AI in CS Education: Transformative Interventions and Research Frameworks *

Giulia Paludo[1], Agnese Del Zozzo[1], Francesca Fiore[1], and Alberto Montresor[1]

Department of Computer Science and Information Engineering, University of Trento
giulia.paludo@unitn.it; agnese.delzozzo@unitn.it;
francesca.fiore@unitn.it; alberto.montresor@unitn.it

Abstract. In recent years, the educational landscape has been caught off guard by the rapid rise of generative AI technologies, which students are increasingly adopting on their own, often without formal guidance. Educators, unprepared for these swift advancements, now face the challenge of navigating a spectrum between two extremes: from avoiding AI altogether to thoughtfully integrating it into their pedagogical practices. Educational research now has the crucial task of providing educators with practical, discipline-specific strategies to use AI effectively, helping them transform what might seem like a crisis into an opportunity for enriched, innovative learning.

Computer science is particularly impacted by the integration of AI, given the high-quality code generated by advanced models like ChatGPT and Codex. Now embedded directly within many software development suites, these tools are transforming professional coding practices, enhancing efficiency and quality. However, they also impact how students learn to code, presenting both challenges and opportunities. While these technologies have the potential to serve as tutors, providing immediate feedback, there is also a risk that overreliance could hinder deep learning. This shift calls for an adaptation in teaching practices with a dual focus: preparing students to use AI tools thoughtfully and critically, while fostering a strong foundation in core programming skills in the age of AI. On both fronts, cultivating metacognitive skills is essential to ensure that generative AI serves to enhance rather than replace fundamental coding abilities. This balanced approach fosters a learning environment where automation complements, rather than undermines, critical and hands-on coding expertise.

This paper makes a twofold contribution. First, it proposes an educational intervention focused on helping advanced students understand both the advantages and potential pitfalls of using AI in programming tasks. This intervention emphasizes critical thinking, enabling students to navigate AI tools with discernment. Second, we introduce a novel comprehensive, multimodal data collection methodology to analyze how students interact with generative AI in coding contexts. This approach provides a robust foundation for evaluating student-AI interactions and refining future pedagogical interventions. Preliminary findings from our first pilot offer encouraging insights, setting the stage for more detailed

* Supported by Euregio

analysis and contributing valuable information for educators seeking to understand and improve the role of AI in computer science education.

1 Introduction

Context and Problem Statement The sudden emergence of accessible generative AI tools is significantly disrupted education, with students increasingly using these resources independently, often without structured guidance from educators. This trend places educators at a critical juncture, where they must navigate a spectrum between two extremes: sidestepping AI entirely, relying solely on traditional approaches, or thoughtfully integrating these tools into their teaching practices [9]. To support this transition, educational researchers now face an urgent task–to develop effective, discipline-specific strategies that responsibly harness AI's potential, transforming what might seem like a challenge into an opportunity for more innovative, enriched learning experiences [21].

This is especially relevant for computer science (CS) education, particularly in in the context of learning programming. General models like ChatGPT and specialized programming models such as Codex, capable of generating high-quality code, are now embedded in many Integrated Development Environments (IDEs) used by professionals and students alike [10]. These tools offer increased efficiency and enhanced code quality, yet they also present challenges for learning. On one hand, AI tools can serve as interactive tutors, providing immediate feedback and guidance [19]. However, if students rely too heavily on these tools, they may struggle to build a solid grasp of essential coding skills, which are crucial for independent problem-solving and for fully understanding AI-generated responses [9].

This scenario brings both promising opportunities and significant challenges to CS educational researchers. AI opens the door to rethinking programming education, allowing educators to explore new pedagogical approaches, content, and instructional methods [3]. However, the lack of a well-established research foundation in this area makes it difficult to assess AI's impact on student learning or to create a research framework that thoroughly examines these effects. Programming itself is a uniquely complex cognitive task, requiring not only technical problem-solving but also elements of verbal reasoning, creativity, and iterative thinking [16]. Current methods for studying learning in computer science are not fully equipped to capture this complexity or to address the nuanced ways in which generative AI impacts learning and metacognition.

Research Questions and Objectives This study addresses two primary objectives at the intersection of educational practice and research methodology. On a practical level, the educational challenge lies in finding effective ways to integrate AI into the learning process while ensuring that students continue to build strong programming skills. This includes developing pedagogical strategies that discourage overreliance on AI, prevent potential academic misconduct, and provide fair assessments of student learning.

On a methodological level, our research aims to fill gaps in current approaches to studying how students interact with AI, especially regarding metacognitive processes in computer science education. Rather than relying solely on traditional task-based assessments, we employ a multimodal data collection framework to capture a comprehensive view of student interactions with AI, including the cognitive strategies they use.

This study addresses two core objectives:

- Implementing a practical intervention to help students critically evaluate AI-generated code, fostering their ability to learn from and assess AI-supported strategies;
- Developing a multimodal data collection framework to analyze student interactions with AI, gaining insights into both learning processes and metacognitive aspects.

By addressing these objectives, this study seeks to advance both pedagogical practice and research methodologies related to AI in computer science education, contributing to a deeper understanding of how students can learn effectively within an AI-enhanced educational landscape.

Methods To address these concerns, we conducted a two-day intervention designed as a learning experience for senior students, incorporating a structured data collection procedure. Building on the work of Ojeda et al. [22], this activity involved 39 students, organized into 13 groups of three, who were tasked with solving programming and algorithmic exercises under specific conditions: only AI-generated code could be used, the number of AI interactions was limited. This approach encouraged learning by requiring students to explicitly outline the steps in their solutions [16] and to apply critical thinking in evaluating the AI-generated code [22].

For data collection, we aimed to gain a comprehensive understanding of the learning processes involved in this experience by using a multimodal approach, which included:

- *Pre- and Post-Activity Survey:* These surveys assessed students' problem-solving approaches, self-awareness, and metacognitive skills before and after the activity [24], as well as their reflections on perceived learning outcomes.
- *Log Analysis:* Students engaged with various AI engines through a dedicated platform designed to log all interactions with AI during the challenge. This data allowed us to analyze the progression of their prompting strategies.
- *Audio and Video Recording of Student Activity:* We recorded sessions for two groups, with one setup capturing screen and vocal interactions only (webcam off), and the other capturing both screen and webcam-enabled interactions. This multimodal data integration enabled us to link logged interactions with AI to the intermediate steps students took during the learning process. We followed the analytical methodology of Powell et al. [25] for video analysis.

Outcomes The initial pilot study using the proposed approach, apart from testing technical equipment and identifying constraints, demonstrated a positive impact on student understanding of the role of AI in programming and collected comprehensive data to analyze their learning experience. This intervention encouraged students to engage deeply in the practice of explaining and discussing their reasoning before moving to the final solution. It also provided researchers with clearer insights into students' problem-solving approaches. The combination of exercise results, logs, and audio-video recordings offered a multimodal view of the learning process, which was further contextualized by pre- and post-activity surveys, enabling an in-depth study of students' reasoning development.

This proposal, merging AI-powered programming sessions with a comprehensive data collection procedure, provides a promising approach to support research on the learning processes within this discipline. Framed within the concept of AI as "Intelligence Augmentation" [14], this approach encourages a shift in focus from mere code writing to the essence of problem-solving, thereby enhancing students' awareness and critical thinking.

By building on the results of the initial pilot, we plan to implement this procedure in laboratory classes for first-year undergraduate CS students during the 2024-25 academic year. This will allow for further validation of the methodology and provide deeper insights into programming learning within this approach. Additionally, the versatility of this data collection methodology suggests potential applications in other fields, such as language teaching, where it could similarly enhance our understanding of student learning dynamics.

2 Background and Literature Review

This section reviews the literature on the transformative impact of AI in computer science education, focusing on both the opportunities and challenges it introduces. It then examines methodological gaps in CS education research, highlighting the interdisciplinary complexities and the need for more structured approaches. Comparisons with math education provide additional insights into potential research methodologies. Finally, the section discusses metacognition's role in learning, underscoring its importance for developing self-regulated, adaptive problem-solving skills in students.

2.1 AI's Impact in Computer Science Education

Generative AI has the potential to transform nearly every discipline, and computer science is among the most affected by the introduction of AI-generated code. Since the inception of computer science, *writing code* has been central as a core competency. The introduction of AI assistants specifically trained for coding has sparked significant debate about the foundations of the profession.

AI is rapidly improving the quality of generated code, with tools like ChatGPT and Codex now capable of replicating the performance of top students in

programming tasks [12, 6]. This advancement is driving the integration of such tools into IDEs to enhance efficiency.

AI-generated code has caught the educational system off guard, leading many teachers to perceive these tools as a threat to learning quality [9]. This raises a significant challenge in training future professionals: how can AI tools be integrated while still maintaining adequate emphasis on core technical skills? This challenge is also tied to the existing issue of balancing academic training with job market demands [19, 7]. Compared to the past, there is now a greater need for solutions that balance AI use with core skills while preparing students for AI integration in the tech industry.

The initial proposals toward integrating AI in computer science education focus on tutoring, using systems that employ chatbots to provide feedback and suggestions on students' code [10]. While formal education incorporates AI as smart tutoring systems that emphasize AI-driven feedback [7], students primarily use AI individually for typo correction, error message interpretation, and additional support during debugging sessions.

2.2 Methodological Gap in Computer Science Education Research

Computer science education research emerged in the late 1980s and has been characterized by a lack of its own specific methodologies, as well as rigor and consensus on effective research strategies. The intersection of education, pedagogy, psychology, technology, engineering, and computer science leads to high complexity and reveals the need for solid methodological foundations. As Almeida points out [1], CS education publications often lack empirical rigor, coherent frameworks, and employ fragmented methodologies.

Reviews of CS education research have highlighted these issues, showing that poorly designed studies, ill-defined hypotheses, and questionable data collection and analysis practices are prevalent [26]. Furthermore, Lishinski et al. [20] revealed the paradox that, despite growing interest in the learning processes of CS education, overall methodological quality has not significantly improved.

While efforts are underway to develop more robust methodological guidelines [1], the field remains underexplored. CS education could benefit from and established research practices from other disciplines, like mathematics education, which has a longstanding tradition of using qualitative methods to analyze learning processes.

In particular, mathematics education has used video recordings of student activities for decades as a valuable data source for understanding learning dynamics. Reflecting on his research, Schoenfeld [27] describes how he began using videotape in the late 1970s to study mathematical problem-solving:

> "[...] there were aspects of problem-solving behavior that could not be captured any other way. These included references to specific drawings as students worked on them ("What if we draw the line from here to here"), the evolution of those figures, signs of shared attention or independent processing, and more. Perhaps most importantly, many aspects

> of exchanges are fleeting and ephemeral; if I didn't "capture" them on tape, they would be lost forever, subject to faulty memory, and more. It was not that videos would be my only source of data, but rather that they offered data that were irreplaceable" (p. 416).

Schoenfeld's reflections underscore the unique insights that video data provide– insights that can also inform methodological choices in CS education research, where understanding problem-solving activities is similarly crucial.

2.3 Metacognition: Theory, Practice and Research

Metacognition refers to higher-order processes involved in "*thinking about thinking*" [8]. It consists of two main components: reflection on one's cognitive functions (knowledge of one's cognitive processes) and the processes for monitoring such functions [24].

Metacognition, in brief, is the interplay between beliefs or knowledge about the mind and regulation strategies. Although it may seem like a purely theoretical construct, metacognition has significant implications for academic performance [17]. In the context of learning, key aspects of metacognition include being aware of one's reasoning while addressing tasks and attempting to adjust behavior accordingly. Over time, this helps build a system of heuristics and approaches that allow for more efficient problem-solving [17].

Academic performance significantly benefits from actively using metacognitive strategies, which can be promoted through specific activities [11]. For instance, while recalling and generalizing previously used strategies is common, stronger metacognitive skills enhance the ability to store, retrieve, compare, and apply appropriate strategies. Fostering these processes not only improves learning quality but also supports key skills like self-directed learning.

Metacognition facilitates a shift from surface to deep learning, as evidenced by correlations between metacognitive skills and learning approaches [13]. This shift is not only about improving learning quality but also enhancing efficiency and well-being: students who rely on surface approaches often report a higher workload and increased pressure [13].

Like other *higher-order thinking skills* (HOTS) such as problem-solving and critical thinking [29], metacognition presents several challenges for scientific inquiry. Due to its complexity, there is a lack of effective tools, especially for assessing metacognition during the learning process. Assessing metacognition in this context requires understanding how students reflect on their reasoning, approaches, and strategies for tackling tasks. Although the scientific literature offers assessment scales (e.g.,[4]) and commonly used methods, such as reflective practices and think-aloud protocols to monitor logical thinking[11], these approaches often fail to capture the full depth of the metacognitive process.

In summary, while metacognition is valued as a skill for its impact on academic performance, its assessment and enhancement through effective practices remain in early stages, with many pointing to the lack of standardized assessments and validated practices. Building on these considerations, we aim to study

the activity of writing code within the context of problem-solving, where interacting with AI according to a given protocol is structurally necessary. To this end, we have developed a multimodal data collection methodology that provides insights into metacognitive processes.

3 Methodological Proposal

To address specific methodological challenges in researching AI integration in CS education, we developed a structured protocol that investigates two core research questions:

- *Validating AI integration in programming education*: How can AI interventions in programming education advance beyond tutoring and feedback?
- *Validating a research methodology*: Can an AI-driven activity provide an effective context for observing learning and metacognitive processes in action?

To this end, we propose an educational intervention alongside a multimodal data collection procedure, both of which are detailed in the following sections.

3.1 Overview of the Intervention

The proposed intervention, called *Hackaprompt*, is a two-day coding challenge in which students solve programming exercises using only AI-generated code on a platform that provides access to various large language models (LLMs). Hackaprompt combines aspects of a traditional curricular lab with a challenge-based activity format [18]. The exercises vary in difficulty, ranging from simple programming tasks to complex algorithmic and parallel programming challenges. To encourage precise and effective prompting, each exercise limits both the number of queries and the amount of text allowed.

Setting and Platform Students participating in the challenge are divided into small groups of three or four, working collaboratively on the assigned exercises using a single computer connected to a custom-designed Hackaprompt platform via a web interface (see Figure 1). Group work encourages peer learning and allows students to discuss and refine their approach to each exercise collectively, which enhances the problem-solving process by integrating multiple perspectives. This setup not only mirrors real-world collaborative coding environments but also provides a rich context for observing how students collectively navigate challenges and leverage AI tools.

The platform offers direct access to multiple large language models (LLMs), including various versions of ChatGPT and LLAMA, and logs each interaction between the groups and the LLMs, capturing both prompts and responses. Each group receives a PDF file with a set of problems and exercises to be solved using the web interface.

For some groups, the setup is enhanced with webcams to capture video and audio, documenting group interactions and collaborative dynamics. These groups

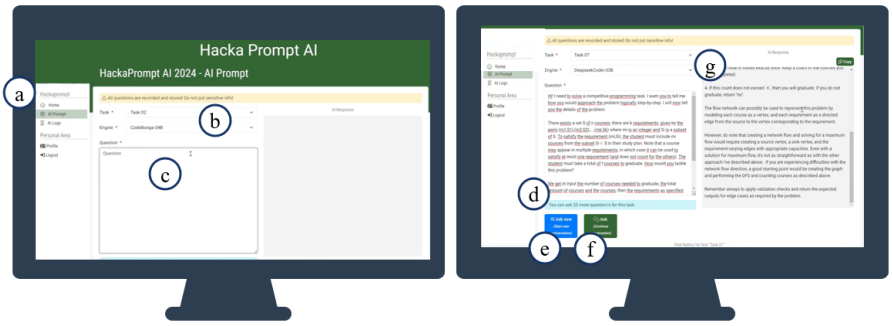

Fig. 1. Hackaprompt's Platform User Interface (UI).

also join separate Zoom rooms with screen sharing enabled, allowing for simultaneous recording of both group conversations and on-screen activity. This multimodal data collection enables us to link logged interactions with AI to the intermediate steps students take during the learning process, providing a comprehensive view of the problem-solving dynamics and collaborative strategies that unfold within the group setting.

The minimalist web interface (Figure 1) features a menu with options for "AI Prompt" and "AI Logs" (a), a prompt input area (c) with a drop-down menu to select the exercise and LLM model (b). Upon submitting a prompt, the response appeared in a gray area on the right (g), and buttons are available to start a new conversation for a different exercise (e) or clear the input area for a follow-up query (f). Additionally, a turquoise box (d) displayed the remaining query count, updated after each submission.

Problems and Rules The problems and exercises provided to students are designed with a structured, progressive difficulty to guide skill development in stages. The Hackaprompt challenge consist of five levels, each introducing increasingly complex tasks:

- *Warmup and programming fundamentals*: These introductory tasks focus on foundational programming skills, allowing students to familiarize themselves with the platform and the use of large language models (LLMs) for generating code.
- *Algorithms and data structures*: At this level, students tackle problems that involve standard algorithms and data structures, building on the fundamentals and requiring a higher level of logical reasoning.
- *Olympics-based challenges*: Inspired by competitive programming, these exercises encourage students to develop efficient and optimized solutions.
- *Parallel computing*: This level introduces tasks involving parallel computing concepts, which required the use of specialized libraries and concepts that many students are unfamiliar with.

– *Extreme difficulty*: The final level may include a single, extremely challenging problem, intended as a buffer to prevent students from completing all the problems before the end of the event.

Each exercise imposes a strict limit on the number of queries and the text length allowed per problem. These caps encourage students to formulate concise, purposeful prompts, discouraging excessive or unreflective use of AI. This approach also mirrors the current limitations in free-tier generative AI tools, where interaction volume is often restricted within set timeframes. Students are not permitted to modify the code manually unless they exhaust their query limit, at which point they may make limited manual adjustments to refine their code.

To enhance the experience further, students are required to select one LLM engine per challenge and rotate through all available engines over the course of Hackaprompt. This rule ensures that students gain exposure to different models and adapt to various AI capabilities.

A code of conduct was also established to maintain a collaborative and focused environment:

– *Cooperation with instructors*: Students are encouraged to seek guidance from instructors while adhering to the rules.
– *No cross-group spoilers*: To preserve the integrity of the challenge, students are asked to avoid sharing solutions or hints with other groups.
– *No out-of-context prompting*: Students are restricted from using any additional devices for prompting (e.g., laptops) to ensure that all interaction with the LLMs occurr within the Hackaprompt platform.

This structured approach to problems and rules is designed to foster not only technical skill development but also strategic prompting, collaboration, and adherence to challenge parameters.

3.2 Multimodal Data Collection Procedure

The data collection utilizes a *multimodal data collection procedure* we developed, which included:

– *Surveys*: Administered before and after the activity to assess problem-solving skills, self-awareness, and perceived learning outcomes;
– *Log analysis*: Used to track interactions with AI tools and analyze prompting strategies;
– *Video recording*: Employed to capture group activities and screen interactions via webcam.

A comprehensive overview of the content, purpose, and analysis methods for each data type is provided in Figure 2.

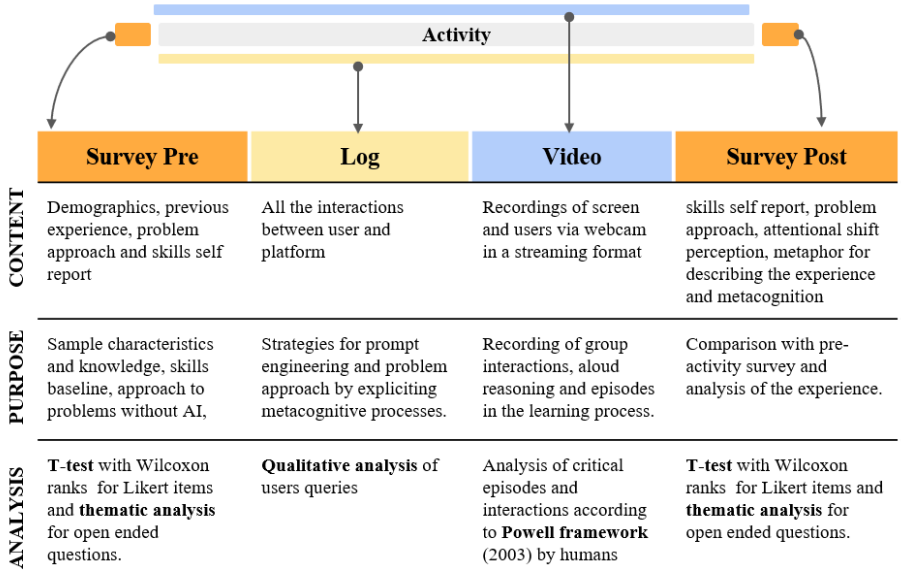

Fig. 2. Summary of the multimodal approach for data collection used in the study. The methods are detailed by content, purpose, and type of analysis performed. The activity chart also indicates the timing of each method's application throughout the Hackaprompt challenge.

Questionnaire's Content In our methodological proposal, the questionnaires serve multiple objectives: they collect demographic data, assess prior programming knowledge and problem-solving style, establish a baseline for qualitative and quantitative skill comparisons, and encourage reflection on the learning experience.

The pre-activity survey included the following items:

- *self-reported AI literacy*, measured on 5-point Likert scale;
- *current use of AI for programming*, measured on a 5-point Likert scale;
- *current approach to problem-solving without AI*, an open-ended question where respondents describe their solution process for a programming problem;
- *self-assessment of AI's impact on key learning skills*, such as critical thinking, creativity, team collaboration, and adaptability for complex problem-solving, measured on a 5-point Likert scale.

These items aim to define the sample's characteristics, knowledge on the topic, and self-rated confidence in key skills. Additionally, they establish a baseline for subsequent comparisons in AI literacy and confidence in critical skills (such as critical thinking, creativity, team collaboration, and adaptability in solv-

ing complex problems). Finally, they capture the students' approach to problem-solving without AI assistance.

The post-activity survey expands on the pre-activity survey items and includes additional questions:

- *Problem-solving approach using AI*: Respondents are presented with the same problem from the previous survey and asked to describe their solution process, this time using AI.
- *Perceived shift in attention and metacognitive reflection*: A closed-ended question using a 5-point Likert scale, with an open-ended prompt for further elaboration.
- *Discussion of problem-solving abilities*: Assessment of skills such as problem component identification, problem representation, solution planning, and solution evaluation, both with and without AI, rated on a 5-point Likert scale [28].
- *Analysis of metacognitive processes*: Evaluation of metacognition within the group context using Biasutti et al.'s group metacognition scale [4].

The second survey aims to assess changes in AI literacy and knowledge, as well as to compare how participants approach programming problems and exercises with and without AI. Additionally, the items are designed to prompt reflection on the learning experience and to gain insights into problem-solving and metacognitive processes, both at the individual level and within the group setting.

Logs and Video Recordings The platform systematically logs interactions between users and the large language models (LLMs) during problem-solving sessions. Each log entry includes:

- *Timestamps*: Each request and response is time-stamped, allowing a detailed reconstruction of interaction sequences and enabling precise timeline analysis.
- *Request and response texts*: Logs capture the full text of each prompt submitted by users as well as the corresponding response generated by the LLM.
- *Token count and model data*: The log records the number of tokens used in each request and response, along with the total tokens for each exchange. Information on the specific model (e.g., ChatGPT 3.5 Turbo or ChatGPT 4) is also logged.
- *Session metadata*: Each log entry includes metadata such as the team ID, task ID, and whether the interaction began a new conversation. This data allows tracking of group progress and specific task engagement.

These comprehensive logs enable full traceability of user interactions and provide a foundation for in-depth analysis of user behavior and model performance.

Video data is collected using an external webcam and the Zoom video conferencing platform (version 6.0.0). This setup captures both the participants and screen activity, enabling a detailed view of user interactions with the interface. Students join designated Zoom rooms, where screen sharing was enabled, ensuring all on-screen actions during problem-solving activities are recorded.

Data Analysis The multimodal data collection supports differentiated analysis for each type of data collected. The theoretical frameworks applied in data analysis are as follows:

- *Questionnaire data*, composed of:
  - Qualitative data, derived from open-ended questions, analyzed according to the thematic analysis framework [5]. Responses to a specific item on the problem-solving approach, where participants describe their steps, are further examined using an"expert review".
  - Quantitative items using a 5-point Likert scale, analyzed for statistical significance through a t-test with Wilcoxon ranks.
- *Log analysis*: Logs are analyzed with a qualitative approach in order to reconstruct the key passages of the solving and writing strategies.
- *Video data*: For the video analysis we refer to Powell's and colleagues' framework for identifying critical episodes and interactions [25]. This framework comes from the mathematics education literature and proposes a video analysis model consisting of seven non-linear steps (p. 413): Attention to the video data; Describe the video data; Identify critical events; Transcribe; Encoding; Storyline construction; Composing the narrative. For the pilot study, we limited ourselves to the first three phases, with the aim of identifying observables to enucleate the characteristics of critical events in this context and in this type of activity. Such characteristics will allow us to better specify not only the setting design of future experiments but also the workload management.

3.3 Rationale

Our multimodal approach is primarily guided by the principle of triangulation, aimed at gaining insights into various levels of the learning experience, including individual cognition, interaction with learning content, and group dynamics [15, 23]. This methodological interplay captures the learning process from three interconnected perspectives: individual cognitive changes, interactions with the AI, and peer-based reflections within the group. The combination of these methods seeks to provide a more representative and ecologically valid understanding of the learner's experience, both individually–by mapping cognitive and metacognitive processes in relation to the topic–and socially, through interactions and discussions within the group (Figure 3).

To achieve this goal, the methodology includes a pre-activity questionnaire to establish a baseline of participants' skills, prior knowledge, and initial problem-solving approaches without AI support. This is complemented by a post-activity questionnaire designed to measure changes and prompt participants to reflect on their problem-solving strategies, particularly any attentional shifts observed when using AI [17].

The activity's log collection provides a detailed view of the steps and reasoning students used to solve the problem [28]. Unlike commented code, the logs

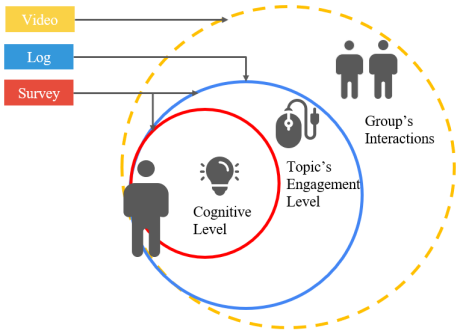

Fig. 3. Visual representation of the three levels of inquiry of the multimodal data collection approach.

outline each step in fine detail to facilitate AI processing and minimize misunderstandings [22], serving as a rich source of content.

Video recordings capture critical moments of both individual and group problem-solving activities, including contextual discussions and genuine reactions to task requirements. Despite the complexity of analyzing video, its added value lies in contextualizing the experience, providing a comprehensive and representative view of students' actions and their on-screen activities, enabling the examination of different levels and their interrelations. In applied disciplines, the learning process is deeply embedded in context; therefore, focusing solely on competencies overlooks significant aspects of what is actually happening.

A guiding principle of this approach is to strive for ecological validity by capturing the spontaneity of processes and interactions. Traditional methods, particularly in metacognitive research, often rely on self-reports and think-aloud protocols. While these can provide valuable insights, they may feel unnatural to participants, potentially compromising the authenticity of the processes being studied. In contrast, extended video recordings, the group setting, and the task of explaining everything in detail to the AI provide a robust framework to mitigate these effects, allowing participants to become immersed in the task and reducing their awareness of being recorded. This combination creates an opportunity to observe and collect rich data on evolving cognitive and metacognitive processes in a naturalistic learning environment.

4 Pilot Study

To validate this methodological proposal, we conducted a pilot study by collecting data using the procedure presented during the Hackaprompt challenge.

4.1 Participants

The sample consisted of 39 students recruited from the universities of Trento ($n = 20$) and Innsbruck ($n = 19$), representing the Bachelor's in Computer Science, Master's in Computer Science, and Master's in Software Engineering programs. Participation in the challenge and data collection was voluntary, with no academic credit or additional benefits provided. Students were selected based on a motivational letter and prior experience.

During the challenge, students were divided into groups of three, ensuring a balance of expertise and a mix of spoken languages. Each group included at least one student from each educational level, and no group spoke a single common language. The groups were intentionally multilingual to better simulate real-world collaborative environments and minimize the impact of language proficiency on group dynamics.

Participants were fully informed prior to the challenge and signed a privacy statement, consenting to data recording. Additionally, two groups agreed to video recording. All collected data were handled in compliance with GDPR regulations, exclusively for research purposes, ensuring confidentiality.

4.2 Preliminary Results from the Pilot Study

The pilot study's results revealed key insights into the impact of the educational intervention on participants' AI literacy and problem-solving processes. Participants reported having some prior knowledge and skills with AI in programming (primarily error correction and fixing typos). On a scale from 1 ("none") to 5 ("highly frequent user"), participants self-rated their AI programming experience, with an average score of 3.08 ($SD = 1.09$).

In more detail, participants demonstrated a significant improvement in AI literacy following the experience ($p < .05$), suggesting that the intervention effectively enhanced their understanding and proficiency in using AI tools.

The qualitative analysis of participants' responses to the problem approach question revealed a notable shift in focus–from an emphasis on implementation and code syntax to a deeper understanding geared toward better problem reformulation. This transition indicates that such an intervention can encourage participants to engage more critically with the given problems.

The use of AI significantly improved aspects of problem decomposition and solution planning ($p < .05$). However, results showed that other aspects of problem-solving, such as solution evaluation and implementation, were not significantly affected by AI. This suggests that AI tools are more beneficial in the early stages of problem-solving but become less impactful in later stages.

Metacognitive reflections, both individual and group-based, revealed a significant shift in participants' attention toward verbal and written reasoning, which facilitated self-reflection on their thought processes. Participants described the learning experience as instrumental in enhancing their communication skills, understanding of their strategies, and ability to transfer knowledge to new contexts.

5 Discussion

Impact on Student's Learning The proposed intervention demonstrated significant effectiveness in fostering learning through critical reflection and valuable discussions around individual strategies. By encouraging students to articulate their reasoning in detail through specific prompts, the intervention shifted their focus from simply writing code to more effectively planning the overall process.

This type of activity explicitly engages metacognitive processes, helping students build the habit of investing time in a deeper understanding of the problem before moving on to coding–an approach that can be transferred to individual practice. Emphasizing explicit articulation of each step supports students in becoming more aware of their strategies when solving problems and optimizing them, ultimately improving their academic performance.

The intervention was also successful in enhancing programming skills, particularly in the areas of code editing and critical evaluation. These skills are vital in industry environments, both for established practices and AI tool integration. In this regard, the activity significantly benefited AI literacy by increasing participants' awareness.

Finally, from a cognitive perspective, the data analysis shows that this activity effectively navigated Bloom's taxonomy [2] within a single setting, progressing from lower-order to higher-order thinking skills.

Multimodal Data Insights The presented methodology offers a comprehensive approach to investigating the individual learning process for programming in computer science education.

The primary strength lies in the richness of the data and the holistic view of the learning process achieved through triangulating surveys, logs, and video recordings. This combination allows for insights into interactions, reactions, verbal reasoning, and the explanation of reasoning processes and code choices during prompt design and problem exploration.

This triangulation proves valuable for simultaneously addressing different aspects of the learning process: individual internal learning, interaction with the system, and interaction within the group. The employed approach effectively provides insights into metacognitive and problem-solving processes within an ecological context by merging self-report measures with observational data. The contextualization of logs and exercises alongside self-report and observational measures significantly enhances validity and the quality of insights obtained.

Although human analysis is valuable, the data collected through logs also enable the use of advanced text analysis techniques.

Limitations The methodology and educational intervention proved to be powerful, yet resource-intensive. Regarding the educational intervention, the primary resource requirement involves the costs associated with using LLM API services. In our study, we had only 40 students and around 10 workstations, and thanks

to the limited number of queries, the costs remained manageable. However, scaling this activity to a curricular laboratory setting would significantly increase costs, as the number of interactions with the LLM grows. Technical constraints may also affect the feasibility of large-scale implementation and accessibility in broader contexts. Additionally, exercises and training materials must be carefully adapted to both the students' level and the capabilities of the LLMs to provide appropriately challenging tasks.

The limitations of the methodology pertain to the complexity of the collected data, resulting in a high workload for analysis. This requires not only substantial time but also advanced expertise to draw meaningful conclusions. While the methodology is certainly highly holistic and detailed, it comes at the cost of a significant workload.

Regarding the pilot study, future efforts will further investigate the impact of the language spoken and the role of training from different universities and backgrounds.

Further Directions Despite the acknowledged limitations, the promising results encourage further efforts to validate and refine new versions of the intervention, with the goal of integrating the activity into an elective laboratory for introductory and advanced programming courses.

With refinements in the methodology–such as improved control over the physical setting, including providing additional screens for some students to enhance collaboration and effectively capture their coding activities–we are confident in advancing this approach as an innovative and holistic methodology for computer science education and the investigation of learning processes. To further enhance scalability and cost-efficiency in future iterations, we plan to adopt a modular activity design adaptable to different class sizes and levels, while also incorporating low-cost technological tools to improve feasibility.

6 Conclusions

In the information era, computer science education faces challenges and opportunities but often lacks readiness due to concerns about the impact of new technologies on learning quality and insufficient research methodologies.

This study addresses these challenges with a structured intervention for programming education and a comprehensive research methodology, providing both a learning opportunity for students and valuable insights into the learning process. The multimodal data collection offered a holistic view of learning, revealing insights into individual and collaborative problem-solving, AI literacy, and metacognition in the context of AI interaction. While the proposed methodology offers a rich framework for understanding student-AI interactions, it also presents limitations, particularly regarding costs and scalability.

Overall, this work presents a model for effectively embedding AI in programming education while investigating underlying learning processes. The pilot study demonstrated AI's positive impact on specific aspects of problem-solving,

critical thinking, and metacognitive skills. Future efforts will focus on scaling the activity and refining the methodology to achieve a more sustainable integration of AI, enhancing both student learning and the collective knowledge of the educational community.

References

1. Almeida, C.: Guideline proposal for quality evaluation of research in software engineering education. In: 2021 16th Iberian Conference on Information Systems and Technologies (CISTI). pp. 1–4. IEEE, Chaves, Portugal (Jun 2021). https://doi.org/10.23919/CISTI52073.2021.9476586
2. Anderson, L.W., Krathwohl, D.R. (eds.): A Taxonomy for Learning, Teaching, and Assessing: A Revision of Bloom's Taxonomy of Educational Objectives. Longman, New York (2001)
3. Beheshti, A.: Natural language-oriented programming (NLOP): Towards democratizing software creation (Jun 2024), https://arxiv.org/abs/2406.05409, accepted in: 2024 IEEE International Conference on Software Services Engineering (SSE), Shenzhen, China, July 7-13, 2024
4. Biasutti, M., Frate, S.: Group metacognition in online collaborative learning: Validity and reliability of the group metacognition scale (GMS). Educational Technology Research and Development **66**(6), 1321–1338 (Dec 2018). https://doi.org/10.1007/s11423-018-9583-0
5. Braun, V., Clarke, V.: Thematic Analysis: A Practical Guide. SAGE Publications (2022)
6. Bucaioni, A., Ekedahl, H., Helander, V., Nguyen, P.T.: Programming with chatgpt: How far can we go? Machine Learning with Applications **15**, 100526 (2024). https://doi.org/https://doi.org/10.1016/j.mlwa.2024.100526
7. Clear, T., Cajander, A., Clear, A., McDermott, R., Bergqvist, A., Daniels, M., Divitini, M., Forshaw, M., Humble, N., Kasinidou, M., Kleanthous, S., Kultur, C., Parvini, G., Polash, M., Zhu, T.: A plan for a joint study into the impacts of AI on professional competencies of IT professionals and implications for computing students. In: Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 2. pp. 757–758. ACM, Milan, Italy (Jul 2024). https://doi.org/10.1145/3649405.3659527
8. Cornoldi, C.: Metacognizione e Apprendimento. Il Mulino, Bologna, Italy (2002)
9. Daun, M., Brings, J.: How ChatGPT will change software engineering education. In: Proceedings of the 2023 Conference on Innovation and Technology in Computer Science Education V. 1 (ITiCSE 2023). pp. 110–116. ACM, Turku, Finland (Jun 2023). https://doi.org/10.1145/3587102.3588815
10. Denny, P., Smith, D.H., Fowler, M., Prather, J., Becker, B.A., Leinonen, J.: Explaining code with a purpose: An integrated approach for developing code comprehension and prompting skills. In: Proceedings of the 2024 on Innovation and Technology in Computer Science Education V. 1. pp. 283–289. ACM, Milan, Italy (Jul 2024). https://doi.org/10.1145/3649217.3653587
11. Diachkova, V.: Development of metacognitive skills of college students in the process of learning mathematics, taking into account natural possibilities. Baltic Journal of Legal and Social Science (2023). https://doi.org/10.30525/2592-8813-2023-3-11

12. Finnie-Ansley, J., Denny, P., Becker, B.A., Luxton-Reilly, A., Prather, J.: The robots are coming: Exploring the implications of openai codex on introductory programming. In: Proceedings of the 24th Australasian Computing Education Conference. p. 10–19. ACE '22, Association for Computing Machinery, New York, NY, USA (2022). https://doi.org/10.1145/3511861.3511863
13. Hands, C.: A longitudinal examination of student approaches to learning and metacognition. Journal of Higher Education Theory and Practice **23**(19) (2023). https://doi.org/10.33423/jhetp.v23i19.6677
14. Hassani, H., Silva, E.S., Unger, S., TajMazinani, M., MacFeely, S.: Artificial intelligence (AI) or intelligence augmentation (IA): What is the future? AI **1**(2), 143–155 (Apr 2020). https://doi.org/10.3390/ai1020008
15. Heale, R., Forbes, D.: Understanding triangulation in research. Evidence Based Nursing **16**(4), 98–98 (2013). https://doi.org/10.1136/eb-2013-101494, https://doi.org/10.1136/eb-2013-101494
16. Igelnik, B. (ed.): Computational Modeling and Simulation of Intellect: Current State and Future Perspectives. IGI Global (2011). https://doi.org/10.4018/978-1-60960-551-3
17. Kittel, A.F.D., Seufert, T.: It's all metacognitive: The relationship between informal learning and self-regulated learning in the workplace. PLOS ONE **18**(5), e0286065 (May 2023). https://doi.org/10.1371/journal.pone.0286065
18. Li, Y., Choi, D., Chung, J., Kushman, N., Schrittwieser, J., Leblond, R., Eccles, T., Keeling, J., Gimeno, F., Dal Lago, A., Hubert, T., Choy, P., de Masson d'Autume, C., Babuschkin, I., Chen, X., Huang, P.S., Welbl, J., Gowal, S., Cherepanov, A., Molloy, J., Mankowitz, D.J., Sutherland, E., Kohli, P., de Freitas, N., Kavukcuoglu, K., Vinyals, O.: Competition-level code generation with AlphaCode. arXiv preprint arXiv:2203.07814 (Mar 2022), https://arxiv.org/abs/2203.07814
19. Liffiton, M., Sheese, B., Savelka, J., Denny, P.: CodeHelp: Using large language models with guardrails for scalable support in programming classes (Aug 2023), https://arxiv.org/abs/2308.06921
20. Lishinski, A., Good, J., Sands, P., Yadav, A.: Methodological rigor and theoretical foundations of CS education research. In: Proceedings of the 2016 ACM Conference on International Computing Education Research. pp. 161–169. ACM, Melbourne, VIC, Australia (Aug 2016). https://doi.org/10.1145/2960310.2960328
21. Ma, Q., Wu, T., Koedinger, K.: Is AI the better programming partner? human–human pair programming vs. human–AI pair programming. arXiv preprint arXiv:2306.05153 (Jul 2023), https://arxiv.org/abs/2306.05153
22. Ojeda-Ramirez, S., Rismanchian, S., Doroudi, S.: Learning about AI to learn about learning: Artificial intelligence as a tool for metacognitive reflection (Aug 2023). https://doi.org/10.35542/osf.io/64ekv
23. Oliver-Hoyo, M., Allen, D.: The use of triangulation methods in qualitative educational research. Journal of College Science Teaching **35**(4), 42–47 (2006)
24. Perry, J., Lundie, D., Golder, G.: Metacognition in schools: What does the literature suggest about the effectiveness of teaching metacognition in schools? Educational Review **71**(4), 483–500 (Jul 2019). https://doi.org/10.1080/00131911.2018.1441127
25. Powell, A.B., Francisco, J.M., Maher, C.A.: An analytical model for studying the development of learners' mathematical ideas and reasoning using videotape data. The Journal of Mathematical Behavior **22**(4), 405–435 (2003). https://doi.org/10.1016/j.jmathb.2003.09.002

26. Randolph, J.J., Julnes, G., Sutinen, E., Lehman, S.: A methodological review of computer science education research. Journal of Information Technology Education: Research **7**(1), 135–162 (2008)
27. Shoenfeld, A.: Uses of video in understanding and improving mathematical thinking and teaching. Journal of Mathematics Teacher Education **20**, 415–432 (2017)
28. Simon, D.J., Swerdlik, M.E.: The problem-solving component. In: Supervision in School Psychology, pp. 295–328. Routledge, New York, 2nd edn. (2022). https://doi.org/10.4324/9781003242222-12
29. Sulistiyani, L., Habiddin, H., Yahmin, Y.: HOTS & problem-based learning (PBL) with blended learning. J-PEK (Jurnal Pembelajaran Kimia) **7**(1), 1–8 (Jun 2022). https://doi.org/10.17977/um026v7i12022p001