O uso das plataformas Scratch e Code.org para o desenvolvimento do Pensamento Computacional na educação básica: relato de experiência alinhado à BNCC

Jaime Antonio Daniel Filho[1], **Luís Gustavo Werle Tozevich**[1], **Guilherme Meneghetti Einloft**[1], **Diego Ribeiro Chaves**[1], **Giovani Rubert Librelotto**[2]

[1]Curso de Ciência da Computação –
Universidade Federal de Santa Maria (UFSM)

[2]Departamento de Linguagens e Sistemas de Computação (DLSC) –
Universidade Federal de Santa Maria (UFSM)

{jafilho, lgtozevich, gmeinloft, drchaves, librelotto}@inf.ufsm.br

Abstract. *This article reports on a pedagogical experience for teaching Computational Thinking (CT) to 9th-grade students, addressing the lack of structured methodologies for implementing the BNCC (Brazilian National Common Curricular Base) in public schools. The intervention, based on the 5E Instructional Model, integrated unplugged activities with the Code.org and Scratch platforms, culminating in the development of digital games. The results demonstrate high student engagement and a grasp of abstract concepts, such as the simulation of gravity, which connected programming with mathematics and physics. It is concluded that the methodology is a playful, effective, and replicable strategy for the development of CT in basic education.*

Resumo. *Este artigo relata uma experiência pedagógica para o ensino do Pensamento Computacional (PC) a alunos do 9º ano do Ensino Fundamental, abordando a carência de metodologias estruturadas para a aplicação da BNCC em escolas públicas. A intervenção, fundamentada no Modelo Instrucional 5E, articulou atividades desplugadas com as plataformas Code.org e Scratch, culminando no desenvolvimento de jogos digitais. Os resultados demonstraram o alto engajamento dos alunos e apropriação de conceitos abstratos, como a simulação de gravidade, que conectou programação à matemática e à física. Conclui-se que a metodologia é uma estratégia lúdica, eficaz e replicável para o desenvolvimento do PC na educação básica.*

1. Introdução

A recente inclusão do Pensamento Computacional (PC) na Base Nacional Comum Curricular (BNCC) estabeleceu um novo imperativo para a educação básica no Brasil, demandando que os sistemas de ensino superem o letramento digital instrumental em favor do desenvolvimento de competências para a resolução de problemas de maior complexidade [Brasil 2022]. Contudo, existe um hiato significativo entre a diretriz normativa e sua efetiva operacionalização em sala de aula. A literatura aponta que essa lacuna é exacerbada por dois fatores interdependentes: a carência de formação docente específica [Oliveira and Cambraia 2020] e, consequentemente, a ausência de modelos pedagógicos

DOI: 10.5753/wie.2025.13496

estruturados e validados. Estudos demonstram que, embora a importância do PC seja reconhecida, muitos professores da educação básica não tiveram formação sobre o tema e trabalham seus conceitos de forma não intencional [Martins et al. 2021]. Isso frequentemente resulta em práticas de ensino fragmentadas e focadas em ferramentas, em detrimento de uma base conceitual sólida [Gatti 2017, Brackmann 2017].

Neste contexto, enquanto plataformas como Scratch e Code.org oferecem recursos acessíveis, seu uso isolado não garante a aprendizagem significativa. Isso porque, as intervenções se restringem muitas vezes a oficinas pontuais, realizadas por pesquisadores externos, sem um acompanhamento e integração efetiva por parte dos professores da escola [Brezolin and Silveira 2021]. Este trabalho endereça, portanto, a carência de um modelo metodológico que articule estas ferramentas dentro de um ciclo pedagógico coeso, capaz de guiar o estudante desde a conceitualização abstrata até a criação autoral. A hipótese central é que a adaptação de um modelo instrucional consolidado, como o 5E, pode fornecer a estrutura necessária para essa integração.

Dessa forma, o objetivo deste artigo é apresentar e analisar um modelo pedagógico para o ensino do PC, cuja arquitetura articula o Modelo Instrucional 5E com uma sequência escalonada de atividades desplugadas e digitais. Através de um estudo de caso conduzido sob a abordagem da pesquisa-ação [Tripp 2005], foi demonstrado a viabilidade e a eficácia deste modelo. A contribuição do trabalho é, portanto, um modelo replicável que oferece um caminho para a implementação qualificada do PC, alinhado à BNCC. O restante do artigo está estruturado da seguinte forma: a Seção 2 apresenta o referencial teórico; a Seção 3 detalha a metodologia; a Seção 4 apresenta e discute os resultados; e a Seção 5 conclui o trabalho, sintetizando as descobertas e apontando direções futuras.

2. Revisão bibliográfica

Esta seção tem como objetivo apresentar os fundamentos teóricos que sustentam esta pesquisa, abordando os conceitos do Pensamento Computacional (PC), as estratégias pedagógicas baseadas na gamificação e o modelo instrucional 5E.

O Pensamento Computacional (PC) é considerado uma habilidade essencial para todos os indivíduos, indo além do campo da Ciência da Computação. De acordo com Wing (2006), esse conceito corresponde ao processo de pensamento necessário para formular problemas e expressar suas soluções de forma que possam ser executadas por um computador, uma pessoa ou qualquer máquina. Os principais pilares que estruturam o PC envolvem a decomposição, que consiste em dividir problemas complexos em partes menores, o reconhecimento de padrões, a abstração de informações relevantes e o desenvolvimento de algoritmos para a criação de soluções passo a passo [Wing 2006].

Expandindo essa visão, destaca-se a proposta tridimensional elaborada por Brennan e Resnick para avaliar o desenvolvimento do PC, com base em estudos desenvolvidos na plataforma Scratch. Esse modelo contempla três dimensões complementares [Brennan and Resnick 2012]:

1. Conceitos computacionais: As ideias centrais utilizadas na programação, como sequências, laços de repetição, paralelismo, condicionais, operadores e dados.
2. Práticas computacionais: As práticas desenvolvidas ao se engajar com os conceitos, como ser iterativo e incremental, testar e depurar, reutilizar e remixar, e abstrair e modularizar.

3. Perspectivas computacionais: As visões sobre o mundo e sobre si mesmo que os aprendizes formam em relação à tecnologia, como a capacidade de se expressar, de construir conexões com outros e de questionar o funcionamento e o impacto das tecnologias.

Para traduzir esses conceitos e práticas em uma experiência de aprendizado engajadora, recorreu-se à gamificação, definida como o uso de mecânicas, estéticas e pensamento baseados em jogos para engajar pessoas, motivar a ação, promover a aprendizagem e resolver problemas [Kapp 2012]. A aplicação de elementos como pontos, níveis, desafios e narrativas em um contexto lúdico tem se mostrado eficaz para aumentar a motivação intrínseca e a persistência dos alunos frente a tarefas complexas, como a programação [Deterding et al. 2011]. A gamificação se alinha a metodologias ativas mais amplas, como a Aprendizagem Baseada em Projetos (PBL), onde os alunos aprendem ao se envolverem ativamente em projetos do mundo real e pessoalmente significativos. Além disso, a abordagem STEAM (*Science, Technology, Engineering, Arts, and Mathematics*) reforça a natureza interdisciplinar desta intervenção, conectando a lógica da programação à criatividade artística.

A estruturação da sequência didática seguiu uma lógica pedagógica consistente, por meio da adoção do Modelo Instrucional 5E [Bybee et al. 2006] para o ensino de ciências, mas amplamente adaptável a outras áreas. Esse modelo apresenta um ciclo de aprendizagem baseado no construtivismo, dividido em cinco fases interligadas que orientam a progressão do conhecimento:

1. Engajar: Captura o interesse dos alunos, ativa o conhecimento prévio e os faz pensar em perguntas sobre o tema.
2. Explorar: Os alunos participam de atividades práticas, explorando os conceitos por si mesmos e colaborando com os colegas.
3. Explicar: O professor guia os alunos a explicarem os conceitos que exploraram, introduzindo formalmente o vocabulário e as definições.
4. Elaborar: Os alunos aplicam o que aprenderam em novos contextos, aprofundando sua compreensão.
5. Avaliar: Tanto os alunos quanto o professor avaliam a compreensão e o progresso, de forma contínua e formativa.

3. Metodologia

Esta pesquisa configura-se como um estudo de caso qualitativo, fundamentado na abordagem da pesquisa-ação [Tripp 2005]. O estudo consistiu no desenvolvimento, aplicação e análise de uma intervenção pedagógica para o ensino do Pensamento Computacional, ajustada iterativamente a partir da observação participante em um contexto educacional real. A pesquisa de campo foi conduzida em uma escola municipal de pequeno porte, localizada na zona rural, com uma turma de estudantes do nono ano do Ensino Fundamental, com idades entre 14 e 16 anos. Uma característica socioeconômica relevante do grupo é que a maioria dos discentes não possuía acesso a computadores em seus domicílios, interagindo com o mundo digital primariamente por meio de celulares. Este cenário informou o delineamento da intervenção, que foi projetada para ser flexível e acessível, independentemente do dispositivo ou do conhecimento prévio dos alunos.

A arquitetura metodológica da intervenção articula três referenciais teóricos. O *Design Thinking* (DT) foi adotado como filosofia norteadora do projeto, guiando desde a fase de Imersão e Empatia para diagnóstico do contexto e das necessidades dos alunos até os ciclos de Ideação e Prototipagem, que corresponderam ao planejamento e refinamento contínuo das atividades [Cavalcanti and Filatro 2016]. A estrutura das sequências didáticas, organizadas em aulas semanais de duas horas-aula, seguiu o Modelo Instrucional 5E (engajar, explorar, explicar, elaborar e avaliar), garantindo uma progressão pedagógica coesa [Bybee et al. 2006]. Por fim, os objetivos de aprendizagem foram definidos com base no *framework* de conceitos, práticas e perspectivas computacionais de Brennan e Resnick (2012).

3.1. Introdução do Pensamento Computacional: atividades desplugadas

O primeiro ciclo da intervenção, alinhado às fases de engajamento e exploração do modelo 5E, foi projetado para introduzir conceitos computacionais básicos por meio de atividades majoritariamente desplugadas. Iniciou-se a fase de engajamento com a aplicação de um diagnóstico para aferir o letramento digital e o acesso tecnológico dos alunos. Em seguida, foram promovidas discussões para conectar o Pensamento Computacional ao cotidiano dos alunos, partindo de questões norteadoras sobre o funcionamento de tecnologias presentes em seu dia a dia, como aplicativos de música, jogos e semáforos. O objetivo era desmistificar a computação, apresentando-a como um fenômeno presente nos mais diversos lugares do cotidiano.

A fase de exploração concentrou-se em atividades desplugadas para materializar conceitos. O conceito de algoritmo, por exemplo, foi trabalhado com a dinâmica do "robô humano", na qual os alunos, atuando como programadores, deveriam fornecer comandos verbais precisos e sequenciais para guiar um colega (o "robô") na execução de uma tarefa. A atividade visava demonstrar a necessidade de clareza, precisão e detalhamento nas instruções de um algoritmo. Além disso, foram organizadas pequenas competições de perguntas e respostas sobre os temas aprendidos em aula (Figura 1).

Figura 1. Competição com perguntas sobre temas da computação.

Posteriormente, introduziram-se outros conceitos como variáveis, condicionais e laços de repetição por meio de analogias (por exemplo, variáveis como "caixas" de alguma informação) e representações visuais, como fluxogramas e pseudocódigos. Tais atividades foram planejadas para desenvolver os Conceitos Computacionais (sequência, condicionais, laços) e introduzir as Práticas Computacionais (testar e depurar) de forma lúdica e concreta, fomentando a Perspectiva Computacional de questionar o funcionamento da tecnologia, conforme o *framework* de Brennan e Resnick (2012).

3.2. Code.org: a ponte entre os conceitos e a prática digital

A transição dos conceitos desplugados para a programação em ambiente digital ocorreu por meio da plataforma Code.org, escolhida por oferecer uma abordagem estruturada, prática e gamificada, alinhada aos objetivos do desenvolvimento do PC. A programação em blocos, livre de erros de sintaxe, permitiu que os alunos focassem diretamente na lógica dos algoritmos e na resolução de problemas. A dinâmica da plataforma, que pode ser observada na Figura 2, com desafios sequenciais de complexidade crescente, serviu de guia para uma aprendizagem progressiva, enquanto sua interface gamificada, com *feedback* visual e imediato, estimulou a experimentação, o teste e a depuração, permitindo aos alunos vivenciar diretamente os pilares do PC, como reconhecimento de padrões e decomposição de problemas.

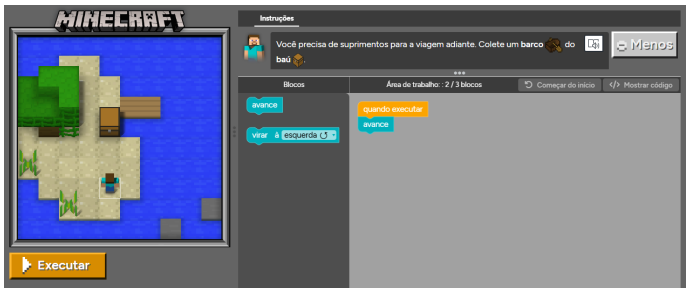


Figura 2. Ambiente de atividades do Code.org

Contudo, o aspecto mais importante da metodologia desenvolvida não foi apenas o uso da ferramenta, mas o método deliberado para garantir que os alunos pudessem transferir o conhecimento do conceitual (desplugado) para o prático (digital), trazendo a noção da aplicação da teoria. Para isso, implementou-se uma técnica de "pausa e reflexão" ao término de cada desafio ou introdução de um novo bloco. Por exemplo, após um exercício que exigia o uso de um laço de repetição, perguntava-se aos alunos "Qual bloco que vocês usaram aqui que se assemelha com o 'Enquanto' que vimos no papel?", ou então "Lembram da atividade do 'robô humano' que utilizava o comando 'Se' para não bater na parede?" para fazer a ligação com um desafio que utilizara um bloco condicional.

Essa prática sistemática, correspondente à fase de explicar do modelo 5E, foi crucial. Ela transformou a interação com a plataforma de uma simples resolução de quebra-cabeças em um exercício consciente de aplicação de conceitos teóricos. Os alunos eram constantemente levados a verbalizar as conexões entre a sintaxe visual dos blocos e a semântica dos conceitos computacionais que já haviam explorado de forma abstrata. Essa ponte explícita solidificou a aprendizagem, garantindo que os conceitos computacionais (laços, condicionais) e as práticas computacionais (depuração, design iterativo), conforme o modelo tridimensional de Brennan e Resnick (2012), fossem não apenas executados, mas compreendidos essencialmente.

3.3. Scratch: desenvolvimento da autonomia junto à expressão criativa

Após a introdução dos conceitos e da prática guida, ocorreu a transição para a plataforma Scratch, com objetivo de promover a transição da aprendizagem estruturada para a criação autônoma e baseada em projetos. A lógica de programação em blocos e a arquitetura

orientada a eventos do Scratch (Figura 3) foram fundamentais para o desenvolvimento das perspectivas computacionais, permitindo que os estudantes expressassem suas ideias, controlassem processos complexos e compartilhassem suas criações digitais, conforme preconiza o modelo de Brennan e Resnick (2012).

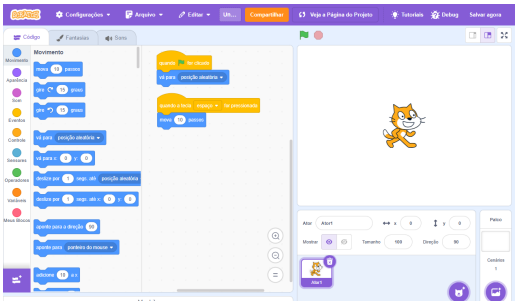


Figura 3. Ambiente de desenvolvimento do Scratch

3.3.1. Projeto para introdução da plataforma: o jogo Pong

Como introdução à plataforma Scratch, foi proposto aos alunos a recriação do jogo Pong (Figura 4), devido à sua simplicidade de comandos e interface. A atividade permitiu a aplicação de conceitos fundamentais, como eventos, movimentação e condicionais, de forma lúdica e acessível. Inicialmente, para a fase de engajamento do modelo 5E, exibiu-se um vídeo do jogo original, seguida de uma discussão guiada por perguntas como "Quais são os componentes principais do jogo?" e "Quando devemos marcar um ponto?", que ajudaram a mapear a lógica e os elementos essenciais do projeto (raquetes, bola, placar), esquematizados coletivamente no quadro.

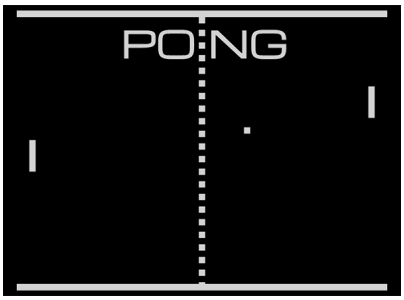


Figura 4. O jogo Pong

A implementação ocorreu de forma iterativa, com cada aula abordando um novo conceito da plataforma Scratch vinculado a uma funcionalidade do jogo. Inicialmente, foi realizada a criação dos sprites (raquetes e bola) e do fundo da cena, seguido pela associação de eventos de teclado para movimentar as raquetes. Em aulas subsequentes, desenvolveu-se a movimentação automática da bola utilizando o evento de início (bandeira verde), detecção de colisões com as raquetes e bordas, e, por fim, o sistema de pontuação, em que blocos de repetição detectavam quando a bola ultrapassava uma raquete, incrementando o placar e reposicionando a bola no centro. Em cada etapa, um conceito novo era introduzido, suas implicações no jogo eram discutidas e os alunos exploravam a aplicação diretamente em seus projetos.

Uma dificuldade significativa observada foi a compreensão do sistema de coordenadas do Scratch. Para resolver essa lacuna, realizou-se uma pausa para uma aula teórica sobre plano cartesiano, com demonstrações práticas na própria plataforma e exercícios de localização de pontos. Essa intervenção fortaleceu a conexão entre conteúdos matemáticos e computacionais, viabilizando o posicionamento adequado dos elementos no palco.

A familiaridade prévia dos alunos com a lógica de construção de projetos na plataforma Code.org facilitou a aplicação do modelo 5E, pois já estavam habituados à sequência de explicação conceitual seguida de exploração prática. Isso contribuiu para um ritmo fluido entre as fases de exploração, explicação e elaboração, permitindo que os alunos construíssem e refinassem seus algoritmos com autonomia. A fase de avaliação ocorreu de forma contínua e individualizada, permitindo identificar e solucionar dificuldades em tempo real. O projeto se mostrou eficaz como estratégia introdutória ao Pensamento Computacional, aliando clareza conceitual, motivação e aplicação prática.

3.3.2. Projeto final: o jogo Geometry Dash

Dando continuidade ao projeto introdutório, foi proposto aos alunos a criação de um jogo mais desafiador como projeto final, que exigisse a aplicação de novos conceitos computacionais e incentivasse maior autonomia criativa. A proposta escolhida foi a recriação de um jogo inspirado no Geometry Dash (Figura 5), um jogo de plataforma com certa familiaridade entre os estudantes, no qual um personagem se movimenta automaticamente para frente e precisa saltar obstáculos e espinhos, evitando colisões. Como referência para a organização da pedagógica das aulas e para implementação das mecânicas, foi utilizado o tutorial disponível no canal griffpatch no YouTube[1].

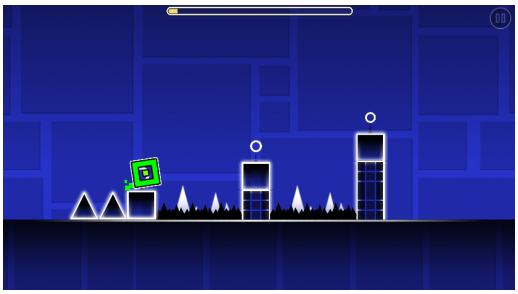


Figura 5. O jogo Geometry Dash.

Para a fase de engajamento, realizou-se uma atividade prática em que alguns alunos foram convidados para jogar a versão original do Geometry Dash, projetando a tela do computador para toda a turma. Enquanto jogavam, os elementos principais que compunham a lógica do jogo eram discutidos com os alunos. Em seguida, questionou-se quais blocos e comandos poderiam ser usados para implementar aquelas funcionalidades no Scratch. A familiaridade com o projeto anterior facilitou a identificação dos componentes e permitiu que os alunos já começassem a sugerir soluções de implementação, como utilizar laços para movimentação contínua, eventos para os pulos e detecção de colisão

[1] https://youtu.be/FYZ1bfB1nho

por cores. No quadro, foram esquematizados os elementos principais do jogo: personagem com movimento automático para frente, comando de pulo, obstáculos fixos, espinhos detectados por cor e uma condição de vitória ao atingir uma certa coordenada no cenário.

Na fase de exploração, os alunos começaram a criar seus próprios personagens, cenários e obstáculos. A personalização do jogo foi fortemente incentivada, e muitos alunos integraram referências pessoais, como Minecraft, Roblox e Free Fire, promovendo um alto grau de engajamento e identidade com o projeto. Nessa etapa, reforçou-se o cuidado com as bordas dos obstáculos e espinhos, pois seriam necessários para a detecção de colisão por cor. Durante esse período, muitos deles começaram, por iniciativa própria, a implementar aspectos funcionais do jogo paralelamente à construção gráfica, o que evidenciou o desenvolvimento das práticas computacionais descritas no modelo tridimensional, como a experimentação incremental, o reuso e a depuração [Brennan and Resnick 2012].

A fase de explicação se deu de forma contínua ao longo das aulas, assim como no desenvolvimento do Pong. A cada nova funcionalidade, o trabalho prático era brevemente interrompido para apresentar conceitos e estratégias de implementação. Entre um dos temas que demonstraram mais dificuldade, foi sobre o chamado “nível infinito”. Anteriormente, os alunos estavam acostumados a mover o personagem na tela, mas agora foi necessário realizar uma inversão: o personagem permaneceria centralizado e o cenário se moveria, criando a ilusão de deslocamento. Para isso, foram criados dois clones do mesmo ator para representar o chão, que se repetiam alternadamente. Quando um dos clones saía completamente da tela, seu x retornava à frente do outro e avançava para a próxima fantasia, criando um ciclo contínuo. Essa lógica foi explicada em termos visuais no quadro, utilizando setas e coordenadas, evidenciando como o controle de visibilidade e movimentação permitia essa rotação. Embora inicialmente confusa, essa lógica foi bem assimilada quando os alunos observaram a funcionalidade em seus próprios projetos.

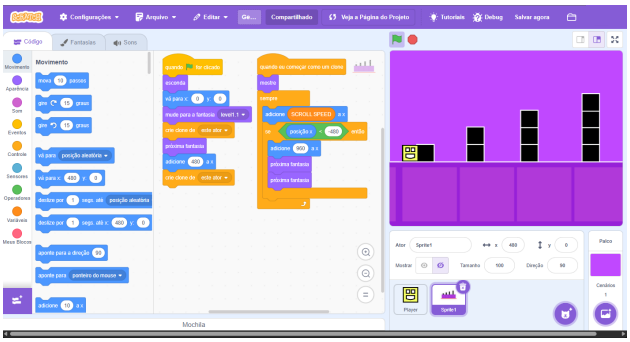


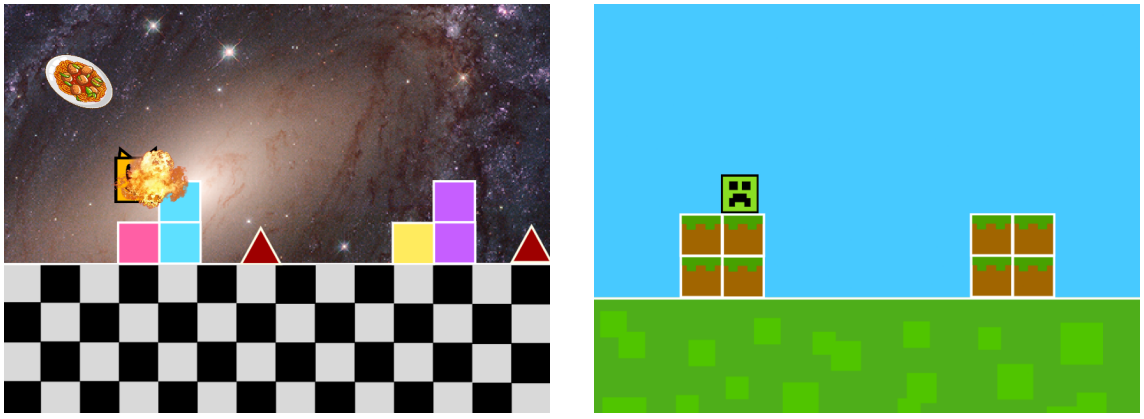
Figura 6. O projeto do jogo Geometry Dash no Scratch.

Durante a fase de explicação, introduziu-se o conceito de aceleração para tornar o pulo mais realista. Inicialmente, o movimento era fixo, com subida e descida a velocidade constante. Para isso, foi apresentado a velocidade como variável que define a mudança de posição e, em seguida, a aceleração como fator que altera essa velocidade com o tempo. Na prática, usaram-se duas variáveis: uma para a velocidade vertical e outra para a gravidade, que era somada a cada ciclo. Isso permitiu que os alunos entendessem, de forma intuitiva, como a aceleração afeta a velocidade e, por consequência, a posição. Com analogias do dia a dia, como um carro acelerando, mesmo sem conhecimento formal em física ou matemática, a maioria compreendeu os efeitos cumulativos das variáveis.

Por fim, a fase de avaliação ocorreu de forma contínua e formativa, como no projeto anterior, mas foi complementada com apresentações formais dos projetos. Cada aluno teve a oportunidade de demonstrar seu jogo para a turma, explicando suas decisões criativas e técnicas. Notou-se que muitos utilizaram tempos livres no laboratório para melhorar seus projetos, incorporando animações e novas mecânicas. A autoavaliação e o *feedback* dos colegas reforçaram o caráter reflexivo da atividade, consolidando o aprendizado não apenas técnico, mas também expressivo, conforme previsto nas dimensões de perspectiva computacional.

4. Resultados e discussão

Após a aplicação da metodologia proposta, obtiveram-se dois resultados centrais. Em primeiro lugar, percebeu-se à evolução do engajamento dos alunos, que avançaram de uma participação inicial motivada pela novidade para níveis mais elevados de autonomia e criatividade. Na etapa final, dedicada à criação de jogos no Scratch, a qual foi concluída por mais de 80% dos alunos da turma, observou-se que muitos estudantes extrapolaram as orientações básicas, desenvolvendo funcionalidades adicionais e personalizando seus projetos com temas de interesse (Figura 7). Esse processo de apropriação do conhecimento refletiu-se, inclusive, na busca espontânea por aprofundamento em programação fora do ambiente escolar, uma vez que 3 alunos relataram que estavam estudando por conta, evidenciando o desenvolvimento de competências ligadas ao protagonismo e à autonomia, o que se alinha às diretrizes da BNCC, que preconizam a formação de sujeitos ativos na construção do próprio saber [Brasil 2022].



(a) (b)

Figura 7. Exemplos dos projetos finais desenvolvidos pelos alunos.

Além disso, observou-se o caráter interdisciplinar, destacando a programação como instrumento de mediação para a aprendizagem de conteúdos de outras áreas. Durante o desenvolvimento dos jogos, os alunos precisaram aplicar conceitos de física, como a gravidade, utilizando variáveis para simular aceleração. Simultaneamente, conceitos matemáticos, como o plano cartesiano, foram mobilizados de forma prática para o posicionamento de personagens e objetos no ambiente digital, melhorando, segundo os alunos, o seu entendimento desses conceitos abstratos ao aplicá-los praticamente. Tal resultado converge com outras pesquisas que utilizaram o Scratch em aulas de Matemática para explorar, por exemplo, o reconhecimento de figuras e simetrias no plano cartesiano, demons-

trando que os alunos constroem o conhecimento matemático de forma mais natural e aplicada quando o utilizam como ferramenta para criar algo significativo [Eggert et al. 2023].

Essas descobertas reforçam a perspectiva da BNCC, que propõe o Pensamento Computacional como uma competência transversal, capaz de articular conhecimentos de diferentes áreas e de fomentar a resolução de problemas de forma criativa e colaborativa. Além disso, alinham-se diretamente à definição do Pensamento Computacional como a capacidade de formular problemas e suas soluções de maneira que um agente computacional, humano ou máquina, possa efetivamente processá-los [Wing 2006].

5. Conclusão

Este trabalho abordou o desafio de operacionalizar o Pensamento Computacional (PC) na educação básica, cuja prática é frequentemente desvinculada de modelos pedagógicos que promovam a integração cognitiva. A resposta a essa lacuna foi a proposição e validação de um modelo metodológico no qual o ciclo instrucional 5E atua como o elemento estruturante para uma sequência de práticas computacionais escalonadas, transitando da conceitualização desplugada à criação em projetos abertos, utilizando plataformas com acesso facilitado.

A principal contribuição do método foi a capacidade dos alunos de aplicar os conceitos abstratos de outras disciplinas, como o sistema de coordenadas cartesiano e o modelo cinemático da física, em jogos eletrônicos funcionais. Esse resultado representa um indicador de transferência de conhecimento, sugerindo que a principal implicação teórica do modelo é sua capacidade de posicionar a programação como uma linguagem para a modelagem de sistemas, em vez de um fim em si mesma, fomentando uma integração cognitiva profunda entre domínios do saber.

Apesar dos resultados positivos, é importante reconhecer que este estudo se baseou em um estudo de caso único, o que limita sua validade externa. Portanto, uma etapa necessária para fortalecer essas evidências é a realização de pesquisas futuras com delineamentos quase-experimentais. Esse tipo de abordagem, que permite comparações entre grupos com e sem intervenção, pode fornecer dados mais robustos sobre os impactos do modelo no desenvolvimento do Pensamento Computacional e na transferência de conhecimentos para outras áreas. Além disso, é fundamental conduzir estudos que permitam observar se os ganhos de aprendizagem se mantêm ao longo do tempo, e investir no desenvolvimento de instrumentos capazes de avaliar se os alunos realmente conseguem aplicar o raciocínio algorítmico na solução de problemas em contextos não computacionais, como matemática, ciências e até situações do cotidiano.

Agradecimentos

Agradecemos à Escola Municipal de Ensino Fundamental Bernardino Fernandes e aos(as) professores(as) pela oportunidade de desenvolver este trabalho em parceria com seus alunos. Também expressamos nossa gratidão ao Programa de Educação Tutorial de Ciência da Computação (PET-CC) e à Universidade Federal de Santa Maria (UFSM), pelo apoio institucional e pelas condições oferecidas para a realização deste projeto.

Referências

Brackmann, C. P. (2017). Desenvolvimento do pensamento computacional através de atividades desplugadas na educação básica.

Brasil (2022). Resolução cne/ceb nº 1, de 4 de outubro de 2022: Normas sobre computação na educação básica – complemento à bncc. Diário Oficial da União, Brasília, DF, 6 out. 2022, Seção 1, p. 33. Conselho Nacional de Educação.

Brennan, K. and Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. In *Proceedings of the 2012 annual meeting of the American educational research association, Vancouver, Canada*, volume 1, page 25.

Brezolin, C. and Silveira, M. (2021). Panorama brasileiro de uso de ferramentas para desenvolvimento do pensamento computacional e ensino de programação. In *Anais do XXIX Workshop sobre Educação em Computação*, pages 398–407, Porto Alegre, RS, Brasil. SBC.

Bybee, R. W., Taylor, J. A., Gardner, A., Van Scotter, P., Powell, J. C., Westbrook, A., and Landes, N. (2006). The bscs 5e instructional model: Origins and effectiveness. *Colorado Springs, Co: BSCS*, 5(88-98).

Cavalcanti, C. C. and Filatro, A. (2016). *Design Thinking na educação presencial, a distância e corporativa*. Saraiva Educação, São Paulo, SP.

Deterding, S., Dixon, D., Khaled, R., and Nacke, L. (2011). From game design elements to gamefulness: defining"gamification". In *Proceedings of the 15th international academic MindTrek conference: Envisioning future media environments*, pages 9–15.

Eggert, K., Asquino, M., and Cruz, D. (2023). Prática pedagógica construcionista com a linguagem de programação scratch em uma abordagem steam. In *Anais do XXIX Workshop de Informática na Escola*, pages 158–168, Porto Alegre, RS, Brasil. SBC.

Gatti, B. A. (2017). Formação de professores, complexidade e trabalho docente. *Revista diálogo educacional*, 17(53):721–737.

Kapp, K. M. (2012). *The gamification of learning and instruction: game-based methods and strategies for training and education*. John Wiley & Sons.

Martins, D., Mota, F., Rocha, M., Santos, C., and Villela, M. L. (2021). O ensino do pensamento computacional nas séries iniciais do ensino fundamental: investigando a percepção docente. In *Anais do XXXII Simpósio Brasileiro de Informática na Educação*, pages 1039–1050, Porto Alegre, RS, Brasil. SBC.

Oliveira, W. and Cambraia, A. (2020). Desafios na formação de professores de computação: Reflexões e ações em construção. In *Anais do XXVI Workshop de Informática na Escola*, pages 319–328, Porto Alegre, RS, Brasil. SBC.

Tripp, D. (2005). Action research: a methodological introduction. *Educação e Pesquisa*, 31(3):443–466.

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3):33–35.