

O TutorIA é um agente conversacional integrado a um ambiente virtual 3D para apoiar a compreensão de conceitos de programação. Seu desenvolvimento seguiu ciclos de melhoria contínua [Pires 2021], organizados nas etapas descritas a seguir.

3.1. Identificar um Problema

O problema foi identificado na literatura de Educação em Computação: a abstração dos processos lógicos é barreira recorrente na aprendizagem de programação [Robins 2019, Lahtinen et al. 2005, Bennedsen and Caspersen 2007]. Para mitigar esse desafio, os referenciais formativos da SBC [Sociedade Brasileira de Computação 2005] e do MEC [BRASIL. 2016] orientaram o projeto do TutorIA em duas frentes: (i) na definição dos conteúdos, estruturando a progressão programática em módulos sobre tópicos essenciais de algoritmos; e (ii) nas funcionalidades, utilizando a alternância dinâmica de diretrizes no *system prompt* associada ao suporte multimodal (voz e imagens). Essa estratégia viabiliza uma evolução pedagógica por níveis, favorecendo o desenvolvimento gradual do raciocínio lógico em vez de fornecer correções automáticas prontas.

3.2. Pesquisar

A investigação abrangeu: (i) barreiras cognitivas na aprendizagem de programação [Robins 2019]; (ii) ferramentas correlatas [Estrela et al. 2024, Mendes et al. 2025, Fernando 2025]; e (iii) ementas de referência [USP 2024, IC UNICAMP 2025, UFRGS 2021]. A partir disso, definiu-se uma trilha com três níveis: iniciante (variáveis, operadores e estruturas de controle), intermediário (funções, vetores e recursão) e avançado (estruturas de dados e algoritmos de ordenação).

3.3. Imaginar e Refletir

Nesta fase, idealizou-se a criação de um ambiente virtual com tutor inteligente baseado em LLM para apoio à aprendizagem de programação. Além disso, foram realizados o levantamento de requisitos, o planejamento da arquitetura de integração entre Unity e as APIs de IA e a modelagem inicial do sistema. Os aspectos resultantes desse processo são detalhados a seguir.

Levantamento de requisitos: com base na literatura sobre carga cognitiva [Bennedsen and Caspersen 2007], ambientes tridimensionais [Paiva and Tori 2017], multimodalidade [Stevenson et al. 2017] e tutoria por LLMs [Dam et al. 2024], analisaram-se desafios amplamente relatados na aprendizagem inicial de programação, tais como a elevada abstração conceitual, a sobrecarga cognitiva e as limitações de interfaces estritamente textuais. A partir dessa análise, definiram-se os requisitos funcionais do ambiente, priorizando mecanismos de suporte visual, auditivo e interativo à aprendizagem, descritos na documentação complementar.¹

Arquitetura da aplicação: a arquitetura foi documentada com o C4 Model [Brown 2013]. A Figura 1 apresenta o diagrama de contêineres, destacando a interação entre o estudante, a aplicação Unity 3D e os serviços externos: modelo de linguagem hospedado na Groq e serviços de voz e geração multimodal da Google Cloud.

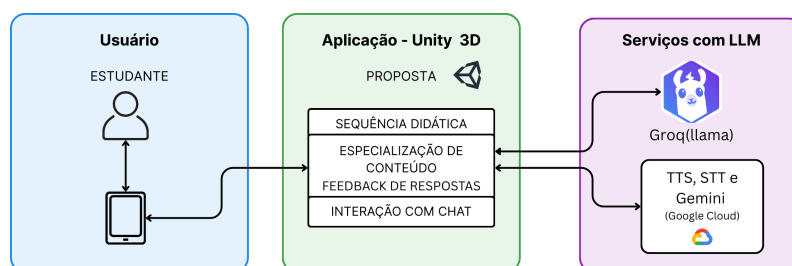


Figura 1. Arquitetura geral da aplicação.

Modelagem e tecnologias: os diagramas de modelagem (caso de uso e sequência) estão disponíveis como material suplementar². O TutorIA foi desenvolvido em Unity, utilizando o modelo Llama 3.1 (Groq) para geração de respostas, Gemini 2.0 Flash para geração de imagens e os serviços Speech-to-Text (STT) e Text-to-Speech (TTS) da Google Cloud para interação por voz.

3.4. Criação

O desenvolvimento iniciou-se na Unity, onde foi implementado um *chatbox* para validar a comunicação com a LLM. O avatar foi refinado iterativamente com base em princípios de design de interação [Preece et al. 2002] até atingir compatibilidade com as animações e uma aparência adequada ao contexto educacional, sendo adotado um robô estilizado da Unity Asset Store.

A arquitetura final do sistema articula um *script* principal na Unity como gerenciador central do fluxo de dados e do *system prompt* pedagógico. A comunicação ocorre

¹Requisitos: <https://bit.ly/4nZoEAE>

²Modelagem: <https://bit.ly/4nZoEAE>

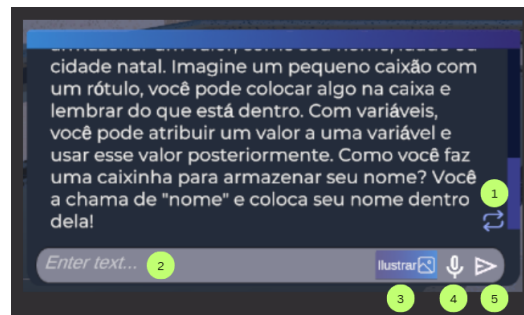
sequencialmente: (i) a entrada do estudante (texto ou voz) é processada, utilizando-se o serviço de *Speech-to-Text* (STT) da Google Cloud para conversão do áudio; (ii) o texto é integrado ao contexto instrucional e enviado via API à LLM; (iii) a LLM avalia a complexidade do tema e insere tags estruturadas ocultas ao final da resposta (indicando o conceito e a necessidade de suporte visual). O gerenciador Unity filtra e remove essas tags antes de exibir o texto ao usuário, disparando simultaneamente a síntese de áudio via *Text-to-Speech* (TTS). Caso a tag de ilustração seja positiva, o sistema ativa um botão na interface e orienta o tutor a oferecer o recurso. Se o usuário acioná-lo, o gerenciador recupera a tag oculta e estrutura um *prompt* secundário para a API do Gemini Flash gerar a imagem, exibida no projetor virtual do cenário (Figura 2a).

Diretrizes pedagógicas e engenharia de *prompt*: para reduzir respostas fora do domínio de programação, o tutor emprega regras de engenharia de *prompt* que restringem o escopo temático, adaptam a linguagem ao nível do estudante e adotam estratégias de *scaffolding*. Apesar disso, como discutido na Seção 4, essas medidas mitigam, mas não eliminam completamente respostas inadequadas.

Personalização e configurações: ao ingressar no ambiente, o usuário define, no menu de configurações,³ o nível de conhecimento (iniciante, intermediário ou avançado) e o perfil de voz do tutor (Robô, Masculino ou Feminino). O *chatbox* (Figura 2b) reúne cinco funções: (1) repetição da última resposta, para revisão; (2) entrada de texto; (3) geração manual de ilustrações; (4) ativação do STT para entrada por voz; e (5) envio da mensagem.



(a) Geração de imagem.



(b) Detalhamento da chatbox.

Figura 2. Recursos visuais e componentes de interação do TutorIA.

3.5. Brincar e Testar

A avaliação adotou uma abordagem mista por meio de uma análise técnica por especialistas (*expert evaluation*) [Klas 2012]. A escolha por especialistas, em vez de alunos iniciantes, justifica-se porque eles têm maior facilidade para identificar erros da LLM e avaliar se o nível de dificuldade está adequado à aprendizagem introdutória. A investigação focou em dois objetivos: (i) robustez da LLM, analisando a correção técnica do código e a linguagem do agente; e (ii) apoio visual, medindo a clareza e a utilidade das imagens para explicar os conceitos abstratos. Cabe destacar que essa etapa avaliativa mensura a percepção especializada sobre a robustez técnica e o potencial pedagógico do sistema,

³Interfaces do sistema: <https://bit.ly/4nZoEAE>

não constituindo uma medida direta de ganho de aprendizagem, investigação reservada a estudos futuros com estudantes.

Participantes: cinco especialistas da área de Computação, com formação em Licenciatura em Computação (N=3), Sistemas de Informação (N=1) e Engenharia da Computação (N=1). Todos declararam uso frequente de LLMs para estudo de programação, geração e revisão de código, e escrita acadêmica. **Instrumento:** protocolo estruturado com roteiro de interação e formulário no Google Forms, organizado em dois eixos: robustez das respostas e qualidade das ilustrações, avaliados em escala Likert de cinco pontos e complementados por questões abertas. O roteiro orientou os participantes a formular perguntas sobre programação, incluindo reformulações para verificar a consistência das respostas e explorar os recursos multimodais do sistema. **Procedimento:** sessões presenciais e remotas (30–60 min), com apresentação do sistema, interação livre com o TutorIA, preenchimento do formulário e entrevista semiestruturada. **Análise:** os dados quantitativos foram tratados por estatística descritiva (média e desvio padrão), enquanto comentários e entrevistas foram analisados por Análise de Conteúdo [Bardin 2016], utilizando categorias definidas a partir do roteiro de entrevista.

3.6. Aspectos Éticos

O estudo seguiu os princípios éticos de pesquisa com seres humanos. Todos os participantes assinaram o Termo de Consentimento Livre e Esclarecido (TCLE), autorizando que a entrevista fosse gravada e que as informações fossem registradas em documento, sendo anonimizadas e utilizadas exclusivamente para fins acadêmicos.

4. Resultados e Discussões

Os resultados são apresentados em dois momentos complementares: formulário (dados quantitativos e qualitativos) e entrevista (aprofundamento das percepções).

4.1. Avaliação das Percepções dos Especialistas Registradas no Formulário

O instrumento de avaliação contemplou questões fechadas e abertas para analisar a percepção dos especialistas quanto à robustez das respostas e à qualidade das imagens geradas. Os dados estão disponíveis para consulta⁴.

Robustez das respostas: os especialistas atribuíram avaliações elevadas às métricas de consistência, correção técnica e adequação da linguagem (média = 4,6; DP = 0,55) (Tabela 1). A restrição ao domínio apresentou menor média (4,0) e maior variabilidade (DP = 1,41), refletindo respostas que extrapolaram o contexto da programação. O Participante 5 relatou que o conceito de "vetor" foi explicado sob uma perspectiva física e matemática, em vez de estrutura de dados, enquanto o Participante 2 observou que a explicação sobre a sequência de Fibonacci enfatizou aspectos matemáticos sem relacioná-los à sua aplicação algorítmica. Esses resultados indicam que as regras de engenharia de *prompt* (Seção 3) reduzem, mas não eliminam, interpretações fora do domínio, reforçando a necessidade de mecanismos adicionais de controle contextual, como bases de conhecimento especializadas.

Qualidade das ilustrações: as imagens geradas receberam avaliações elevadas nos critérios de correspondência ao conceito, clareza visual e coerência entre texto e

⁴Dados de avaliação: <https://bit.ly/4nZoEAE>

Tabela 1. Resultados da avaliação da robustez das respostas.

Métrica	Média	Desvio Padrão
Consistência das respostas	4.6	0.55
Correção técnica	4.6	0.55
Adequação da linguagem	4.6	0.55
Restrição ao domínio	4.0	1.41

imagem, todos com média de 4,2 e desvio padrão de 0,84 (Tabela 2). Os resultados indicam que as ilustrações complementaram adequadamente as explicações do tutor e contribuíram para a representação visual dos conceitos. As diferenças de percepção entre os especialistas estiveram associadas principalmente à legibilidade de elementos textuais em algumas imagens, e não à pertinência ou à coerência das ilustrações produzidas.

Tabela 2. Resultados da avaliação da qualidade das ilustrações.

Métrica	Média	Desvio Padrão
Correspondência da imagem	4.2	0.84
Clareza da ilustração	4.2	0.84
Coerência texto–imagem	4.2	0.84

Nas questões abertas, os especialistas destacaram a interatividade, as diferentes formas de entrada e o uso de exemplos contextualizados. As principais sugestões envolveram maior controle sobre o detalhamento das respostas, delimitação mais rigorosa do domínio de programação e melhorias na legibilidade dos textos presentes nas imagens.

4.2. Análise das Entrevistas com Participantes

Com o objetivo de aprofundar os resultados obtidos nos questionários, realizou-se uma análise das entrevistas conduzidas com os participantes. Os relatos na íntegra e a síntese das transcrições encontram-se disponíveis na documentação complementar do projeto⁵. Para fins de análise, essas percepções qualitativas foram organizadas nos quatro eixos definidos no protocolo de avaliação: impressão geral, interface e avatar, comunicação e conteúdo, e utilidade e adaptação. A análise foi conduzida por meio da técnica de Análise de Conteúdo [Bardin 2016], utilizando categorias definidas a priori a partir desses eixos e buscando identificar percepções recorrentes, divergências e sugestões de melhoria.

No eixo “**impressão geral**”, os participantes relataram uma percepção positiva da proposta, destacando o potencial do ambiente para apoiar a aprendizagem de programação por meio da integração entre tutor conversacional, avatar 3D e recursos visuais. A possibilidade de receber explicações acompanhadas de ilustrações foi frequentemente mencionada como um diferencial para reduzir a abstração dos conceitos computacionais. Como exemplos dessa percepção, o Participante 2 considerou a proposta “relevante e pertinente”, enquanto o Participante 3 a descreveu como uma forma divertida de aprender programação. O Participante 1 também destacou a possibilidade de formular perguntas e ouvir as explicações por voz como alternativa à digitação.

No eixo “**interface e avatar**”, os entrevistados sugeriram melhorias relacionadas à organização visual, incluindo a ampliação da área de chat, melhor posicionamento das

⁵Transcrição das entrevistas: <https://bit.ly/4nZoEAE>

imagens e aprimoramentos na navegação. Também foram apontadas oportunidades de evolução para o avatar, especialmente quanto à sincronização do áudio, velocidade da fala e maior dinamismo das animações. A ampliação da área do chat foi uma sugestão recorrente, mencionada pelos Participantes 1, 2, 3, 4 e 5. Também foram sugeridos botões mais intuitivos, uma opção clara para sair do ambiente, ajustes na paleta de cores e a possibilidade de interromper a reprodução do áudio.

No eixo **“comunicação e conteúdo”**, os participantes avaliaram positivamente a clareza das explicações. Entretanto, alguns relataram situações em que o sistema produziu respostas excessivamente longas ou apresentou conceitos fora do contexto da programação, indicando a necessidade de mecanismos para controle do nível de detalhamento e maior restrição ao domínio. Os Participantes 1, 2 e 4 observaram que algumas explicações eram mais extensas do que o necessário. Em relação à restrição ao domínio, o Participante 5 relatou que o termo “vetor” foi interpretado fora do contexto da programação, enquanto o Participante 2 identificou inconsistências nas respostas sobre caminho mínimo e sequência de Fibonacci. Esses relatos corroboram os resultados quantitativos relativos à restrição ao domínio e indicam que as diretrizes de engenharia de *prompt* mitigam, mas não eliminam, interpretações inadequadas.

Por fim, em relação ao eixo **“utilidade e adaptação”**, os entrevistados reconheceram o potencial da ferramenta como apoio à aprendizagem inicial de programação, dada a combinação de texto, voz e imagens. Como melhorias, sugeriram a inclusão de modos de uso distintos (como tutorial e revisão), personalização do estilo de resposta, mecanismos de anotação e adaptação contínua do conteúdo ao perfil do estudante.

4.3. Limitações e Ameaças à Validade

Este estudo apresenta resultados preliminares de uma avaliação piloto com especialistas, realizada para apoiar o refinamento do TutorIA. Assim, os achados não podem ser generalizados nem utilizados para inferir impacto na aprendizagem. Além disso, as respostas geradas pela LLM estão sujeitas à variabilidade inerente aos modelos generativos, o que, como discutido na Seção 4, se manifesta sobretudo em conceitos polissêmicos ainda não totalmente restritos pelas diretrizes de *prompt* adotadas.

5. Considerações Finais

Este trabalho apresentou o TutorIA, um ambiente educacional 3D que integra um tutor conversacional baseado em LLM, avatar, voz e geração de imagens para apoiar a aprendizagem de programação introdutória. A avaliação com especialistas indicou percepções favoráveis quanto à consistência e correção técnica das respostas, adequação da linguagem e qualidade das ilustrações. Os dados qualitativos destacaram a interatividade e as representações visuais como recursos promissores para reduzir a abstração dos conteúdos, além de indicar melhorias no controle do detalhamento das respostas, na delimitação do domínio e na legibilidade dos textos nas imagens.

Como continuidade, pretende-se realizar estudos experimentais com estudantes, investigando separadamente a contribuição de avatar, voz e imagens para a redução da abstração percebida e para a aprendizagem, além de aprimorar a personalização das respostas e das representações visuais.

Declaração sobre uso de Inteligência Artificial

Declaramos o uso das ferramentas Claude (Anthropic) e Gemini (Google) como apoio na revisão textual, para apoio à correção de erros ortográficos e gramaticais ao longo do texto. Todo o conteúdo científico, incluindo a concepção da pesquisa, coleta de dados e conclusões, foi desenvolvido pelos autores. O uso das ferramentas não exime os autores da responsabilidade sobre o conteúdo do trabalho.

Referências

- Bardin, L. (2016). *Análise de Conteúdo*. Edições 70, São Paulo.
- Baylor, A. and Kim, Y. (2005). Simulating instructional roles through pedagogical agents. *International Journal of Artificial Intelligence in Education*, 15:95–115.
- Bennedsen, J. and Caspersen, M. (2007). Failure rates in introductory programming. *SIGCSE Bulletin*, 39:32–36.
- BRASIL. (2016). Resolução cne/ces nº 5, de 16 de novembro de 2016. institui as diretrizes curriculares nacionais para os cursos de graduação na área da computação. Diário Oficial da União, Brasília, 17 nov. 2016.
- Brown, S. (2013). Software architecture for developers. *Coding the Architecture*.
- Campos, E., Hidrogo, I., and Zavala, G. (2022). Impact of virtual reality use on the teaching and learning of vectors. In *Frontiers in Education*, volume 7, page 965640. Frontiers Media SA.
- Chen, B., Zhang, Z., Langrené, N., and Zhu, S. (2023). Unleashing the potential of prompt engineering for large language models. *Patterns*, 6.
- Chiu, T. K., Moorhouse, B., Chai, C., and Ismailov, M. (2023). Teacher support and student motivation to learn with artificial intelligence (ai) based chatbot. *Interactive Learning Environments*.
- Clark, J. and Paivio, A. (1991). Dual coding theory and education. *Educational Psychology Review*, 3:149–210.
- da Silva, F. X., de Menezes, C. S., and de Castro Junior, A. N. (2025). Dificuldades dos estudantes na aprendizagem de programação: Uma revisão sistemática da literatura. *Cadernos de Educação Tecnologia e Sociedade*, 18(3):994–1009.
- Dam, S. K., Hong, C. S., Qiao, Y., and Zhang, C. (2024). A complete survey on llm-based ai chatbots.
- Estrela, I., Barbosa, K., Brito, C., and Neto, C. S. (2024). Ferramenta cosmobot: Um chatbot de apoio a alunos em avaliações de algoritmos. In *Anais do XXXV Simpósio Brasileiro de Informática na Educação*, pages 380–391, Porto Alegre, RS, Brasil. SBC.
- Fernando, D. H. (2025). Integração de modelo generativo em sistema tutor inteligente para aulas de estruturas de dados.
- França, G. M. (2024). *O uso de jogos e inteligência artificial na educação superior: experiência de uso no ensino de tecnologia de informação*. PhD thesis, Universidade de São Paulo.

- IC UNICAMP (2025). Mc102 – algoritmos e programação de computadores. Acesso em: 4 jun. 2025.
- Kamalov, F., Calonge, D. S., Smail, L., Azizov, D., Thadani, D. R., Kwong, T., and Atif, A. (2026). Evolution of ai in education: Agentic workflows.
- Kiran, A. S. U., Khan, A., Khan, M. U. J. A., Ahmad, S. A. A. D., Patil, A. H., and Kore, S. (2025). Voice assisted data structure implementation and visualization. *International Journal For Multidisciplinary Research*.
- Klas, C.-P. (2012). Expert evaluation methods. *Dobrev et al*, pages 75–84.
- Kreijns, K., Xu, K. M., and Weidlich, J. (2021). Social presence: Conceptualization and measurement. *Educational Psychology Review*, 34:139 – 170.
- Lahtinen, E., Ala-Mutka, K., and Järvinen, H.-M. (2005). A study of the difficulties of novice programmers. *ACM SIGCSE Bulletin*, 37(3):14–18.
- Liu, M. X., Liu, F., Fiannaca, A. J., Koo, T., Dixon, L., Terry, M., and Cai, C. J. (2024). ”we need structured output”: Towards user-centered constraints on large language model output. In *Extended Abstracts of the CHI Conference on Human Factors in Computing Systems*, CHI EA ’24, New York, NY, USA. Association for Computing Machinery.
- Mayer, R. E. (2001). *Multimedia Learning*. Cambridge University Press, New York.
- Mendes, A., Parizotto, A., Garcia, R., Garcia, M., Guedes, G., Vilela, R., Valle, P., Balancieri, R., and Silva, W. (2025). Coderbot 2.0: Integrating llm and prompt engineering in the evolution of an educational chatbot. In *Anais do XXXVI Simpósio Brasileiro de Informática na Educação*, pages 1497–1506, Porto Alegre, RS, Brasil. SBC.
- Morais, C., MENDES NETO, F., and Osório, A. (2020). Dificuldades e desafios do processo de aprendizagem de algoritmos e programação no ensino superior: uma revisão sistemática de literatura. *Research, Society and Development*, 9.
- Olea, C., Tucker, H., Phelan, J., Pattison, C., Zhang, S., Lieb, M., Schmidt, D. C., and White, J. (2024). Evaluating persona prompting for question answering tasks. *Security, Privacy and Trust Management*.
- Paas, F., Renkl, A., and Sweller, J. (2010). Cognitive load theory and instructional design: Recent developments. *Educational Psychologist*, 38:1–4.
- Paiva, C. A. and Tori, R. (2017). Jogos digitais no ensino: processos cognitivos, benefícios e desafios. *XVI Simpósio Brasileiro de Jogos e Entretenimento Digital*, pages 1–4.
- Pires, F. G. d. S. (2021). Thinkted lab, um caso de aprendizagem criativa em computação no nível superior.
- Preece, J., Rogers, Y., and Sharp, H. (2002). *Interaction Design: Beyond Human-Computer Interaction*. John Wiley & Sons.
- Robins, A. V. (2019). 12 novice programmers and introductory programming. *The Cambridge handbook of computing education research*, page 327.

- Sahoo, P., Singh, A. K., Saha, S., Jain, V., Mondal, S., and Chadha, A. (2024). A systematic survey of prompt engineering in large language models: Techniques and applications. *ArXiv*, abs/2402.07927.
- Silva, C., Solano, J., Santos, A., and Reis, J. (2023). Previsão de reprovações em disciplinas introdutórias de programação: Um estudo em um ambiente de correção automática de códigos. In *Anais do XXXIV Simpósio Brasileiro de Informática na Educação*, pages 1524–1535, Porto Alegre, RS, Brasil. SBC.
- Sociedade Brasileira de Computação (2005). Currículo de referência da sbc para cursos de graduação em computação. Acesso em: 4 jun. 2025.
- Stevenson, R. A., Siemann, J. K., Schneider, B. C., and et al. (2017). Multisensory temporal integration in autism spectrum disorders. *Autism Research*, 10(5):777–789.
- Stracke, C., Ende, P., Adler, S., Heller, E., Deuchler, J., Pomino, J., and Wölfel, M. (2025). Immersive virtual reality in higher education: a systematic review of the scientific literature. *Virtual Reality*, 29.
- Sweller, J. (1988). Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12(2):257–285.
- UFRGS (2021). INF01126 - Estruturas de Dados. Acesso em: 6 jun. 2025.
- USP (2024). SME0230 - Introdução à Ciência da Computação. Acesso em: 6 jun. 2025.
- Wu, T., Terry, M., and Cai, C. J. (2022). Ai chains: Transparent and controllable human-ai interaction by chaining large language model prompts.