International Journal of STEM Education

RESEARCH Open Access

The impact of AI-assisted pair programming on student motivation, programming anxiety, collaborative learning, and programming performance: a comparative study with traditional pair programming and individual approaches

Guangrui Fan[1*], Dandan Liu[2], Rui Zhang[1] and Lihu Pan[1]

Abstract

Purpose This study investigates the impact of AI-assisted pair programming on undergraduate students' intrinsic motivation, programming anxiety, and performance, relative to both human–human pair programming and individual programming approaches.

Methods A quasi-experimental design was conducted over two academic years (2023–2024) with 234 undergraduate students in a Java web application development course. Intact class sections were randomly assigned to AI-assisted pair programming (using GPT-3.5 Turbo in 2023 and Claude 3 Opus in 2024), human–human pair programming, or individual programming conditions. Data on intrinsic motivation, programming anxiety, collaborative perceptions, and programming performance were collected at three time points using validated instruments.

Results Compared to individual programming, AI-assisted pair programming significantly increased intrinsic motivation ($p < .001$, $d = 0.35$) and reduced programming anxiety ($p < .001$), producing outcomes comparable to human–human pair programming. AI-assisted groups also outperformed both individual and human–human groups in programming tasks ($p < .001$). However, human–human pair programming fostered the highest perceptions of collaboration and social presence, surpassing both AI-assisted and individual conditions ($p < .001$). Mediation analysis revealed that perceived usefulness of the AI assistant significantly mediated the relationship between the programming approach and student outcomes, highlighting the importance of positive perceptions in leveraging AI tools for educational benefits. No significant differences emerged between the two AI models employed, indicating that both GPT-3.5 Turbo and Claude 3 Opus provided similar benefits.

Conclusion While AI-assisted pair programming enhances motivation, reduces anxiety, and improves performance, it does not fully match the collaborative depth and social presence achieved through human–human pairing. These findings highlight the complementary strengths of AI and human interaction: AI support can bolster learning outcomes, yet human partners offer richer social engagement. As AI capabilities advance, educators should integrate

*Correspondence:
Guangrui Fan
fgr@tyust.edu.cn
Full list of author information is available at the end of the article

such tools thoughtfully, ensuring that technology complements rather than replaces the interpersonal dynamics and skill development central to effective programming education.

Keywords AI-assisted pair programming, Intrinsic motivation, Programming anxiety, Collaborative learning, Programming performance

Introduction

Integration of AI in education

The integration of Artificial Intelligence (AI) technologies into educational settings has been rapidly expanding in recent years. This shift represents a significant transformation in educational technology, offering new possibilities for personalized learning, automated support, and enhanced student engagement (Gu & Cai, 2021). AI's potential to augment educational experiences has led to its adoption across various levels of education, from primary schools to universities, with applications ranging from adaptive learning systems to intelligent tutoring platforms (Abdelghani et al., 2022).

In higher education, the integration of AI has been particularly notable. Several universities have begun to allow or even encourage the use of AI-powered coding assistants in computer science courses, recognizing their potential to support student learning and problem-solving skills (Becker et al., 2023). Large language models are being explored for various educational applications, including generating practice questions, providing explanations for intricate concepts, and offering personalized feedback on student work (Denny et al., 2022). These developments suggest a broader shift towards AI-enhanced education, where technology serves as a collaborator in the learning process rather than merely a tool.

However, there is a notable lack of quantitative research examining their impacts, particularly concerning affective factors such as student motivation, anxiety, and perceptions of collaboration. While both early anecdotal reports and several qualitative research studies have provided valuable insights into the potential benefits of AI integration in education (Jiao et al., 2023), large-scale quantitative analyses remain scarce. This gap in the literature is particularly concerning given the significant role that affective factors play in student learning outcomes and overall educational experiences.

Challenges in programming education

Programming education faces significant challenges in maintaining student engagement and managing anxiety levels, which can substantially impact learning outcomes and retention rates in computer science programs. One of the primary issues is the difficulty in sustaining high levels of intrinsic motivation among students. The complex and abstract nature of coding often leads to a decline in motivation as students' progress through their coursework (Lye & Koh, 2014). This decrease can be attributed to the cognitive demands of programming, which require students to simultaneously grasp abstract concepts, apply logical thinking, and master syntax rules (Buitrago Flórez et al., 2017).

These challenges are compounded by the rapid pace at which programming languages and technologies evolve. The constant change can create a sense of uncertainty and inadequacy among learners, potentially diminishing their confidence and enthusiasm for the subject (Demir, 2022). Additionally, the disconnect between theoretical knowledge and practical application in many programming courses can lead to a lack of perceived relevance, further eroding student motivation (Iqbal Malik & Coldwell-Neilson, 2017). Another significant challenge is the prevalence of programming anxiety among students. This anxiety can manifest as fear of failure, apprehension about making mistakes, or a general sense of unease when faced with coding tasks (Chang, 2005). Programming anxiety is particularly problematic as it can create a self-reinforcing cycle where anxiety leads to avoidance, resulting in reduced practice and skill development, which further exacerbates anxiety levels (Cheng et al., 2022). The impact of programming anxiety on learning outcomes is substantial, leading to decreased performance and, in severe cases, contributing to higher dropout rates in computer science programs (Watson & Li, 2014).

Addressing these challenges requires a multifaceted approach that considers both the cognitive and affective dimensions of learning programming. Innovative teaching methods, such as game-based learning and pair programming, have shown promise in enhancing student motivation and reducing anxiety (Wei et al., 2021; Zapata-Cáceres & Martín-Barroso, 2021). Moreover, integrating real-world projects and emphasizing problem-solving skills can help students perceive the relevance of their learning, potentially boosting intrinsic motivation (Malik et al., 2022).

Pair programming as a pedagogical approach

Pair programming, a collaborative software development technique, has gained significant traction as a

pedagogical approach in computer science education. This method involves two programmers working together at a single workstation, with one acting as the "driver" who writes the code and the other as the "navigator" who reviews each line of code as it is typed (Freeman et al., 2003).

One of the primary advantages of pair programming is its positive impact on code quality. Collaboration in real-time allows for immediate error detection and correction, leading to fewer defects and improved overall code structure (Müller, 2004). This continuous peer review process not only enhances the immediate output, but also contributes to the development of critical thinking and problem-solving skills in students. Knowledge sharing is another crucial benefit of pair programming in education. As students work together, they engage in continuous dialogue, explaining their thought processes and strategies. This verbal articulation of ideas facilitates deeper understanding and retention of programming concepts (Beck & Chizhik, 2013). Recent studies have demonstrated the effectiveness of various collaborative approaches in programming education, with Chen and Huang (2024) showing significant improvements in learning performance through Jigsaw-based collaborative programming in virtual environments. Additionally, Ouyang et al. (2022) found that instructor scaffolding in small group programming collaborations can have both immediate and lasting positive effects on learning outcomes. Moreover, the collaborative nature of pair programming fosters a supportive learning environment where students can learn from each other's strengths and experiences. Increased engagement is a notable outcome of implementing pair programming. The interactive nature of this approach helps maintain student focus and motivation throughout the coding process. Studies have shown that pair programming can lead to higher levels of student satisfaction and reduced frustration compared to individual programming tasks (Wei et al., 2021).

One significant challenge of pair programming is the potential mismatch in skill levels between partners. When there is a considerable disparity in programming abilities, the collaboration may become less effective, with the more skilled student potentially dominating the process or becoming frustrated with their partner's pace (Sfetsos et al., 2006). Another notable limitation is the difficulty in scheduling collaborative sessions. Coordinating the availability of two students can be challenging, especially in higher education settings where students often have diverse schedules and commitments (Govender & Grayson, 2006). This scheduling challenge can make consistent implementation of pair programming difficult in traditional educational environments.

AI-assisted pair programming

The advent of advanced language models like GPT and Claude has opened new possibilities for addressing some of the limitations of traditional pair programming while potentially enhancing its benefits. AI-assisted pair programming proposes using AI models as virtual partners, offering a novel approach to collaborative coding in educational contexts. Introducing AI assistants in pair programming could mitigate several challenges associated with human–human pairing (Lau & Guo, 2023). For instance, an AI partner can provide immediate feedback and personalized assistance, effectively playing the role of both driver and navigator as needed (Bird et al., 2022). This capability addresses the issue of mismatched skill levels by offering consistent, high-quality support tailored to the individual student's needs. The flexibility in scheduling is a significant advantage of AI-assisted pair programming. Students can engage with their AI partner at any time, eliminating the need to coordinate with a human partner (Dakhel et al., 2023). Moreover, AI models can provide real-time suggestions, explain concepts, and offer alternative approaches, enhancing the learning experience (Barke et al., 2023). This constant availability of expert-level guidance could significantly boost student confidence and reduce frustration often associated with solo programming.

However, concerns remain regarding its ability to replicate the social and collaborative aspects of human interaction that are valuable in pair programming. Human partners contribute to learning not just through technical assistance, but also through social support, motivation, and the development of communication skills (Williams et al., 2000). An AI partner may not fully replicate these social dimensions. Additionally, there is a notable lack of quantitative research examining their affective and collaborative impacts on students. Specifically, there is limited understanding of how AI-assisted pair programming influences student motivation, programming anxiety, and perceptions of collaboration and social interaction compared to traditional human–human pair programming (Ji et al., 2023). Given the aforementioned challenges and the rapid evolution of AI technologies, there is a critical need for empirical research that examines the affective and collaborative impacts of AI-assisted pair programming. Specifically, gaps exist in: (1) understanding how AI-assisted pair programming impacts students' intrinsic motivation, programming anxiety, and programming performance compared to traditional pair programming and individual programming; (2) assessing how AI-assisted pair programming influences students' perceptions of

collaboration and social interaction during programming tasks compared to human–human pair programming; (3) determining whether AI-assisted pair programming can effectively replicate or compensate for the social and collaborative learning benefits and performance outcomes associated with human–human pair programming; (4) comparing different AI models, such as GPT-3.5 Turbo and Claude 3 Opus, in terms of their affective, collaborative, and performance impacts on students.

To comprehensively understand the impact of AI-assisted pair programming, it is essential to explore not only the direct effects on student motivation, anxiety, and performance, but also the underlying mechanisms that facilitate these outcomes. The Technology Acceptance Model (TAM) (Davis, 1989; Venkatesh & Davis, 2000) provides a robust framework for examining how users come to accept and utilize new technologies. According to TAM, perceived usefulness (PU) and perceived ease of use (PEOU) are critical determinants of technology acceptance and subsequent usage behaviors. In the context of AI-assisted pair programming, PU refers to the extent to which students believe that using the AI assistant enhances their programming performance, while PEOU pertains to the ease with which students can interact with the AI partner. Building on TAM, this study hypothesizes that the perceived usefulness of AI-assisted programming mediates the relationship between the programming approach (AI-assisted vs. traditional methods) and key student outcomes, including intrinsic motivation, programming anxiety, and performance. By investigating this mediation, the study aims to uncover how the effectiveness of AI tools is not only direct, but also operates through students' perceptions of their utility, thereby providing deeper insights into the mechanisms driving the observed outcomes.

Current study

To address the identified gaps, this study aims to investigate how AI-assisted pair programming using GPT-3.5 Turbo and Claude 3 Opus affects undergraduate students' intrinsic motivation, programming anxiety, and perceptions of collaboration and social interaction compared to traditional human–human pair programming and individual programming without assistance. The study also aims to assess whether advancements in AI technology, specifically the transition from GPT-3.5 Turbo to Claude 3 Opus, offer improvements in replicating or compensating for the social and collaborative benefits associated with human interaction in programming education. Additionally, this study explores the underlying mechanisms by which AI-assisted programming influences these outcomes, specifically examining the role of perceived usefulness of AI as a mediator.

To achieve the overarching aim of the study, the following research questions have been formulated:

RQ1: How does AI-assisted pair programming (using GPT-3.5 Turbo and Claude 3 Opus) impact students' intrinsic motivation, programming anxiety, and programming performance compared to traditional human–human pair programming and individual programming without assistance?
RQ2: In what ways does AI-assisted pair programming influence students' perceptions of collaboration and social interaction during programming tasks compared to traditional human–human pair programming?
RQ3: Can AI-assisted pair programming effectively replicate or compensate for the social and collaborative learning benefits and programming performance outcomes associated with human–human pair programming?
RQ4: Are there significant differences in the affective, collaborative, and performance impacts between using GPT-3.5 Turbo and Claude 3 Opus in AI-assisted pair programming?
RQ5: To what extent does the perceived usefulness of the AI assistant mediate the relationship between the programming approach (AI-assisted vs. traditional methods) and student outcomes (intrinsic motivation, programming anxiety, and programming performance)?

Method

Research design and participants

This study employed a quasi-experimental, quantitative design conducted over two consecutive academic years (2023 and 2024) to investigate the impact of AI-assisted pair programming on undergraduate students' intrinsic motivation, programming anxiety, collaborative perceptions, and programming performance. The research was carried out at Taiyuan University of Science and Technology in ShanXi, China, specifically targeting students enrolled in Java web application development course.

Participants were selected through a convenience sampling method from six intact class sections over the two academic years—three classes each year. These classes were chosen based on their alignment with the study's focus on programming and the willingness of instructors to participate. By including entire class sections, we aimed to maintain the ecological validity of the study and minimize disruption to the educational environment (Durlak & DuPre, 2008). Prior to the study, all students

were asked to complete a pre-study survey assessing their demographic information and prior Java programming experience. Students reporting more than 12 months of Java programming experience or prior completion of advanced programming courses were excluded from the final analysis to ensure baseline equivalence across groups. However, these students continued to participate in all class activities to preserve the integrity of the instructional environment.

To assign classes to experimental conditions, we employed cluster randomization at the class level. The three classes each year were randomly assigned to one of the following conditions:

- **Group A:** AI-assisted pair programming (using GPT-3.5 Turbo in 2023 and Claude 3 Opus in 2024).
- **Group B:** human–human pair programming.
- **Group C:** individual programming (control group).

In 2023, the AI-assisted pair programming condition employed the GPT-3.5 Turbo model, and in 2024, the Claude 3 Opus model was introduced. Both models were selected due to their status as state-of-the-art, large language models widely recognized and utilized at the time of data collection. Importantly, both GPT-3.5 Turbo and Claude 3 Opus were accessible to students at no cost via publicly available interfaces, ensuring that participants could engage with these tools without financial barriers.

Randomization was conducted using a random number generator to ensure impartial assignment. This process was supervised by a researcher not involved in data collection to maintain objectivity. To ensure equivalence between groups at baseline, we conducted statistical analyses on key demographic and pre-study variables. One-way ANOVAs and Chi-square tests were used to compare age, gender distribution, prior Java programming experience, and baseline scores on the Intrinsic Motivation Inventory (IMI) and Programming Anxiety Scale (PAS). The results indicated no significant differences among the groups on these measures (all $ps > 0.05$), confirming that the groups were comparable at the outset of the study. The detailed participant distribution and exclusion is presented in Table 1. In addition to the demographic and baseline Java programming experience data, we also collected each participant's Grade Point Average (GPA) from the university's academic records at the start of each semester. GPA was chosen as a proxy for general academic ability and conscientiousness. Preliminary analyses confirmed that GPA did not significantly differ across the groups.

Ethical considerations were prioritized throughout the study. All procedures were approved by the Taiyuan University of Science and Technology's IRB (Protocol No. 20230317), and informed consent was obtained from all participants before data collection. Confidentiality and anonymity were maintained by assigning unique identifiers to participants and securely storing all data in accordance with institutional policies and data protection laws. Participants were informed of their right to withdraw from the study at any time without penalty, as per ethical guidelines for educational research. At the conclusion of each semester, debriefing sessions were conducted to provide participants with information about the study's objectives, preliminary findings, and to address any questions or concerns. To mitigate potential ethical issues related to differential instruction, a percentile-based grading system was implemented, ensuring that students' final grades were determined relative to their peers within the same treatment group.

Table 1 Participant distribution and exclusion across study groups and years

Year	Group	Initial n	Excluded n	Final *n*
2023	A (AI-assisted with GPT-3.5 Turbo)	41	3	38
2023	B (human–human pair programming)	42	2	40
2023	C (individual programming control)	42	4	38
2024	A (AI-assisted with Claude 3 Opus)	41	2	39
2024	B (human–human pair programming)	43	3	40
2024	C (individual programming control)	41	2	39
Total		250	16	234

n number of participants

Instruments and measures

Affective measures

To assess students' intrinsic motivation, we utilized the Intrinsic Motivation Inventory (Ryan, 1982). The IMI is a multidimensional measurement device intended to assess participants' subjective experience related to a target activity. For this study, we employed four subscales: interest/enjoyment, perceived competence, effort, and value/usefulness. The IMI has demonstrated good internal consistency, with Cronbach's alpha coefficients ranging from 0.78 to 0.84 across subscales. Content and construct validity have been established through factor analyses and correlational studies (Deci et al., 1994). The IMI was administered at three time points each year (T1: beginning, T2: mid-semester, T3: end) to track changes in motivation over time.

Programming-specific anxiety was measured using the Programming Anxiety Scale (Yildirim & Ozdener, 2022).

The PAS is a 11-item instrument designed to assess anxiety levels specifically related to computer programming tasks. It has shown high internal consistency (Cronbach's alpha = 0.901) and good construct validity as evidenced by significant correlations with related constructs (Yildirim & Ozdener, 2022). The PAS was administered concurrently with the IMI at all three time points to allow for analysis of potential relationships between anxiety and motivation.

Collaborative and social measures

To evaluate students' perceptions of collaboration and social interaction, we employed two instruments. The Collaborative Learning Scale (So & Brush, 2008) was used to assess perceptions of collaboration, mutual support, and learning from others. The CLS has demonstrated good internal consistency (Cronbach's alpha = 0.72) and construct validity (So & Brush, 2008). The Social Presence Questionnaire (Kreijns et al., 2011) was utilized to measure the degree of perceived social presence during programming tasks. The SPQ has shown excellent reliability (Cronbach's alpha = 0.92) and good construct validity as evidenced by confirmatory factor analysis (Kreijns et al., 2011). Both the CLS and SPQ were administered at T2 and T3 to capture the development of collaborative and social perceptions over the course of the study.

Technology acceptance measures

To assess students' acceptance of AI technology in programming education, we adapted the Technology Acceptance Model (TAM) survey (Davis, 1989; Venkatesh & Davis, 2000). Our modified version focused specifically on the context of AI as a programming partner, incorporating items that reflected students' perceptions of interacting with an AI model rather than a conventional software tool. The original TAM items reference "this product" and "my job". In our adaptation, these were replaced with "the AI assistant" and "my programming tasks" to better reflect the learning context. For example, an original Perceived Usefulness (PU) item, "Using [this product] in my job would enable me to accomplish tasks more quickly", became "Using the AI assistant in my programming tasks enables me to complete coding activities more quickly." Similarly, a Perceived Ease of Use (PEOU) item such as "I would find [this product] easy to use" was adapted to "It is easy to get the AI assistant to understand my programming questions." The detailed modified version is presented in supplementary material S1.

All adapted items were pilot tested with 30 undergraduate students engaged in similar coursework to ensure clarity, contextual relevance, and linguistic appropriateness. The final adapted TAM demonstrated high internal consistency (Cronbach's alpha = 0.88 for PEOU and 0.91 for PU). The survey was administered at T2 and T3 to measure changes in students' technology acceptance over time as they gained experience interacting with the AI programming partner.

Programming performance metrics

To assess students' programming performance, we collected multiple objective measures. These included assignment grades, code quality assessments using established rubrics, and error rates in submitted code. Assignment grades were standardized across groups to ensure comparability. Code quality was assessed by two independent raters using a rubric that evaluated factors such as functionality, efficiency, readability, and adherence to coding standards. Inter-rater reliability was high (Cohen's $\kappa = 0.85$). Error rates were calculated using automated code analysis tools, providing an objective measure of code correctness.

All surveys were originally developed in English. Given that the research was conducted in China, it was necessary to ensure that the instruments were both linguistically and culturally appropriate. To achieve this, we used a forward–backward translation procedure following established guidelines for cross-cultural instrument adaptation. First, two bilingual experts fluent in English and Chinese independently translated the English versions of the surveys into Chinese. A third bilingual expert, who had not seen the original English versions, then back-translated the Chinese versions into English to verify accuracy. Any discrepancies identified during this process were resolved through discussion until a final agreed-upon Chinese version was obtained.

Procedure

The study procedure was designed to ensure a systematic and ethically sound approach to data collection and participant engagement over the course of two academic years. At the outset of each semester, an orientation session was conducted for all participants. During this session, students were introduced to the study's objectives and procedures. Following consent procedures, baseline surveys were administered.

Participants in Group A (AI-assisted pair programming) received additional training on effectively using the AI assistants as programming partners. This training was based on best practices for human–AI collaboration in educational contexts and was tailored to the specific AI model used each year (GPT-3.5 Turbo in 2023, Claude 3 Opus in 2024). Programming sessions were conducted throughout each semester, with participants engaging in their assigned modality: AI-assisted pair programming

(Group A), human–human pair programming (Group B), or individual programming (Group C). Data collection occurred at three primary time points: Time 1 (T1) at the beginning of the semester, Time 2 (T2) at mid-semester, and Time 3 (T3) at the end of the semester. Programming assignments were standardized across all groups to ensure comparability. These assignments were designed in collaboration with course instructors to align with learning objectives and to present equivalent levels of difficulty, following guidelines for task equivalence in programming education research. The assignments were piloted with a separate group of students prior to the study to verify their appropriateness and equivalence.

To minimize the likelihood of participants using external programming tools or unauthorized AI assistants, we conducted all programming sessions in a controlled, supervised environment. In-class activities took place in computer labs where instructors monitored student screens. Participants were informed of an honor code stating that reliance on external coding resources beyond those provided was not permitted. Additionally, online activity logs were periodically reviewed to detect suspicious patterns of code similarity or external consultations. While these measures reduced the risk of external tool usage, we acknowledge that complete prevention is not guaranteed. Students may have accessed unapproved resources outside of class hours, and this remains a limitation that could influence the internal validity of our findings.

Surveys (IMI, PAS, CLS, SPQ, TAM) were administered electronically at T1, T2, and T3 using secure, IRB-approved platforms. The detailed survey administration for different groups in different times is presented in Table 2 below. Programming performance metrics, including assignment grades, code quality assessments, and error rates, were collected after each assignment. For Group A, AI usage logs were recorded, capturing the frequency, duration, and nature of students' interactions with the AI assistants (Table 3).

Since AI-assisted pair programming tools may produce code snippets with unclear licensing origins, we took precautions to ensure that no licensing or copyright violations occurred. All code generated by the AI assistants was reviewed by the course instructors and teaching assistants, who were knowledgeable about common licensing frameworks and open-source code patterns. Any suspiciously specific or known copyrighted code was flagged and removed from the assignment before students were graded. We did not observe any instances of direct code plagiarism or recognized license infringements. Nevertheless, this aspect presents an ongoing challenge in using AI-generated code within educational

Table 2 Summary of survey administration

Survey instrument	Group A (AI-assisted)	Group B (human–human)	Group C (individual)
Intrinsic motivation (IMI)	T1, T2, T3	T1, T2, T3	T1, T2, T3
Programming anxiety (PAS)	T1, T2, T3	T1, T2, T3	T1, T2, T3
Collaborative learning (CLS)	T1, T2, T3	T1, T2, T3	T1, T2, T3
Social presence (SPQ)	T1, T2, T3	T1, T2, T3	T1, T2, T3
Technology acceptance (TAM)	T1, T2, T3	–	–

Table 3 Participant demographics and baseline measures by group

Characteristic	AI-assisted (GPT-3.5)	AI-assisted (Claude 3)	Human–human pair	Individual
N	38	39	80	77
Age (years)	20.342 (1.876)	20.564 (2.103)	20.175 (1.942)	20.416 (2.058)
Gender (% female)	36.8%	33.3%	35.0%	37.7%
Prior Java programming experience (months)	2.237 (3.142)	1.985 (4.676)	2.313 (2.058)	2.182 (1.201)
IMI score	5.127 (0.924)	5.243 (0.837)	5.186 (1.291)	5.092 (0.863)
PAS score	2.876 (0.731)	2.759 (0.795)	2.912 (0.702)	2.834 (0.768)
CLS score	4.823 (0.612)	4.956 (0.687)	4.791 (0.674)	N/A
SPQ score	4.532 (0.783)	4.615 (0.742)	4.478 (0.806)	N/A
TAM score	5.076 (0.715)	5.189 (0.768)	N/A	N/A

Values are presented as mean (SD) unless otherwise noted. IMI Intrinsic Motivation Inventory, PAS Computer Programming Anxiety Scale, CLS Collaborative Learning Scale, SPQ Social Presence Questionnaire, TAM Technology Acceptance Model. All scales range from 1 to 7, with higher scores indicating higher levels of the measured construct, except for PAS where lower scores indicate lower anxiety

contexts, as the transparency of LLM-generated outputs remains limited.

Data analysis

The data analysis was commenced with preliminary procedures to ensure data quality and reliability. This included screening for missing data and outliers using Little's MCAR test and Mahalanobis distance, respectively. Assumptions of normality, linearity, and homoscedasticity were assessed. Internal consistency of all survey instruments were evaluated using Cronbach's alpha, with values above 0.70 considered acceptable (Nunnally & Bernstein, 1994).

To address RQ1 and RQ4, we employed a mixed between-within subjects ANOVA (Guo et al., 2013). Time (T1, T2, T3) served as the within-subjects factor, while Group (AI-Assisted with GPT-3.5, AI-Assisted with Claude 3, Human–Human, Individual) was the between-subjects factor. Post hoc tests using Bonferroni correction were conducted for significant effects. Effect sizes were reported using partial eta-squared (η^2) for ANOVA results and Cohen's d for pairwise comparisons. For RQ2 and RQ3, a MANOVA was performed with CLS and SPQ scores as dependent variables and Group as the independent variable. This was followed by discriminant function analysis to identify key differentiating variables (Huberty & Olejnik, 2006). We conducted a mediation analysis to determine whether the perceived usefulness of AI mediated the relationship between the programming approach (AI-assisted vs. others) and collaborative outcomes (CLS, SPQ) as well as programming performance. Mediation was examined using structural equation modeling (SEM). We employed maximum likelihood estimation with robust standard errors (MLR) to account for potential non-normality in the data. Model fit was assessed with the comparative fit index (CFI), Tucker–Lewis index (TLI), root mean square error of approximation (RMSEA), and standardized root mean square residual (SRMR). Following conventional guidelines (Hu & Bentler, 1999), CFI and TLI values greater than 0.90 indicated acceptable fit, and above 0.95 indicate good fit, while RMSEA values below 0.06 and SRMR values below 0.08 signified acceptable model fit. The decision to use SEM-based mediation was guided by standard practice in educational research, which encourages the examination of indirect effects and underlying mechanisms (Hayes, 2017).

In addition to the primary analyses, we conducted supplementary analyses using analysis of covariance (ANCOVA) to control for potential confounding effects of academic ability. Specifically, we included GPA as a covariate in the ANCOVAs for our primary outcome measures (intrinsic motivation, programming anxiety, and programming performance). By doing so, we aimed to ensure that observed differences between groups were not attributable solely to pre-existing differences in general academic ability. ANCOVA assumptions of homogeneity of regression slopes were tested and met before proceeding with the adjusted analyses. All analyses were conducted using R and *lavaan* package for SEM (Rosseel, 2012), with a significance level set at $\alpha = 0.05$. To address potential issues of multiple comparisons, we employed the false discovery rate correction method (Benjamini & Hochberg, 1995).

Results

Descriptive statistics

The study sample consisted of 234 undergraduate students (Mage = 20.374 years, SD = 1.995) enrolled in Java web application development course. The gender distribution varied slightly across groups, with females representing between 33.3% and 37.7% of each group. Participants reported an average of 5.279 months (SD = 3.169) of prior Java programming experience, with no significant differences between groups, $F(3, 230) = 0.072$, $p = 0.975$. Baseline measures revealed comparable levels of intrinsic motivation (IMI), programming anxiety (PAS), perceived collaboration (CLS), social presence (SPQ), and technology acceptance (TAM) across all groups. One-way ANOVAs indicated no significant differences between groups on these measures at the outset of the study (all $p > 0.05$).

RQ1: Impact on intrinsic motivation, programming anxiety, and programming performance

Changes in motivation, programming anxiety, and programming performance over time are presented in Fig. 1 and Table 4. The average AI-assisted group (combining GPT-3.5 and Claude 3) showed higher intrinsic motivation (M = 5.532, SD = 0.819) at T3 compared to the Human–Human Pair group (M = 5.423, SD = 0.852) and the Individual group (M = 5.237, SD = 1.147). The average AI-assisted group demonstrated lower anxiety levels (M = 2.355, SD = 0.692) at T3 compared to both the Human–Human Pair group (M = 2.643, SD = 0.775) and the Individual group (M = 2.756, SD = 0.741). In addition, the average AI-assisted group showed superior performance (M = 83.901, SD = 10.611) at T3 compared to the Human–Human Pair group (M = 80.123, SD = 11.654) and the Individual group (M = 75.789, SD = 11.234).

A 4 (Group) × 3 (Time) repeated measures ANOVA was conducted to examine the effects of different programming approaches on students' intrinsic motivation, programming anxiety, and programming performance over time (Table 5). For intrinsic motivation, results revealed significant main effects of Time,

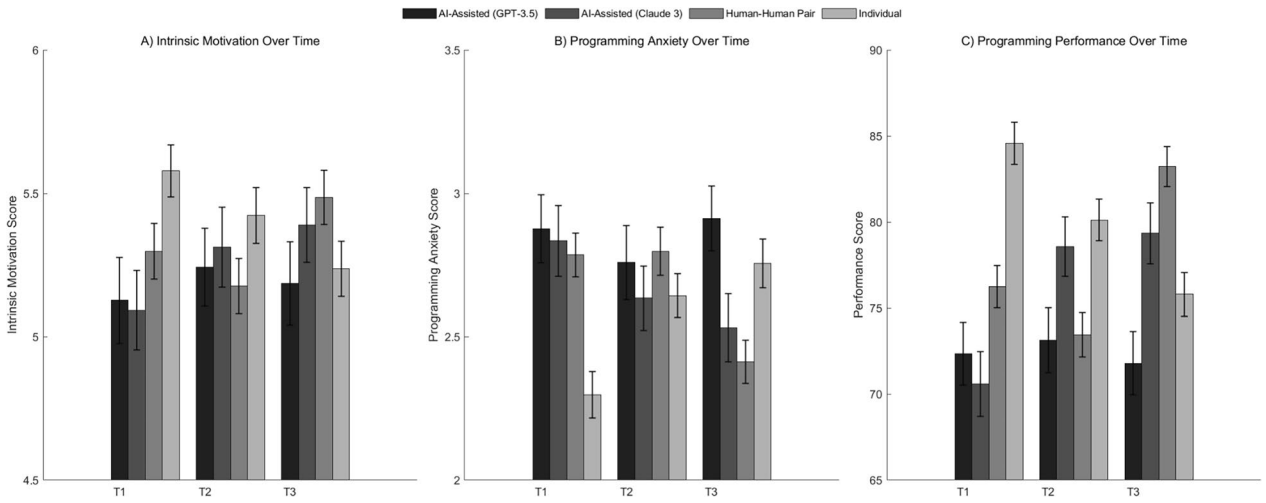

Fig. 1 Changes in motivation, anxiety, and performance over time

Table 4 Mean scores and standard deviations for intrinsic motivation (IMI), programming anxiety (PAS), and programming performance by group and time

Group	Measure	T1	T2	T3
AI-assisted (GPT-3.5)	IMI	5.127 (0.924)	5.312 (0.876)	5.486 (0.843)
	PAS	2.876 (0.731)	2.634 (0.698)	2.412 (0.672)
	Performance	72.345 (11.243)	78.567 (10.876)	83.234 (10.432)
AI-assisted (Claude 3)	IMI	5.243 (0.837)	5.389 (0.812)	5.578 (0.795)
	PAS	2.759 (0.795)	2.531 (0.743)	2.298 (0.711)
	Performance	73.124 (11.567)	79.345 (11.123)	84.567 (10.789)
Human–human pair	IMI	5.186 (1.291)	5.298 (0.867)	5.423 (0.852)
	PAS	2.912 (0.702)	2.785 (0.689)	2.643 (0.775)
	Performance	71.789 (11.345)	76.234 (10.987)	80.123 (11.654)
Individual	IMI	5.092 (0.863)	5.176 (0.859)	5.237(1.147)
	PAS	2.834 (0.768)	2.798 (0.754)	2.756 (0.741)
	Performance	70.567 (11.789)	73.456 (11.543)	75.789 (11.234)

Values are presented as mean (SD). IMI Intrinsic Motivation Inventory, PAS Programming Anxiety Scale. IMI scores range from 1 to 7, with higher scores indicating higher motivation. PAS scores range from 1 to 7, with lower scores indicating lower anxiety. Performance scores range from 0 to 100

$F(2, 460) = 47.623$, $p < 0.001$, $\eta^2 p = 0.171$, and Group, $F(3, 230) = 3.842$, $p = 0.010$, $\eta^2 p = 0.048$, as well as a significant Time × Group interaction, $F(6, 460) = 2.976$, $p = 0.007$, $\eta^2 p = 0.037$. Post hoc analyses using Bonferroni correction indicated that both AI-assisted groups showed significantly higher increases in intrinsic motivation compared to the Individual group ($p_{GPT\text{-}3.5} = 0.008$, $p_{Claude3} = 0.003$).

For programming anxiety, significant main effects were found for Time, $F(2, 460) = 53.189$, $p < 0.001$, $\eta^2 p = 0.188$, and Group, $F(3, 230) = 4.216$, $p = 0.006$, $\eta^2 p = 0.052$, along with a significant Time × Group interaction, $F(6, 460) = 3.354$, $p = 0.003$, $\eta^2 p = 0.042$. Post hoc comparisons revealed that both AI-assisted groups experienced significantly greater reductions in anxiety compared to the Individual group ($p_{GPT\text{-}3.5} = 0.005$, $p_{Claude3} = 0.002$). Programming performance analysis showed significant main effects of Time, $F(2, 460) = 89.765$, $p < 0.001$, $\eta^2 p = 0.281$, and Group, $F(3, 230) = 6.543$, $p < 0.001$, $\eta^2 p = 0.079$, as well as a significant Time × Group interaction, $F(6, 460) = 4.321$, $p < 0.001$, $\eta^2 p = 0.053$. Post hoc tests indicated that both AI-assisted groups and the Human–Human Pair group significantly outperformed the Individual group at T3 (all ps < 0.01), with the AI-assisted groups showing slightly higher performance than the Human–Human Pair group, although this difference was not statistically significant.

Table 5 Repeated measures ANOVA results for intrinsic motivation (IMI), programming anxiety (PAS), and programming performance

Source	df	F	p	η^2p
Intrinsic motivation (IMI)				
Time	2, 460	47.623	< .001	.171
Group	3, 230	3.842	.010	.048
Time × Group	6, 460	2.976	.007	.037
Programming anxiety (PAS)				
Time	2, 460	53.189	< .001	.188
Group	3, 230	4.216	.006	.052
Time × Group	6, 460	3.354	.003	.042
Programming performance				
Time	2, 460	89.765	< .001	.281
Group	3, 230	6.543	< .001	.079
Time × Group	6, 460	4.321	< .001	.053

Df degrees of freedom; η^2p partial eta-squared

The effect sizes (η^2p) suggest moderate to large effects for Time and small to moderate effects for Group and the Time × Group interaction across all measures. These results indicate that AI-assisted pair programming approaches were associated with greater improvements in intrinsic motivation, larger reductions in programming anxiety, and enhanced programming performance compared to individual programming, with effects comparable to or slightly exceeding those of traditional human–human pair programming. In addition to the omnibus ANOVA results presented in Table 5, we conducted Bonferroni-corrected post hoc comparisons to further understand which groups differed significantly on intrinsic motivation, programming anxiety, and programming performance at T3. These post hoc results are summarized in Supplementary Table S1. AI-assisted groups significantly differed from the Individual programming group on both intrinsic motivation and programming performance, while the difference between AI-assisted and human–human pairs was not statistically significant for performance.

Table 6 Correlations between affective measures and programming performance at T3

Measure	1	2	3
1. IMI	–		
2. PAS	– .412**	–	
3. Performance	.523**	– .487**	–

** p < .01. IMI Intrinsic Motivation Inventory, with higher scores indicating greater intrinsic motivation, PAS Programming Anxiety Scale, with lower scores indicating lower anxiety levels. For correlations at T1 and T2, see Supplementary Table S2

Correlation analysis at T3 in Table 6 revealed significant relationships between affective measures and programming performance. Intrinsic motivation was positively correlated with performance (r = 0.523, $p < 0.01$), while programming anxiety was negatively correlated with performance (r = -0.487, $p < 0.01$). These correlations suggest that higher intrinsic motivation and lower programming anxiety are associated with better programming performance.

RQ2: Influence on perceptions of collaboration and social interaction

A one-way multivariate analysis of variance (MANOVA) was conducted to examine the effect of programming approach (AI-Assisted GPT-3.5, AI-Assisted Claude 3, Human–Human Pair, and Individual) on students' perceptions of collaboration and social interaction, as measured by the Collaborative Learning Scale (CLS) and Social Presence Questionnaire (SPQ) at the end of the study period (T3), the result is presented in Table 7.

The MANOVA revealed a significant multivariate effect of programming approach on the combined dependent variables, Wilks' $\lambda = 0.742$, $F(6, 458) = 12.376$, $p < 0.001$, $\eta^2p = 0.139$. This indicates that the programming approach had a significant impact on students' perceptions of collaboration and social interaction. Follow-up univariate ANOVAs showed significant effects of programming approach on both CLS ($F(3, 230) = 23.487$, $p < 0.001$, $\eta^2p = 0.234$) and SPQ ($F(3, 230) = 18.923$, $p < 0.001$, $\eta^2p = 0.198$). The effect sizes suggest that the programming approach had a large effect on perceptions of collaboration and a medium-to-large effect on perceptions of social presence.

Post hoc comparisons using the Tukey HSD test revealed that for CLS, the Human–Human Pair group scored significantly higher than both AI-Assisted groups (pGPT-3.5 = 0.031, pClaude 3 = 0.047), which in turn scored significantly higher than the Individual group (all ps < 0.001). For SPQ, the Human–Human Pair group scored significantly higher than all other groups (all ps < 0.01), while both AI-Assisted groups scored significantly higher than the Individual group (all ps < 0.001).

Table 7 MANOVA and univariate ANOVA results for CLS and SPQ

Test	Statistic	F	df	*p*	η^2p
MANOVA					
Wilks' λ	0.742	12.376	6, 458	< .001	.139
Univariate ANOVAs					
CLS	–	23.487	3, 230	< .001	.234
SPQ	–	18.923	3, 230	< .001	.198

df degrees of freedom; η^2p partial eta-squared, CLS Collaborative Learning Scale, SPQ Social Presence Questionnaire

Table 8 Standardized discriminant function coefficients and structure matrix

Variable	Function 1 coefficient	Structure matrix
CLS	.723	.876
SPQ	.412	.792

Function 1 explained 92.3% of the variance, canonical $R^2 = .387$. CLS Collaborative Learning Scale, SPQ Social Presence Questionnaire

A discriminant function analysis was conducted to identify which variables differentiated between the groups. One discriminant function was statistically significant, $\lambda = 0.742$, $\chi^2(6) = 68.234$, $p < 0.001$, and accounted for 92.3% of the variance in programming approach. The standardized discriminant function coefficients and structure matrix (Table 8) indicate that CLS contributed more to group separation than SPQ, although both variables were important discriminators. The result suggests that while human–human pair programming fostered the highest levels of perceived collaboration and social presence, AI-assisted pair programming also significantly enhanced these perceptions compared to individual programming.

RQ3: Replication or compensation of social and collaborative benefits and programming performance outcomes

The comparison of AI-assisted and human–human pair programming (Table 9) revealed small but significant differences in collaborative learning (CLS), social presence (SPQ), and programming performance. Human–human pairing showed slightly higher CLS (t = -2.143, $p = 0.034$, d = 0.298) and SPQ scores (t = -2.876, $p = 0.005$, d = 0.425), indicating stronger perceptions of collaboration and social interaction. However, AI-assisted programming led to marginally better performance outcomes (t = 2.456, $p = 0.015$, d = 0.355). The result suggests that while AI-assisted pair programming may not fully replicate the collaborative and social benefits of human–human pairing, it compensates through enhanced performance outcomes.

Table 10 Comparison of GPT-3.5 Turbo and Claude 3 Opus on key measures

Measure	GPT-3.5 Turbo	Claude 3 Opus	t	p	Cohen's d
IMI	5.486 (0.843)	5.578 (0.795)	− 0.987	.326	0.112
PAS	2.412 (0.672)	2.298 (0.711)	1.456	.149	0.166
CLS	5.231 (0.682)	5.342 (0.701)	− 1.234	.220	0.160
SPQ	4.876 (0.793)	4.923 (0.812)	− 0.543	.589	0.059
Performance	83.234 (10.432)	84.567 (10.789)	− 1.087	.280	0.125

Values for GPT-3.5 Turbo and Claude 3 Opus are presented as mean (SD). IMI Intrinsic Motivation Inventory; PAS Computer Programming Anxiety Scale; CLS Collaborative Learning Scale; SPQ Social Presence Questionnaire

RQ4: Differences between GPT-3.5 Turbo and Claude 3 Opus

The comparison between GPT-3.5 Turbo and Claude 3 Opus (Table 10) revealed no statistically significant differences across the measured variables. The largest difference was observed in programming anxiety (PAS), with Claude 3 Opus users reporting slightly lower anxiety, but this difference was not significant (t = 1.456, $p = 0.149$, d = 0.166). The small effect sizes (Cohen's d < 0.2 for all comparisons) suggest that the two AI models had similar impacts on students' affective states, collaborative perceptions, and programming performance. These results indicate that the benefits of AI-assisted programming are consistent across different advanced language models, with both GPT-3.5 Turbo and Claude 3 Opus providing comparable support to students in pair programming contexts.

RQ5: The role of perceived usefulness of AI

Mediation analysis (Table 11) indicated that perceived usefulness of AI significantly mediated the relationship between AI-assisted programming and collaborative outcomes (CLS and SPQ) as well as performance. The indirect effects were significant for all three outcomes (CLS: 0.283 [0.159, 0.429], SPQ: 0.258 [0.131, 0.408], performance: 1.976 [0.876, 3.245]), suggesting that the positive impact of AI-assisted programming on collaboration and performance is partially explained by students' perceptions of AI usefulness. The direct effect of AI-assisted

Table 9 Mean differences in CLS, SPQ, and programming performance between AI-assisted and human–human pair groups

Measure	AI-assisted (combined)	Human–human pair	Mean difference	t	p	Cohen's d
CLS	5.287 (0.693)	5.487 (0.654)	-0.200	− 2.143	.034	0.298
SPQ	4.900 (0.803)	5.234 (0.765)	-0.334	− 2.876	.005	0.425
Performance	83.901 (10.611)	80.123 (11.654)	3.778	2.456	.015	0.355

Note. Values for AI-assisted (combined) and human–human pair are presented as mean (SD). CLS = Collaborative Learning Scale, SPQ = Social Presence Questionnaire

Table 11 Mediation analysis: effect of AI-assisted programming on collaborative outcomes and performance through perceived usefulness

Path	b	SE	t	p	95% CI
AI-assisted → perceived usefulness (a)	0.687	0.124	5.540	< .001	[0.443, 0.931]
Perceived usefulness → CLS (b1)	0.412	0.078	5.282	< .001	[0.258, 0.566]
Perceived usefulness → SPQ (b2)	0.376	0.089	4.225	< .001	[0.201, 0.551]
Perceived Usefulness → performance (b3)	2.876	0.745	3.859	< .001	[1.412, 4.340]
AI-assisted → CLS (c'1)	− 0.089	0.092	-0.967	.335	[− 0.270, 0.092]
AI-assisted → SPQ (c'2)	− 0.178	0.106	-1.679	.095	[− 0.387, 0.031]
AI-assisted → performance (c'3)	1.845	0.876	2.106	.037	[0.123, 3.567]

CI confidence interval. Indirect effects (ab) for CLS: 0.283 [0.159, 0.429], SPQ: 0.258 [0.131, 0.408], performance: 1.976 [0.876, 3.245]. CLS Collaborative Learning Scale, SPQ Social Presence Questionnaire

programming on performance remained significant (b = 1.845, $p = 0.037$), indicating partial mediation.

The structural equation model (Supplementary Table S3) demonstrated good fit, supporting the proposed relationships among AI-assisted programming, perceived usefulness, collaborative outcomes, and performance. Although the Chi-square test was significant ($\chi^2(5) = 12.345$, $p = 0.030$), which is common in large samples, other fit indices indicated good model fit (CFI = 0.982, TLI = 0.964, RMSEA = 0.048 [0.015, 0.079], SRMR = 0.028). The model explained 42.3% of the variance in CLS, 38.7% in SPQ, and 45.6% in programming performance, suggesting that AI-assisted programming and perceived usefulness are important factors in predicting collaborative and performance outcomes.

Additional analyses controlling for GPA

To further ensure that differences in intrinsic motivation, programming anxiety, and programming performance were not attributable to pre-existing variations in general academic ability, we conducted ANCOVAs using GPA as a covariate. GPA was included in each model examining the primary outcome measures at the end of the semester (T3). The results are presented in Supplementary Table S4. After controlling for GPA, the group effect on intrinsic motivation remained significant, $F(3, 229) = 3.56$, $p = 0.015$, $\eta^2 p = 0.045$. The AI-assisted groups continued to show higher adjusted mean IMI scores compared to the Individual group, with no substantial reduction in effect size relative to the original ANOVA. Similarly, for programming anxiety, the inclusion of GPA as a covariate did not diminish the group differences observed initially. The adjusted analysis revealed a significant group effect, $F(3, 229) = 3.99$, $p = 0.008$, $\eta^2 p = 0.050$, indicating that AI-assisted groups still exhibited significantly lower anxiety levels than the Individual group. Finally, the advantage of AI-assisted pair programming on performance also persisted after adjusting for GPA, $F(3, 229) = 6.22$, $p < 0.001$, $\eta^2 p = 0.077$. Both AI-assisted groups continued to outperform the Individual programming group, and while the Human–Human Pair condition provided high-quality outcomes, the AI-assisted conditions remained marginally superior, consistent with the initial findings. The results confirm that the observed benefits of AI-assisted pair programming on student motivation, anxiety reduction, and performance enhancement are not simply byproducts of differences in academic ability. Instead, these results highlight the robustness of our main conclusions, demonstrating that even when controlling for GPA, AI-assisted pair programming exerts a positive influence on key educational outcomes.

Discussion

Discussion of findings

RQ1: Impact on intrinsic motivation, programming anxiety, and programming performance

Our findings reveal significant positive impacts of AI-assisted programming on students' intrinsic motivation, programming anxiety, and performance. The increased intrinsic motivation aligns with recent research on AI-enhanced learning environments (Abdelghani et al., 2022), suggesting that AI partners can provide motivational support similar to human instructors or peers. This is particularly relevant in the context of programming education, where motivation has been a persistent challenge (Chittum et al., 2017). The reduction in programming anxiety observed in AI-assisted groups is noteworthy. This effect may be attributed to the non-judgmental nature of AI assistance, allowing students to experiment and make mistakes without fear of social evaluation. Recent studies have highlighted the potential of AI in reducing anxiety in educational settings (Lin & Hou, 2024), and our findings extend this to the domain of programming education. In terms of programming

performance, AI-assisted groups demonstrated slightly superior outcomes compared to both human–human pairs and individual programmers. This finding is consistent with recent research on the benefits of AI-assisted coding (Barke et al., 2023; Peng et al., 2023). The performance enhancement could be due to the immediate, consistent feedback provided by AI, as well as its vast knowledge base, which aligns with the concept of "augmented intelligence" in education (Persico & Pozzi, 2015).

RQ2: Influence on perceptions of collaboration and social interaction

Students' perceptions of collaboration and social interaction with AI partners were positive, albeit not as strong as those in human–human pairs. This finding contributes to the ongoing discourse on social presence in technology-mediated learning (Bull & Kharrufa, 2024). While AI partners were perceived as collaborative entities, the lower social presence scores compared to human partners suggest that current AI technologies may not fully replicate the social-emotional aspects of human collaboration. The comparison with human partners revealed that while AI-assisted programming enhanced collaborative perceptions relative to individual programming, it did not match the level of perceived collaboration in human–human pairs. This aligns with recent research on the limitations of AI in replicating human social interactions (Ji et al., 2023). However, the positive impact on performance despite lower social presence scores suggests that AI assistance may compensate for reduced social interaction through other means, such as consistent availability and rapid response times.

RQ3: Replicating or compensating for social and collaborative benefits and performance outcomes

Our findings indicate that while AI-assisted programming may not fully replicate the social and collaborative benefits of human–human pairing, it compensates through enhanced performance outcomes and perceived usefulness. This compensatory effect aligns with recent research on the effectiveness of AI tools in educational settings (Xu et al., 2023). The mediation analysis revealed that perceived usefulness of AI significantly influenced both collaborative outcomes and programming performance. This finding extends recent research on technology acceptance in AI-assisted learning (Megahed et al., 2024), highlighting the importance of students' perceptions in determining the effectiveness of AI-assisted learning. The strong mediating role of perceived usefulness suggests that fostering positive attitudes towards AI tools may be crucial for maximizing their educational benefits.

RQ4: Differences between AI models

Contrary to our expectations, we found no significant differences in affective, collaborative, or performance outcomes between GPT-3.5 Turbo and Claude 3 Opus. This lack of differentiation may indicate that both models have reached a threshold of capability beyond which incremental improvements do not significantly impact educational outcomes. Alternatively, it could suggest that the current implementation and integration of these AI models in educational settings may be more influential than the specific capabilities of each model. These findings contribute to the ongoing discussion about the pace and impact of AI advancements in education (Jiao et al., 2023). While rapid progress in AI technology is evident, our results suggest that the educational benefits may plateau once a certain level of capability is reached. This has important implications for educational institutions and policymakers considering investments in AI technologies for programming education.

In addition to these immediate outcomes, it is crucial to consider the long-term implications of integrating AI-assisted pair programming into education. While the current study highlights the potential of AI tools to enhance motivation, reduce anxiety, and improve performance, over-reliance on such technologies may inadvertently hinder the development of deeper cognitive and problem-solving abilities. Research in related domains has warned of the risks associated with depending too heavily on AI-generated content. For instance, Abd-Alrazaq et al. (2023) caution that generative AI tools, prone to producing convincing yet fabricated information, could lead learners to overly trust these systems at the expense of honing critical thinking and communication skills. Similarly, Duhaylungsod and Chavez (2023) and Koos and Wachsmann (2023) argue that an excessive reliance on AI for information and content generation can undermine creativity, innovation, and the analytical faculties necessary for constructing logical arguments and integrating diverse sources of knowledge. Santiago et al. (2023) further emphasize that while AI writing aids can improve technical accuracy, they also risk diminishing students' motivation to rigorously research, carefully evaluate sources, and cultivate independent thought. These concerns underscore the importance of integrating AI tools in ways that complement, rather than replace, essential human cognitive processes. Future research should thus explore instructional strategies and pedagogical frameworks that leverage AI's benefits without sacrificing the deeper cognitive engagement and skill mastery that define expert problem-solving in computer science education.

Besides the concern about over-reliance on AI and diminished problem-solving capacity, it is also important

to consider how reliance on these tools might affect the sustained cognitive effort essential for deep learning and long-term skill acquisition. While AI-assisted pair programming can produce immediate gains in task completion speed and initial performance, research on technology use and mental effort (Aru & Rozgonjuk, 2022) suggests that easy access to quick solutions may reduce learners' willingness and ability to engage in prolonged, effortful practice. Aru and Rozgonjuk (2022) argue that habitually relying on technologies that provide immediate rewards and novelty can undermine the motivation to invest in challenging tasks that require sustained cognitive engagement. Applied to programming education, this implies that students who consistently turn to AI tools for instant answers may be less inclined to grapple with complex coding problems independently, potentially stunting the development of core competencies and limiting their capacity to adapt when AI assistance is not readily available. Thus, while AI tools offer short-term productivity benefits, educators and researchers must also anticipate and address the possibility that such convenience might compromise the cultivation of foundational programming competence and the intellectual resilience crucial for long-term success in computer science.

Theoretical implications

The findings of this study have significant implications for theoretical frameworks in educational technology and psychology, particularly Self-Determination Theory (SDT), Social Presence Theory, and the Technology Acceptance Model (TAM). According to SDT (Ryan & Deci, 2020), individuals have basic psychological needs for autonomy, competence, and relatedness that, when satisfied, enhance intrinsic motivation and engagement. Our results suggest that AI-assisted programming environments can effectively support these needs. The increased intrinsic motivation in AI-assisted groups indicates that working with AI partners may enhance students' sense of autonomy and competence. The AI assistant provides immediate feedback and personalized support, allowing students to explore and solve problems independently, thus fulfilling the need for autonomy. The reduction in programming anxiety and improvement in performance reflect an enhanced sense of competence, as students gain confidence in their abilities through successful interactions with the AI assistant. Although the AI-assisted groups reported lower social presence scores compared to human–human pairs, the significant improvements in collaborative perceptions imply that AI partners can partially fulfill the need for relatedness (Cheng et al., 2022). This challenges traditional views within SDT that emphasize human interaction for relatedness satisfaction, suggesting that AI entities can contribute meaningfully to this psychological need.

From the perspective of Social Presence Theory (Garrison, 2016), our findings expand the understanding of social interaction in technology-mediated learning. While AI partners did not fully replicate the social presence of human collaborators, they provided a sufficient degree of perceived social interaction to support collaborative learning and enhance performance. The positive correlation between social presence and programming performance implies that even AI-generated social presence can facilitate learning outcomes, highlighting the evolving nature of social dynamics in educational settings involving AI.

From the perspective of TAM, our mediation analysis revealed that perceived usefulness of the AI assistant significantly mediated the relationship between the programming approach (AI-assisted vs. traditional methods) and student outcomes. This finding underscores the critical role of students' perceptions in embracing AI-assisted tools. The strong mediating effect of PU suggests that when students perceive the AI assistant as useful in enhancing their programming performance, it positively influences their intrinsic motivation and reduces programming anxiety. This aligns with TAM's assertion that perceived usefulness is a primary determinant of technology acceptance and subsequent usage behaviors (Venkatesh & Bala, 2008).

Practical implications

The results of this study offer several practical implications for educational practice, policy, and implementation. In terms of educational practice, our findings support the integration of AI assistants in programming education as a means to enhance student motivation, reduce anxiety, and improve performance. Educators should consider implementing AI-assisted pair programming as a complement to traditional teaching methods, particularly for students who may struggle with anxiety or motivation in traditional programming courses (Xu et al., 2023). However, care should be taken to balance AI assistance with human interaction to ensure that students develop robust collaborative skills.

For policy and implementation at the institutional level, our results suggest that adopting AI technologies in programming education can yield significant benefits. Institutions should consider developing comprehensive strategies for integrating AI assistants into their computer science curricula, including providing training for instructors and support staff (Bull & Kharrufa, 2024). Policies should also address ethical considerations, such

as ensuring equitable access to AI tools and maintaining academic integrity in AI-assisted learning environments.

Best practices emerging from our study include gradually introducing AI assistants to allow for adaptation, explicitly discussing the role and limitations of AI in programming education, and using AI tools to complement rather than replace human instruction. Educators should focus on fostering positive perceptions of AI assistants, as our results indicate that these perceptions significantly influence learning outcomes. Additionally, regular assessment of students' experiences with AI tools can help in fine-tuning their implementation and addressing any emergent issues.

Implementation challenges and potential barriers

While our findings highlight the significant benefits of integrating AI-assisted pair programming into educational settings, we acknowledge that there are potential barriers to its widespread adoption. One major challenge is the resource requirement for incorporating advanced AI tools, which may include financial costs for software licenses or subscriptions, as well as the need for robust technological infrastructure (Bull & Kharrufa, 2024). Educational institutions, particularly those with limited budgets, may find it difficult to allocate funds for these technologies. Another barrier is the potential resistance from educators and students who are unfamiliar with AI tools or skeptical about their effectiveness (Lau & Guo, 2023). Instructors may lack the necessary training to integrate AI assistants effectively into their teaching practices, leading to suboptimal utilization of these tools. Additionally, concerns about data privacy and security can hinder adoption, as AI tools often require access to user data to function optimally (Cotton et al., 2024). There is also the risk of students becoming overly reliant on AI assistance, which may impede the development of fundamental programming skills and critical thinking (Prather et al., 2023). To mitigate these challenges, institutions should consider providing professional development for educators, establishing clear guidelines for AI use, and ensuring that AI tools are integrated in a way that complements rather than replaces traditional teaching methods.

Limitations

While this study provides valuable insights into the impact of AI-assisted pair programming on student outcomes, several limitations should be acknowledged. Firstly, the quasi-experimental nature of the study design introduces potential confounding variables. Despite our efforts to control for factors such as prior Java programming experience, unaccounted variables may have influenced the results. Future research could benefit from randomized controlled trials to establish more robust causal relationships. The generalizability of our findings may be limited by the specific context of the study. Our sample consisted of undergraduate students in a particular educational institution, and the results may not be directly applicable to other educational levels or cultural contexts. Further research is needed to examine the effectiveness of AI-assisted programming across diverse student populations and educational settings (Lau & Guo, 2023). Data collection constraints pose another limitation. While we used validated instruments to measure constructs such as motivation and anxiety, self-report measures are subject to potential biases. Additionally, the measurement of programming performance, while comprehensive, may not capture all aspects of programming skill development. Future studies could incorporate more diverse assessment methods, including qualitative analyses of student code and long-term retention measures. Lastly, the rapid evolution of AI technology means that the specific AI models used in this study (GPT-3.5 Turbo and Claude 3 Opus) were superseded by more advanced systems (GPT-4.0 and GPT-O1). While our results showed no significant differences between these models, future research should continue to evaluate the impact of new AI technologies as they emerge in educational contexts.

Conclusion

This study provides robust evidence for the effectiveness of AI-assisted pair programming in enhancing students' intrinsic motivation, reducing programming anxiety, and improving programming performance. By comparing AI-assisted pair programming using GPT-3.5 Turbo and Claude 3 Opus with traditional human–human pair programming and individual programming approaches, we have demonstrated AI's potential as a valuable collaborator in educational settings.

Our findings indicate that students engaged in AI-assisted pair Java programming experienced a significant increase in intrinsic motivation compared to those in individual programming conditions and achieved performance levels comparable to those in human–human pair programming. This suggests that AI assistants can effectively foster a motivating learning environment through immediate, personalized feedback and support. Additionally, programming anxiety decreased more substantially in AI-assisted groups than in both human–human pairs and individual programming, likely due to the non-judgmental and patient nature of AI assistance.

While human–human pair programming yielded the highest levels of perceived collaboration and social presence, AI-assisted pair programming significantly enhanced these perceptions compared to individual

programming. This indicates that AI assistants can partially replicate the collaborative benefits of human interaction, offering a sense of partnership that positively contributes to the learning experience. The absence of significant differences between the two AI models used suggests that the benefits of AI-assisted programming are consistent across different advanced language models.

The study's results challenge traditional assumptions about the necessity of human-to-human interaction in collaborative learning environments, highlighting AI's potential to serve as an active participant in the learning process. Integrating AI-assisted pair programming into computer science education holds significant promise for improving student outcomes and addressing persistent challenges such as low motivation and high anxiety.

As AI technology continues to advance, educational institutions should consider thoughtful implementation strategies that leverage these tools to complement traditional teaching methods. Future research will explore the dynamics of students' perceptions of AI-assisted programming, aiming to further understand how these perceptions influence educational outcomes and to optimize the integration of AI tools in diverse educational contexts.

Supplementary Information

The online version contains supplementary material available at https://doi.org/10.1186/s40594-025-00537-3.

Additional file 1.

Acknowledgements
We extend our gratitude to the reviewers for their insightful comments that substantially improved the quality of this manuscript. We gratefully acknowledge the collaboration of the students and instructors who participated in this research.

Author contributions
GF made the main contribution to the research, including data collection, topic selection, theme development, and manuscript writing. DL assisted with proofreading, manuscript revision, and response to reviewers. RZ conducted the data analysis and provided proofreading. LP assisted with writing, proofreading, and data collection. All authors read and approved the final manuscript.

Funding
Research Project of Humanities and Social Sciences of the Ministry of Education, China. No. 23YJCZH299.

Availability of data and materials
The datasets used and/or analyzed during the current study are available from the corresponding author on reasonable request.

Declarations

Competing interests
The authors declare that they have no competing interests.

Author details
[1]School of Computer Science and Technology, Taiyuan University of Science and Technology, 66 Waliu Road, Wanbailin District, Taiyuan 030024, Shanxi, China. [2]Department of Media and Communication Studies, Faculty of Arts and Social Sciences, Universiti Malaya, 50603 Kuala Lumpur, Wilayah Persekutuan Kuala Lumpur, Malaysia.

Received: 20 September 2024 Accepted: 21 February 2025
Published online: 04 March 2025

References

Abd-Alrazaq, A., AlSaad, R., Alhuwail, D., Ahmed, A., Healy, P. M., Latifi, S., Aziz, S., Damseh, R., Alrazak, S. A., & Sheikh, J. (2023). Large language models in medical education: Opportunities, challenges, and future directions. *JMIR Medical Education, 9*(1), e48291.

Abdelghani, R., Oudeyer, P.-Y., Law, E., de Vulpillières, C., & Sauzéon, H. (2022). Conversational agents for fostering curiosity-driven learning in children. *International Journal of Human-Computer Studies, 167*, 102887.

Aru, J., & Rozgonjuk, D. (2022). The effect of smartphone use on mental effort, learning, and creativity. *Trends in Cognitive Sciences, 26*(10), 821–823.

Barke, S., James, M. B., & Polikarpova, N. (2023). Grounded copilot: How programmers interact with code-generating models. *Proceedings of the ACM on Programming Languages, 7*(OOPSLA1), 85–111.

Beck, L., & Chizhik, A. (2013). Cooperative learning instructional methods for CS1: Design, implementation, and evaluation. *ACM Transactions on Computing Education (TOCE), 13*(3), 1–21.

Becker, B. A., Denny, P., Finnie-Ansley, J., Luxton-Reilly, A., Prather, J., & Santos, E. A. (2023). Programming is hard-or at least it used to be: Educational opportunities and challenges of ai code generation. In *Proceedings of the 54th ACM Technical Symposium on Computer Science Education V*. 1(pp. 500–506).

Benjamini, Y., & Hochberg, Y. (1995). Controlling the false discovery rate: A practical and powerful approach to multiple testing. *Journal of the Royal Statistical Society: Series B (Methodological), 57*(1), 289–300.

Bird, C., Ford, D., Zimmermann, T., Forsgren, N., Kalliamvakou, E., Lowdermilk, T., & Gazit, I. (2022). Taking Flight with Copilot: Early insights and opportunities of AI-powered pair-programming tools. *Queue, 20*(6), 35–57.

Buitrago Flórez, F., Casallas, R., Hernández, M., Reyes, A., Restrepo, S., & Danies, G. (2017). Changing a generation's way of thinking: Teaching computational thinking through programming. *Review of Educational Research, 87*(4), 834–860.

Bull, C., & Kharrufa, A. (2024). Generative artificial intelligence assistants in software development education: A vision for integrating generative artificial intelligence into educational practice, not instinctively defending against it. *IEEE Software, 41*(2), 52–59.

Chang, S. E. (2005). Computer anxiety and perception of task complexity in learning programming-related skills. *Computers in Human Behavior, 21*(5), 713–728.

Chen, C. M., & Huang, M. Y. (2024). Enhancing programming learning performance through a Jigsaw collaborative learning method in a metaverse virtual space. *International Journal of STEM Education, 11*(1), 36.

Cheng, Y.-P., Shen, P.-D., Hung, M.-L., Tsai, C.-W., Lin, C.-H., & Hsu, L. C. (2022). Applying online content-based knowledge awareness and team learning to develop students' programming skills, reduce their anxiety, and regulate cognitive load in a cloud classroom. *Universal Access in the Information Society, 21*(2), 557–572.

Chittum, J. R., Jones, B. D., Akalin, S., & Schram, Á. B. (2017). The effects of an afterschool STEM program on students' motivation and engagement. *International Journal of STEM Education, 4*, 1–16.

Cotton, D. R., Cotton, P. A., & Shipway, J. R. (2024). Chatting and cheating: Ensuring academic integrity in the era of ChatGPT. *Innovations in Education and Teaching International, 61*(2), 228–239.

Dakhel, A. M., Majdinasab, V., Nikanjam, A., Khomh, F., Desmarais, M. C., & Jiang, Z. M. J. (2023). Github copilot ai pair programmer: Asset or liability? *Journal of Systems and Software, 203*, 111734.

Davis, F. D. (1989). Perceived usefulness, perceived ease of use, and user acceptance of information technology. *MIS Quarterly, 13*(3), 319–340.

Deci, E. L., Eghrari, H., Patrick, B. C., & Leone, D. R. (1994). Facilitating internalization: The self-determination theory perspective. *Journal of Personality, 62*(1), 119–142.
Demir, F. (2022). The effect of different usage of the educational programming language in programming education on the programming anxiety and achievement. *Education and Information Technologies, 27*(3), 4171–4194.
Denny, P., Sarsa, S., Hellas, A., & Leinonen, J. (2022). Robosourcing educational resources-leveraging large language models for learnersourcing. *arXiv*. https://doi.org/10.48550/arXiv.2211.04715
Duhaylungsod, A. V., & Chavez, J. V. (2023). ChatGPT and other AI users: Innovative and creative utilitarian value and mindset shift. *Journal of Namibian Studies: History Politics Culture, 33*, 4367–4378.
Durlak, J. A., & DuPre, E. P. (2008). Implementation matters: A review of research on the influence of implementation on program outcomes and the factors affecting implementation. *American Journal of Community Psychology, 41*, 327–350.
Freeman, S. F., Jaeger, B. K., & Brougham, J. C. (2003). Pair programming: More learning and less anxiety in a first programming course. In *Proceedings of the ASEE Annual Conference*.
Garrison, D. (2016). *E-learning in the 21st century: A community of inquiry framework for research and practice*. Routledge.
Govender, I., & Grayson, D. (2006). Learning to program and learning to teach programming: A closer look. *EdMedia+ Innovate Learning* (pp. 1687–1693). Association for the Advancement of Computing in Education.
Gu, X., & Cai, H. (2021). Predicting the future of artificial intelligence and its educational impact: A thought experiment based on social science fiction. *Educational Research, 42*(05), 137–147.
Guo, Y., Logan, H. L., Glueck, D. H., & Muller, K. E. (2013). Selecting a sample size for studies with repeated measures. *BMC Medical Research Methodology, 13*, 1–8.
Hayes, A. F. (2017). Introduction to mediation, moderation, and conditional process analysis: A regression-based approach. Guilford publications.
Huberty, C. J., & Olejnik, S. (2006). *Applied MANOVA and discriminant analysis*. John Wiley & Sons.
Iqbal Malik, S., & Coldwell-Neilson, J. (2017). Impact of a new teaching and learning approach in an introductory programming course. *Journal of Educational Computing Research, 55*(6), 789–819.
Ji, H., Han, I., & Ko, Y. (2023). A systematic review of conversational AI in language education: Focusing on the collaboration with human teachers. *Journal of Research on Technology in Education, 55*(1), 48–63.
Jiao, J., Chen, L., & Wu, W. (2023). Educational issues triggered by ChatGPT: Possible impacts and counter measures. *Chinese Journal of ICT in Education, 29*(03), 19–32.
Koos, S., & Wachsmann, S. (2023). Navigating the impact of ChatGPT/GPT4 on legal academic examinations: Challenges, Opportunities and Recommendations. *Media Iuris, 6*(2), 255–270.
Kreijns, K., Kirschner, P. A., Jochems, W., & Van Buuren, H. (2011). Measuring perceived social presence in distributed learning groups. *Education and Information Technologies, 16*, 365–381.
Lau, S., & Guo, P. (2023). From" Ban it till we understand it" to" Resistance is futile": How university programming instructors plan to adapt as more students use AI code generation and explanation tools such as ChatGPT and GitHub Copilot. In *Proceedings of the 2023 ACM Conference on International Computing Education Research-Volume 1*. 106–121.
Lin, Y.-C., & Hou, H.-T. (2024). The evaluation of a scaffolding-based augmented reality educational board game with competition-oriented and collaboration-oriented mechanisms: Differences analysis of learning effectiveness, motivation, flow, and anxiety. *Interactive Learning Environments, 32*(2), 502–521.
Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41*, 51–61.
Malik, S. I., Mathew, R., Al-Sideiri, A., Jabbar, J., Al-Nuaimi, R., & Tawafak, R. M. (2022). Enhancing problem-solving skills of novice programmers in an introductory programming course. *Computer Applications in Engineering Education, 30*(1), 174–194.
Megahed, F. M., Chen, Y.-J., Ferris, J. A., Knoth, S., & Jones-Farmer, L. A. (2024). How generative AI models such as ChatGPT can be (mis) used in SPC practice, education, and research? An Exploratory Study. *Quality Engineering, 36*(2), 287–315.
Müller, M. M. (2004). Are reviews an alternative to pair programming? *Empirical Software Engineering, 9*(4), 335–351.
Nunnally, J., & Bernstein, I. (1994). *Psychometric Theory* (3rd ed.). MacGraw-Hill.
Ouyang, F., Dai, X., & Chen, S. (2022). Applying multimodal learning analytics to examine the immediate and delayed effects of instructor scaffoldings on small groups' collaborative programming. *International Journal of STEM Education, 9*(1), 45.
Peng, S., Kalliamvakou, E., Cihon, P., & Demirer, M. (2023). The impact of ai on developer productivity: Evidence from github copilot. *arXiv*. https://doi.org/10.48550/arXiv.2302.06590
Persico, D., & Pozzi, F. (2015). Informing learning design with learning analytics to improve teacher inquiry. *British Journal of Educational Technology, 46*(2), 230–248.
Prather, J., Denny, P., Leinonen, J., Becker, B. A., Albluwi, I., Craig, M., Keuning, H., Kiesler, N., Kohn, T., & Luxton-Reilly, A. (2023). The robots are here: Navigating the generative ai revolution in computing education (*Proceedings of the 2023 Working Group Reports on Innovation and Technology in Computer Science Education*. 108–159.
Rosseel, Y. (2012). lavaan: An R package for structural equation modeling. *Journal of Statistical Software, 48*, 1–36.
Ryan, R. M. (1982). Control and information in the intrapersonal sphere: An extension of cognitive evaluation theory. *Journal of Personality and Social Psychology, 43*(3), 450.
Ryan, R. M., & Deci, E. L. (2020). Intrinsic and extrinsic motivation from a self-determination theory perspective: Definitions, theory, practices, and future directions. *Contemporary Educational Psychology, 61*, 101860.
Santiago, C. S., Jr., Embang, S. I., Conlu, M. T. N., Acanto, R. B., Lausa, S. M., Ambojia, K. W. P., Laput, E. Y., Aperocho, M. D. B., Malabag, B. A., & Balilo, B. B., Jr. (2023). Utilization of writing assistance tools in research in selected higher learning institutions in the Philippines: A text mining analysis. *International Journal of Learning, Teaching and Educational Research, 22*(11), 259–284.
Sfetsos, P., Stamelos, I., Angelis, L., & Deligiannis, I. (2006). Investigating the impact of personality types on communication and collaboration-viability in pair programming–an empirical study. *International conference on extreme programming and agile processes in software engineering* (pp. 43–52). Berlin Heidelberg: Springer Berlin Heidelberg.
So, H.-J., & Brush, T. A. (2008). Student perceptions of collaborative learning, social presence and satisfaction in a blended learning environment: Relationships and critical factors. *Computers & Education, 51*(1), 318–336.
Venkatesh, V., & Davis, F. D. (2000). A theoretical extension of the technology acceptance model: Four longitudinal field studies. *Management Science, 46*(2), 186–204.
Venkatesh, V., & Bala, H. (2008). Technology acceptance model 3 and a research agenda on interventions. *Decision sciences, 39*(2), 273-315.
Watson, C., & Li, F. W. (2014). Failure rates in introductory programming revisited. In *Proceedings of the 2014 conference on Innovation & technology in computer science education*. 39–44.
Wei, X., Lin, L., Meng, N., Tan, W., & Kong, S.-C. (2021). The effectiveness of partial pair programming on elementary school students' computational thinking skills and self-efficacy. *Computers and Education, 160*, 104023.
Williams, L., Kessler, R. R., Cunningham, W., & Jeffries, R. (2000). Strengthening the case for pair programming. *IEEE Software, 17*(4), 19–25.
Xu, W., Wu, Y., & Ouyang, F. (2023). Multimodal learning analytics of collaborative patterns during pair programming in higher education. *International Journal of Educational Technology in Higher Education, 20*(1), 8.
Yildirim, O. G., & Ozdener, N. (2022). The development and validation of the programming anxiety scale. *International Journal of Computer Science Education in Schools, 5*(3), 17–34.
Zapata-Cáceres, M., & Martín-Barroso, E. (2021). Applying game learning analytics to a voluntary video game: Intrinsic motivation, persistence, and rewards in learning to program at an early age. *IEEE Access, 9*, 123588–123602.

Publisher's Note