Promoting Computational Thinking Skills in Non-Computer Science Students

Gamifying Computational Notebooks to Increase Student Engagement

Alessio De Santo, Juan Carlos Farah, Marc Lafuente Martínez, Arielle Moro, Kristoffer Bergram, Aditya Kumar Purohit, Pascal Felber, Denis Gillet, Adrian Holzer

***Abstract*—Computational thinking (CT) skills are becoming increasingly relevant for future professionals across all domains, beyond computer science (CS). As such, an increasing number of bachelor and masters programs outside of the computer science discipline integrate CT courses within their study program. At the same time, tools such as notebooks and interactive apps designed to support the teaching of programming concepts are becoming ever more popular. However, in non-CS majors, CT might not be perceived as essential, and students might lack the motivation to engage with such tools in order to acquire solid CT skills. This paper presents a field study conducted with 115 students during a full semester on a novel computational notebook environment. It evaluates computational notebooks and CT skills in an introductory course on information technology for first-year undergraduates in business and economics. A multidimensional evaluation approach makes use of pre- and post-test surveys, lectures, and self-directed lab sessions tracking analytics. Our findings suggest that, in the process of learning CT for non-CS students, engagement in active learning activities can be a stronger determinant of learning outcomes than initial knowledge. Furthermore, gamifying computational notebooks can serve as a strong driver of active learning engagement, even more so than initial motivational factors.**

***Index Terms*—fieldwork learning, active learning, computational thinking, computational notebooks, gamification, motivation**

I. Introduction

The use of computers, smartphones, and other connected devices is becoming part of the daily routine of an ever-increasing number of people around the world. The increased usage of computing devices and their processing power allows us to solve problems that we could not tackle before and, at the same time, this has complexified the way society works, leading to an increased presence of non-routine work [1]. Even people without computing skills need to use computers to carry out specific tasks in their daily lives. In this hyper-connected era, individuals must be aware of how to make the most of computers, which involves being fully capable of communicating with them and of extracting all their computing potential to solve complex problems in a wide range of domains [2]. As such, computational thinking (CT) is part of the essential skill set that a student should master in order to solve problems in the digital era [3]. This may include several key concepts such as abstraction, decomposition, pattern recognition, and algorithms [4]. However, the assessment of CT competence is not straightforward [5] due to the plethora of concepts involved, the fact that frameworks are different across authors, and the lack of validated tools. Furthermore, the recent COVID-19 health crisis has introduced additional complexity as many courses can no longer rely on in-class teaching support and have to be exclusively taught online. These different factors mean that educators should pay particular attention in engaging students in active learning to increase learning gains [6], and to promote specific pedagogies likely to increase their motivation, such as gamification [7], [8]. In majors outside of computer science, CT might not be perceived as essential, and students might lack the intrinsic motivation to engage fully in learning, which may prevent them from acquiring solid CT skills [9].

Computational notebooks are promising tools for teaching students how to solve complex problems using a programming language [10], [11]. These tools allow students to recreate and simulate exercises in an interactive manner, where they can manipulate chunks of code and observe the results of their actions in real time.

This study tackles this specific issue and brings new insights through a multidimensional evaluation approach of CT skills using multiple sources of data. Quantitative scores, insights of problem-solving strategies deployed by students, and usage data from the computational notebook used as course support have been analyzed. This study also includes a controlled experiment with a gamified feedback feature. In particular, it makes the research contributions outlined below.

A. Contributions

First, the paper introduces a novel computational notebook environment using the Graasp open digital education platform with associated learning scenarios [12]. The computational notebook application offers a rich learning environment with dynamic code execution, integrated learning analytics, and modular gamification modules.

Second, the paper presents a field study conducted using data captured during a full semester introductory course on information technology for first-year undergraduate students in business and economics. More specifically, we analyzed

Manuscript received XXX; Revised XXX, 2021.(*Corresponding author: Alessio De Santo*)

This research has been co-funded by Swissuniversities.

Alessio De Santo, Arielle Moro, Kristoffer Bergram, Aditya Kumar Purohit, Pascal Felber, and Adrian Holzer are with the University of Neuchâtel, Switzerland (e-mail: firstname.lastname@unine.ch). Juan Carlos Farah and Denis Gillet are with EPFL, Switzerland (e-mail: firstname.lastname@epfl.ch). Marc Lafuente Martínez is an independent consultant based in Switzerland (e-mail: marc.lafuentem@gmail.com).

Digital Object Identifier XX/TLT XXX

the data of 115 students who took the lecture course between February and June 2021 and who agreed to participate in this study.

Third, in the context of non-CS undergraduate students, this paper investigates whether computational notebooks can support active learning scenarios for promoting CT skills in non-CS students (RQ1), how engagement with computational notebooks is associated with student situational motivation (RQ2), and how gamification can contribute to increased engagement with computational notebooks (RQ3).

B. Roadmap

The remainder of the paper is structured as follows. Section II defines CT and discusses related work about computational notebooks and related motivational aspects and gamification mechanisms that may influence engagement in the context of CT knowledge acquisition. Section III presents *Graasp*, the education platform used for this study, as well as the learning scenarios. Section IV presents the research case study, the data used to carry out the analysis, and the techniques applied to address this research. Section V presents the results about each research question. Finally, Sections VI and VII conclude the paper and summarize the main insights.

II. Related work

CT and its use in educational settings stems from the work of Seymour Papert at the Massachusetts Institute of Technology in the second half of the 20th century [13]. Papert proposed that computers should be an integral tool of young people's learning, and put forward the use of programming languages such as Logo. The topic of CT has re-emerged as an increasingly relevant issue in education over the past few years. Jeanette Wing—considered to be the author who coined the term CT—asserts that it is a fundamental competence for everyone, not just for computer scientists [14]. Although there is a plethora of definitions and conceptualizations of the term, Jeanette Wing has conceived CT as the thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent. However, the promotion of CT in the classroom is challenging because—among other reasons—research on how to teach and learn CT in the classroom is scarce and does not provide clear measures as to which pedagogical methods are most effective [15]. If we want to improve these teaching practices at the university level, we must be able to distinguish effective methodologies and motivational affordances, such as gamification. The tools built up to now to evaluate CT in higher education are, to the best of our knowledge, still somewhat limited [16]. A literature review of computational notebooks, motivational aspects, gamification mechanics, and existing evaluation tools in a CT educational context is presented below.

A. Computational notebooks

Previous studies demonstrated that students increase their conceptual comprehension, critical thinking, and interpersonal skills when they participate actively in their study [17]. Such active participation is better known in the literature as active learning and is a teaching method that pushes students to continually assess their understanding by doing things. Active learning is an effective alternative to more passive types of knowledge acquisition, such as attending lectures [17]. One way to apply active learning in CT-related knowledge acquisition relies on blended learning. Blended learning combines traditional face-to-face learning with digital interaction in class or at home [18]. The shift to blended learning has been a key trend in education in the past decade. Currently, most learning activities are delivered using a blended approach to some degree [19], [20]. Blended learning also provides digital education platforms with the possibility to integrate learning analytics into the instructor's awareness and reflection processes, potentially allowing instructors—and other stakeholders (e.g., parents, researchers)—to assess how students are performing and to predict student success or failure early on in the course [21]. Furthermore, a blended learning approach can also potentially be used in a fully online learning context [22].

Blended learning is particularly applicable to introductory programming courses [23], which often incorporate rich learning environments with dynamic code integration, such as computational notebooks. Computational notebooks, which are widely used in data science education [24], combine code snippets and text with other multimedia content to create rich interactive environments for data exploration and programming [25]. Combining an online coding environment without the need for external software, and the ability to run code embedded in text and multimedia content, make computational notebooks a tool well suited to teaching CT [26]. Previous work has explored the use of computational notebooks to teach CT in different learning activities. For instance, researchers evaluated their use for (i) lectures, (ii) reading, (iii) homework, and (iv) exams [26].

The Jupyter Notebook [27] (henceforth *Jupyter*)—a popular computational notebook—has seen a particularly significant increase in popularity over the past few years, becoming a valuable teaching tool. One of the keys for Jupyter's rise in popularity is its support for the Python programming language, whose simplicity and readability make it attractive as an introductory programming language [28]. As such, Jupyter is becoming more popular in introductory Python courses [29], [30], despite the fact that there are many other web-based tools that have been suggested for teaching Python [31], [32]. This preference for Jupyter could be explained by the fact that it offers many features aimed at students, including the ability to work on coding assignments without having to switch between the assignment's instructions and the coding software [33]. Furthermore, Jupyter includes many tools that are specifically made for teaching, such as grading modules [33]. Certain personalized learning environments (PLEs) allowing the creation of rich interactive learning spaces, gamified learning experiences, and learning analytics, have also started to provide support for computational notebook integration [12].

Nevertheless, these notebooks can also have a negative effect on learning. Some argue that they encourage poor coding practices, given that it is not straightforward to break

down code into smaller, reusable modules, and that it is hard to write and run tests [27]. Furthermore, the fact that computational notebooks are used both for exploratory and explanatory purposes can also lead to complications, since it takes a lot of effort to transform a messy exploratory notebook into a clean one that can be shared with others [34]. Moreover, these environments lack support for greater interaction, collaboration, activity awareness, access control, and other features [25]. Therefore, it has been argued that while computational notebooks can be useful for introductory-level students, they are not suitable for more experienced learners [35]. To address this issue, notebooks can be customized according to learning preferences, programming experience, and learning context [26]. The above observations lead to the following research question:

(RQ1) *Can computational notebooks support active learning scenarios for promoting CT skills in non-CS students?*

B. Motivation

As active learning scenarios rely on voluntary student engagement, it raises the question of the underlying motivations that drive or hinder engagement. As non-CS undergraduates may not perceive CT as essential, which could potentially make it difficult for them to develop strong computational thinking skills [9], it is critical to understand the motivational aspects of students engaging in active learning scenarios. This observation leads to the following research question:

(RQ2) *How is the engagement with computational notebooks associated with student situational motivation?*

Over the past 60 years, self-determination theory (SDT) has emerged as a fundamental theory of human motivation [36]. SDT's basic premises propose that motivation operates on three levels: global, contextual, and situational [37], [38]. Motivation on a global scale reflects how an individual interacts with his or her surroundings in general [38]. A motivating tendency toward a certain setting, such as a job or education, is known as contextual motivation [37]. Situational motivation relates to the "here and now" of motivation, or the motivation felt when participating in a certain activity [37]. All three levels can be further refined and described by various constructs, among them the motivational factors proposed by SDT [39], [40]: intrinsic motivation, identified regulation, external regulation and amotivation, constituting a self-determination continuum from self-determined to non-self-determined motivation. Intrinsically motivated behaviors are those that are done for the purpose of doing them, or for the pleasure and satisfaction that comes from doing them [39]. In contrast, extrinsic motivation refers to a wide range of behaviors in which the goals of action are not limited to those that are inherent in the activity. [39]. Different types of extrinsic motivations have been proposed by SDT; these are external and identified regulations [39], [40]. External regulation happens when behavior is regulated by rewards or to avoid negative consequences. Identified regulation, on the other hand, happens when a behavior is valued and viewed as one's own choice. However, the motivation still remains extrinsic because the activity is done as a means to an aim rather than for its own sake. Amotivation defines a completely non-autonomous behavior, with no drive to speak of and likely struggling to have any of one's self needs met. To measure a person's situational motivation, the Situational Motivation Scale (SIMS) can be used, as it demonstrates good reliability and factorial validity in educational context [41].

C. Gamification

For the use of gamified settings to promote CT, Kotini and Tzelepi [42] find that the use of gamification—e.g., using grading characteristics comparable with those of video games, such as points or levels—can increase the engagement of students. There are many types of settings one can apply, and instructional design has to be careful not to only promote external goals, such as points and prizes related to performance, because this would only lead to increasing the extrinsic motivation of students. The educational setting also has to integrate aspects that can grow students' interest in mastering their learning, thus leading to promoting intrinsic motivation as well. One key element is whether gamification can provide feedback and scaffolding for students and, if so, by which means. Providing feedback for learning activities has long been identified as an important component allowing students to identify gaps and to assess their learning progress [43]. Some experiments [44] have shown that gamified environments where the digital environment itself produces the scaffolds necessary so that students' acquisition of CT skills can be implemented. In another study, where a mobile app game was used to promote CT [45], the authors found that, generally, the average time that students spent on a level in the game increased with the level of progression. Other studies [46] found that it is the didactic sequence itself that scaffolds the students to acquire CT, and the authors report an increasing learning rate in the experimental group compared to the control group.

However, the literature does not clarify what role gamification can play in affecting learning outcomes and student engagement in the context of higher education, specifically in the case of non-CS students aiming to acquire CT skills. Given the different kinds of tools that appear in the literature, it seems wise to use a combination of tools that can provide greater reliability to evaluate students' CT skills and cover the different facets of their competence. This is precisely the perspective that will be adopted in this paper, where we will use multiple instruments to assess a student's CT expertise based both on programming and non-programming activities. The above observations lead to the following research question:

(RQ3) *How can gamification contribute to increased engagement with a computational notebook?*

D. Tools for evaluating CT skills

Compentency-based tests propose abstract items for assessing CT skills. For example, Gouws et al. created a test to evaluate CT performance in higher education students [47]. Sometimes, tests created for other purposes have been used as a tool to measure CT skills (e.g., including tasks related to conservation or probabilistic reasoning). That is the case

This article has been accepted for publication in IEEE Transactions on Learning Technologies. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TLT.2022.3180588

for the GALT test [48], which was used, for instance, in the context of higher education [49]. Recently, Lafuente et al. [50] developed a psychometric test to evaluate algorithmic thinking skills. The authors validated the test based on factor analyses and opinions of experts in the field, obtaining a 20-item test capable of discriminating experts in CT from students without any training in computational issues.

Self-assessment tools have been developed so that students can evaluate by themselves to what extent they have mastered different skills related to CT [51], [52]. These tests have been validated by researchers and used by students in higher education. However, self-reported questionnaires may yield measurement errors based on an overestimation of the student's own skills or lack of understanding of the concepts involved in the questionnaire [53]. This type of tool also includes interviews, which are used to extract qualitative evidence, mainly of the thought processes used by students to solve CT tasks [54].

Exams and other *ad hoc* tools are probably the most frequently used tools to evaluate CT [55]. The authors usually construct an artifact with tasks that resemble very much the ones used in the classroom for teaching and learning the subject (i.e., the evaluation tool is an exam), and very often the tools include the use of programming in a language that students have been learning in the class. These tools are mainly oriented to evaluating a student's CT-related knowledge. Likewise, portfolios and reports constructed by students are also used to evaluate CT competence, using evidence of understanding and achievement in CT-related activities [56]. Furthermore, the ability to properly assess a student's acquisition of CT skills could also provide valuable insight into how CT should be taught in the classroom, which is an active area of research [57].

This paper will make use of this body of research to design, implement, and evaluate adequate support for promoting CT skills.

III. Graasp Digital Notebook

The digital notebook environment studied in the paper is built using the Graasp personalized learning environment. Graasp is an open digital education platform providing two interfaces [12]. An *authoring view* allows instructors to combine and configure resources that they use to create their online lessons, which we refer to as *learning capsules*. Learning capsules can then be broken down into step-by-step exercises, which can be contextualized with text, images, links, chatrooms, and other interactive content (Figure 1). The second interface is the *live view*, a student-oriented environment that can be accessed through a link. By clicking on the link, students can take part in the online lesson, navigating through pages that contain lectures and exercise materials prepared by the instructor.

To provide a context resembling computational notebooks, we designed an open source coding application (henceforth the *code app*[1]) to provide a ready-made Python environment within Graasp. The code app uses the Pyodide[2] library to execute Python directly on the browser without any additional dependencies. Students can read and write files, provide manual input, and generate graphics using libraries such as Matplotlib [58]. The code app also includes a command-line interface to display output and to allow students to navigate a virtual file system, as well as a feedback functionality that allows instructors to review and annotate the code written by students. To enable advanced features such as custom configuration, saving student-generated code, and tracking learning analytics, the code app can leverage application programming interfaces (APIs) exposed by digital education platforms. In our case, we use Graasp's API to preconfigure the code app with sample code, data files, and instructions for students. Within the live view, students could then write, execute and save code, review feedback provided by the instructor, and visualize any graphical output.

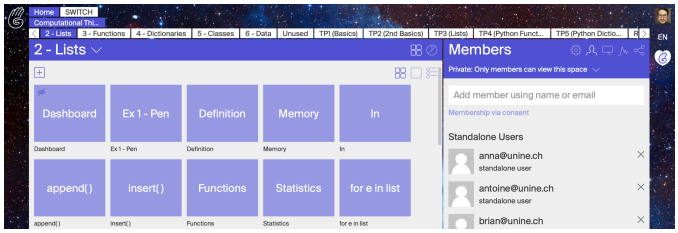

Fig. 1. Graasp authoring view.

The *code app* was then coupled with two others Graasp apps to gamify the active learning experience. The first additional app is a simple *answer app*,[3] which allows students to enter an answer and get feedback if it is correct or not. The second app is a *point counter app*.[4] This point counter is a gamification app added to the learning space which reads the output of the answer app, i.e., it adds points to the score if an answer is correct, and removes points if a hint is displayed.

A. Learning scenario 1: active lectures

This learning scenario supports knowledge transmission by an instructor in a live session, whether remote or in-class. It aims to make traditional lectures more interactive by providing dynamic slides to students who can write and execute their own code during the lecture. The goal is that in a first step, students follow the code that the instructor presents. Then, in a second step, students are encouraged to deviate from the code presented, in order to test some corner cases or validate some expected behaviors. Using the computational notebook, this scenario allows instructors to structure the course content into blocks or slides, each with an independent space to write and execute code and possibly images, videos, or other interactive content. In this real-time learning scenario, it is expected that students move along the slides at the same pace as the instructor. Figure 2 shows a typical example of a learning capsule with different slides (e.g., definition, memory). In the selected slide (*Definition*) there is a block of static text with the interactive code app below. Concretely, the learning activity in

[1]Code App: github.com/graasp/graasp-app-code
[2]Pyodide: github.com/iodide-project/pyodide
[3]Answer App: https://github.com/graasp/graasp-app-submit-answer
[4]Point counter app: https://github.com/alessio265/graasp-app-levelvis

This article has been accepted for publication in IEEE Transactions on Learning Technologies. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TLT.2022.3180588

Figure 2 depicts one of the Python lessons and showcases one of the hands-on exercises performed during the theoretical part of the course. In this example, students are presented with a list they should print and index, providing an introduction to the concept of list in Python.

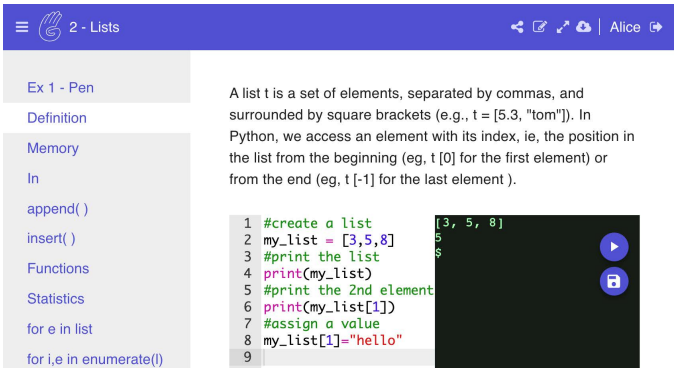

Fig. 2. Interactive lecture: a computational notebook learning capsule on Graasp. The instructor and students can write and execute code during the course.

B. Learning scenario 2 — self-guided labs

This learning scenario aims to support self-guided knowledge acquisition during lab sessions. The idea is to present students with exercises and to include auto-correction and formative feedback. Several tools can be included within the learning capsule to provide formative feedback. A simple input app allows students to submit text, while a real-time communication app enables students to spontaneously ask questions and to respond to multiple-choice questions posed by the instructor. Students could also use the app to complete homework assignments and provide answers to the problems presented during the lab sessions. Figure 3 shows three apps in the learning capsule to support lab sessions. The first is the *code app*, which allows students to run code. It should be noted that it can make use of hidden lines of code that can be executed before or after the visible code. Second, there is the *answer app*, which allows students to enter an answer and get feedback if it is correct or not correct. This app also allows teachers to set a hint for each question. Such a hint can then be displayed by students if they wish. Third, there is the *point counter app* on the right-hand side of the live view. This point counter app reads the output of the answer app, illustrating accumulation of points for each correct answer given, but also the loss of points when asking for a hint. The goal was to increase the time spent by students on activities by decreasing their need for help, i.e., the number of hints asked for.

IV. Methodology

In this section, we present the research case study, the data we used to carry out this analysis, and the techniques we applied to address our research questions. The case study for this paper is a full semester introductory course on information technology for first-year undergraduate students in business and economics (February–June 2021). This course consisted of two 45-minute periods per week of theoretical lectures and two

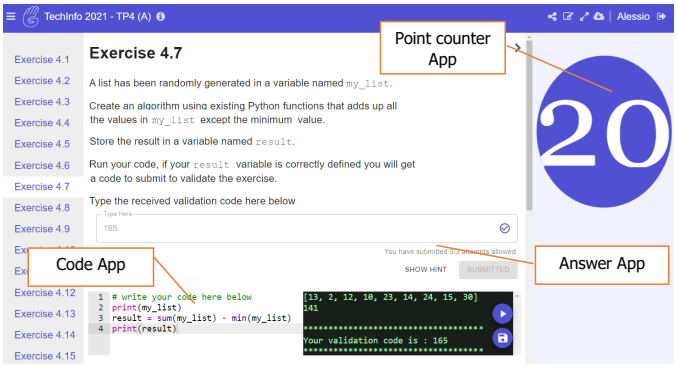

Fig. 3. Lab support with Graasp. A self-guided learning activity with visual point feedback.

TABLE I
Course Outline

Week	Lecture	Lab session
1	Pre-test survey	CT concepts
2	CT concepts	
3		
4		
5	Spreadsheet formulas and computational models	
6		
7		
8		
9	Python – variables and conditions	
10	Python – loops	
11	Python – lists	
12	Python – functions	
13	Web technologies	
14	Questions & Answers	Post-test survey

45-minute periods per week of lab sessions (see Table I). This course covers an introduction to CT concepts (2 weeks), introduction to spreadsheet formulas and computational models (5 weeks), Python programming (4 weeks), web technologies (2 weeks), and a final week with an exam dry run. The course is evaluated through a one-hour online exam. During the first and the last week, respectively, students filled in a pre- and a post-test survey, which inquired about their CT skills and attitudes. Out of the 115 students in the course, 112 gave their consent for this study.

A. Learning outcome data

There were no prerequisites for this course, and the learning outcomes of the course were for students (1) to be able to conceptualize problems computationally, i.e., use CT principles to describe and attempt to solve problems, and (2) to be able to solve simple problems algorithmically using the Python programming language. These learning outcomes were measured informally at the beginning (pre-test) and at the end of the course (post-test), and formally during the written exam at the end of the semester.

The pre- and post-tests were each composed of six problem-solving questions and six Python programming questions (examples are given in Figures 4 and 5). The problem-solving questions were extracted from the Algorithmic Thinking Test for Adults [50]. For all questions, there was one correct answer. For all problem-solving questions, besides asking

This article has been accepted for publication in IEEE Transactions on Learning Technologies. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TLT.2022.3180588

students for an answer, we asked them to provide a textual description of their problem-solving strategy for tackling the problem. This second part was not taken into account for the scoring of their answer, but allowed us to get an impression of how many CT concepts and higher levels of thinking were used in the process of solving or attempting to solve the questions.

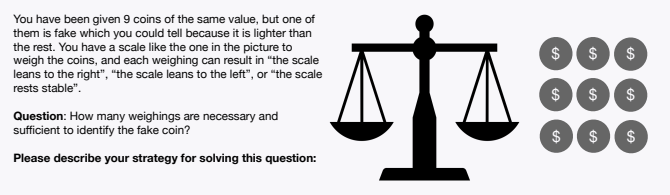

Fig. 4. Example of a general problem-solving question. Note that the question has two components. The first is quantitative and requires a precise answer, the second is qualitative and requires an open-ended answer describing the problem-solving strategy.

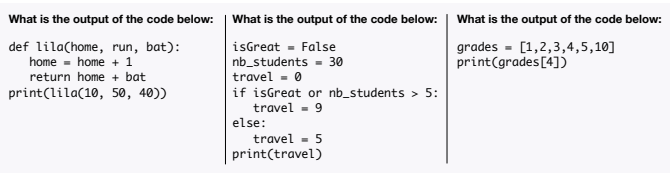

Fig. 5. Example of three basic programming questions. These questions each require a precise answer.

More specifically we analyzed the six problem-solving questions of the pre- and post-tests, where students had to explain their reasoning process. We sought to determine whether the key terms and concepts presented during the course had been assimilated and reused in the explanations given by the students using an approach inspired by grounded theory [59], [60] and open coding techniques, making the categories emerge from theoretical content of the courses, resulting in 22 different concepts:

decomposition, sub-problem, rule, specification, repetition, generalization, variables, function, instruction, abstraction, model, class, algorithm, loop, repeat, sequence, condition, trial, error, iteration, increment, test.

For each word of the student explanations and for each target concept presented here above, we performed lemmatization to transform words with roughly the same semantics to one standard form. Lemmatization was performed through WordNet corpus of the Python Natural Language Toolkit (NLTK). WordNet is a large, freely and publicly available lexical database for the English language, establishing structured semantic relationships between words. [5]

The final exam consisted of five open-ended Python questions, asking for simple functions or programs, such as: "Write a function that takes two parameters as input (a string called word and an integer called n) and returns a new string made of n times the word".

[5]https://www.nltk.org/howto/wordnet.html

B. Lecture data

During the lectures, we used learning analytics in Graasp to track student attendance and visual analysis to evaluate if the student followed the lecture. As an example, Figure 6 shows a learning dashboard to track user activity. More specifically, it shows the order in which each student has visited the pages available in the live view, as well as the time spent on each of them. If the instructor uses the live view at the same time, then the instructor's data can be compared against the student's data. Each color represents a page inside the live view. If students were to be perfectly synchronized with the instructor, their color patterns would all be the same. The activity of each student could then be visually evaluated assigning a score of 0 if the student was absent, of 1 if participation was passive and 2 if active (pefectly synchronized).

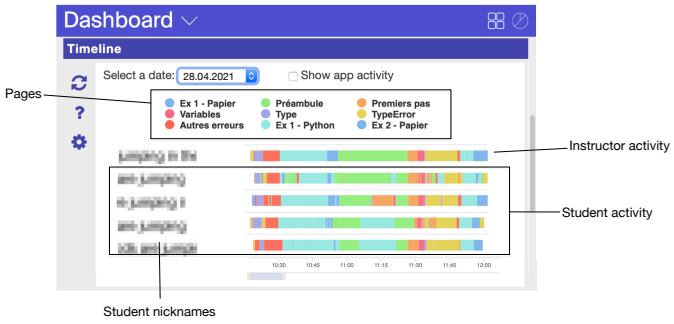

Fig. 6. Activity dashboard.

C. Lab session data

During the Python programming lab sessions, students were randomly split into treatment (70 students) and control group (45 students). The students went through four series of 15 exercises, a total of 60 exercises. The various series of exercises corresponding to the different topics introduced each week in the theoretical courses were: 1) variables and conditions, 2) loops, 3) lists, and 4) functions. The treatment group was provided with an extensive gamified feedback, including a level visualization (point counter App), as shown in Figure 3, while the control group had limited feedback, only knowing if their answers were right or wrong. The gamified feedback appears on the right-hand side of the interface in the form of a chain of bubbles that scrolls along with the score. To help them in the resolution of the exercise, students could ask for hints. For each exercise, the code block of the computational notebook was preconfigured to perform a list of tests on the execution of students' code, providing validation keys. Once the student's algorithm could execute properly, a validation key was returned back. For each exercise, two different validation keys could be received back by the students. In the first case, the algorithm acts as expected, while in the second one the algorithm does not act as expected. There were no limits on the number of tests and executions of the algorithms, and students were not aware of the meaning of the validation key received. Validation keys were randomly predefined and therefore different for each exercise. Theses validation keys

should then be submitted by the students in the answer app. When submitting their validation key, a checkmark or a cross allows feedback to be provided to the students on the correctness of their algorithm. Furthermore, for each correct answer on the first attempt, students get three points. For each correct answer provided after the first attempt, students get two points. For every hint revealed, students lose one point. The control group has no visual feedback of its score. Figure 3 illustrates one exercise of the self-guided lab session regarding Python functions. The hidden hint for that specific exercise being "You should be able to write this algorithm in one line".

D. Psychometric and demographic data

In addition to the above data, we also collected demographic data, student situational motivation, and the computational notebook usability level.

Student motivation was assessed in the post-test survey which aimed to measure their motivation to perform the lab sessions through the computational notebook. Situational motivation was assessed using the 16-item *Situational Motivation Scale* (SIMS). SIMS is designed to assess intrinsic motivation, identified regulation, external regulation, and amotivation [41].

In order to evaluate the usability level of the computational notebook, the students answered ten questions about the computational notebook, based on the system usability scale (SUS) [61] at the end of the post-test.

E. Path model and analysis

To provide a global view of the different factors influencing the learning outcomes, we designed a path model and conducted a partial least squares (PLS) analysis technique using SmartPLS. PLS is a variance-based structural equation modelling (SEM) analysis technique increasingly popular for analyzing explanation and prediction of information systems phenomena [62]. Central to PLS is the path model that can be visualized by a diagram that displays the hypotheses and variable relationships to be estimated in an SEM analysis [63]. T-statistics are used to test the proposed hypotheses for the standardized path coefficients, by specifying the same number of cases that existed in the dataset and bootstrapping 1000 resamples. The resulting design of the path model for this analysis is depicted in Figure 7. It contains three main independent variables: (1) initial skills, as measured by the score on the pre-test, (2) situational motivation, as measured by the SIMS scale, and (3) gamified feedback, which indicates whether the student was in the gamified feedback condition or not. Note that situational motivation can be further broken down into its four components (intrinsic motivation, identified regulation, external regulation, and amotivation). These variables potentially influence lab performance positively [64], as measured with the score on the lab exercises and the engagement on the online platform. Engagement is measured by tracking student interactions (i.e., number of clicks, number of code executions, text written) on the Graasp platform. "Need for help" construct is measured by the number of hints requested by a student (the more hints, the greater the need for help). Gamified feedback can motivate people to perform tasks that will increase virtual rewards (e.g. points) [65]. ' As such, we hypothesize that it will increase lab performance and reduce the need for help. In other words, this would mean that gamification of the activity, as well as increased motivation would lead students to try to get the answers on their own to get more points, without asking for hints. Finally, lab performance and initial skills potentially positively influence the learning outcome [66], as measured by the grade of the exam.

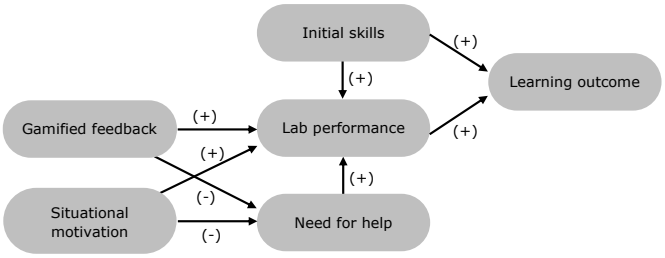

Fig. 7. Path model. Positive influence (+) is expected among the linked constructs of the model.

To validate the reflective constructs of our path model (i.e., lab performance, intrinsic motivation, identified regulation, external regulation, and amotivation), we evaluated their reliability, convergence, and discriminant validity.

1) Reliability: we used composite reliability (CR) and average variance extracted (AVE) as indicators. As shown in Table II, the CR of the constructs was greater than 0.7 and the AVE greater than 0.5, thus these constructs are reliable [62].

2) Convergent validity: we used the outer loadings and the AVE as convergent validity indicators [67]. The outer loadings of all our reflective variables were above 0.7, which is the standard threshold [67], except for one variable in the amotivation construct which was only above 0.5. As this study should be deemed as exploratory research—indicators between 0.4 and 0.7 were kept, as recommended by Hair et al. [67].

TABLE II
EVALUATION OF REFLECTIVE CONSTRUCTS

	Cronbach's Alpha	***rho_A***	***CR***	***AVE***
Lab session perf.	0.727	0.761	0.878	0.783
Intrinsic motivation	0.887	0.921	0.920	0.742
Identified regulation	0.838	0.861	0.891	0.673
External regulation	0.762	0.715	0.820	0.553
Amotivation	0.784	0.921	0.856	0.606

3) Discriminant validity: We used the heterotrait–monotrait (HTMT) ratio as a measure of discriminant validity [62]. Values lower than 0.85 are considered as acceptable for conceptually distinct constructs [62]. As shown in Table III, with the exception of Amotivation → Identified Regulation which scores 0.857 and thus is on the margin, all values were lower than 0.85, demonstrating the discriminant validity of our constructs.

This article has been accepted for publication in IEEE Transactions on Learning Technologies. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TLT.2022.3180588

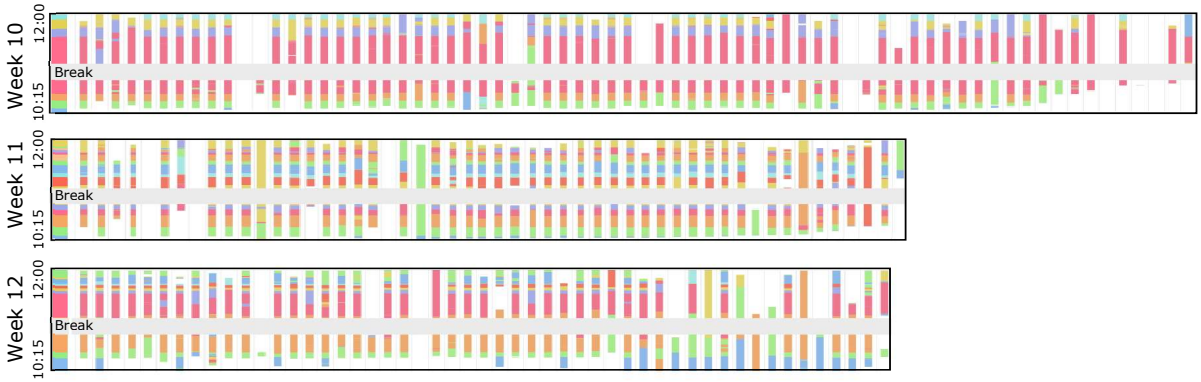

Fig. 8. Visualization of students following the lectures in Weeks 10–12.

TABLE III
HETEROTRAIT-MONOTRAIT RATIO (HTMT)

	Lab perf.	*Int. motiv.*	*Iden. reg.*	*Ext. reg.*	*Amotiv.*
Lab session perf.					
Intrinsic motivation	0.569				
Identified regulation	0.685	0.748			
External regulation	0.335	0.183	0.426		
Amotivation	0.544	0.688	0.857	0.413	

V. RESULTS

A. Can computational notebooks support active learning scenarios for promoting CT skills in non-CS students? (RQ1)

To answer the first research question, we evaluate if the notebook was considered usable, if it was used as intended in the learning scenarios, and whether there were learning gains.

1) Usability: The average SUS score is equal to 67.4 ($N = 63$), which represents okay usability [68]. There is no significant difference $t(61) = 0.74, p = 0.5$ in terms of usability between males ($M = 65.8, SD = 19.7$) and females ($M = 69.4, SD = 18.4$).

2) Learning scenario: Using data from the learning dashboard presented in Figure 6, we examined usage patterns from Week 10 to Week 12 as shown in Figure 8.

The dashboard gives a visual impression of how synchronized students are during the lecture. Note that the first slide (blue on the bottom) is always a pen and paper exercise, which explains why students are not always looking at the slide on the computational notebook. A visual analysis shows that during the lecture on Week 10, 62 followed at least part of the lecture on the computational notebook and 40 of them (64.5%) followed actively (meaning that around 80% of the lecture material was followed in the same order as the instructor, switching slides at around the same time), the other 22 students are considered as following the course passively. In Week 11, there were a total of 49 students online, among them 39 were active (79.5%). In Week 12, there were a total of 50 online, of whom 39 were active again (78%), and 33 were the same as the previous lecture.

The overall engagement of students during the live online lecture is depicted in Figure 9. It shows how many students were mostly active, mostly passive, or absent during these three lectures ($N = 112$). Of the 112 students who at some point appeared on the course, 42 did not participate in the online lectures, 28 were passive, and 42 were active. Among the 96 who ended up taking the exam, 31 did not follow the lectures, 23 were passive, and 41 were active.

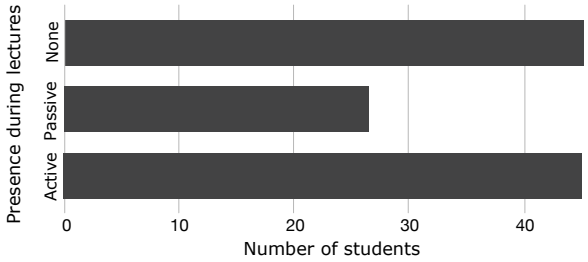

Fig. 9. Bar chart of lecture presence during Weeks 10–12 ($N = 112$).

Finally, we also analyzed whether students watched the recordings of the course that were put online after the lecture. Figure 10 shows student engagement with the lecture (live on the computational notebook and after the lecture viewing videos) over three weeks. Real-time activity is reported as a percentage of active participation in the real-time lecture discussed above. The video watching activity is reported as a percentage of the total time of the videos posted for the three weeks capped at a 100%.[6] The total number of videos posted was 149 minutes for the three lectures. The results show that a significant number of students (around 20%) did not participate actively in the live lessons, but nevertheless watched the videos at home. One student spent more than 900 minutes watching videos.

3) Learning gains: The main goal of this analysis is to explore the evolution of the CT skills of the students and to see if we can observe differences between their initial knowledge and their learning outcomes (Python score, CT score). To answer this question, we compared student scores on the pre- and post-tests as well as the evolution of their problem-solving strategies (CT concepts). Figure 11 provides a visual overview of the results of the mean scores in percentage points with

[6]Students could watch videos several times, which could lead to some playing times exceeding 100%.

This article has been accepted for publication in IEEE Transactions on Learning Technologies. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TLT.2022.3180588

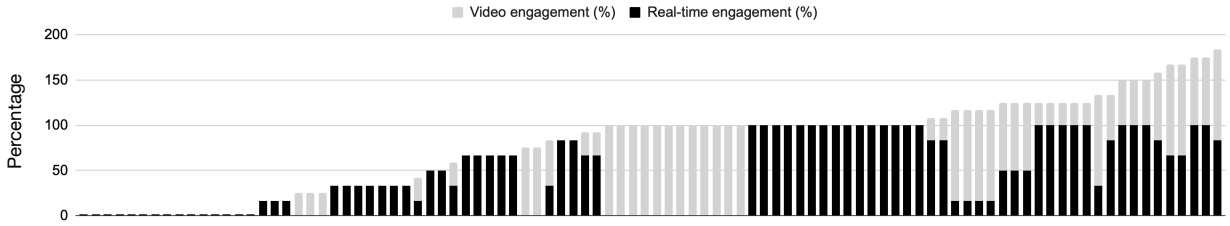

Fig. 10. Student engagement with the lecture in number of minutes of attention, live on the computational notebook, or later by watching videos ($N = 112$).

pre-test results as baseline. To perform the analysis, mean scores were normalized and ranged between 0 and 1, which represents the maximum achievable score.

When it comes to the Python score, a statistically significant difference was found for the Python exercises ($t(59) = 11.25, p < 0.01$) between the pre- ($N = 60, M = 0.16, SD = 0.24$) and post-test ($N = 60, M = 0.61, SD = 0.29$) scores.

Regarding CT score, a paired t-test revealed that there is a statistically significant difference ($t(59) = 3.73, p < 0.01$) in problem-solving exercises between the pre- ($N = 60, M = 0.39, SD = 0.25$) and post-test ($N = 60, M = 0.52, SD = 0.27$) scores.

Finally, in terms of CT concepts, we analyzed student answers to the CT questions from a semantic and linguistic point of view. To observe the potential evolution of the use of such conceptual terms in the problem-solving explanations given by the students, we compared the appearance frequency of each concept in students' pre- and post-test explanations. The results show a statistically significant ($t(59) = 2.31, p < 0.05$) positive evolution between the pre- ($N = 60, M = 0.43, SD = 0.79$) and post-test ($N = 60, M = 0.97, SD = 1.65$) scores.

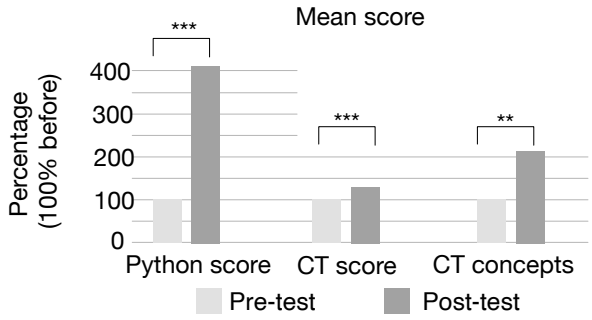

Fig. 11. Pre-test and post-test results (pre-test used as baseline). *** $p < 0.01$, ** $p < 0.05$.

4) Learning outcome: Figure 12 gives an overview of the final grades of the course ($N = 96$) between 1 (worst) and 6 (best) with the passing grade being 4. The pass rate for this course was 60.4%. There is also no significant difference t(94) = 0.26, p = 0.8 in grades between males (M = 3.9, SD = 1.3) and females (M=3.9, SD= 1.4).

Looking at engagement with the lecture material in real-time on the computational notebook a median-split of the student grade results (pass/fail) ordered according to the time that they spent following the lecture created two natural groups: one with high engagement and one with low engagement. A χ^2 test of independence indicated a significant association between lecture engagement and having a passing final grade in the course $\chi^2\ (1, n = 96) = 6.27, p = 0.012$. In fact, the group with high lecture engagement was about 50% more likely to pass the course (35 students had a passing grade (76%) in the high engagement group compared to 23 (48%) in the low engagement group. To assess whether engagement with videos was also associated with a higher pass rate, we performed a median-split of student grade results (pass / fail) ordered according to the time spent watching the videos. However, no significant difference was found.

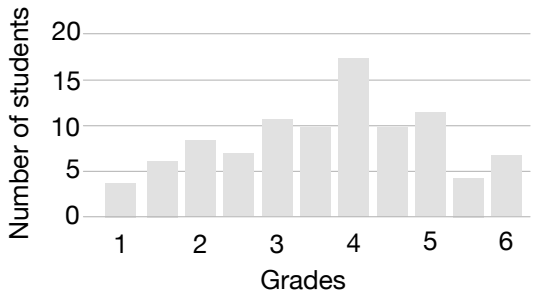

Fig. 12. Distribution of raw grades. $M = 3.88, SD = 1.30, N = 96$.

B. How is the engagement with computational notebooks associated with students situational motivation? (RQ2)

To answer this second research question we evaluate if the engagement with computational notebooks is associated with students' intrinsic motivation, identified regulation, external regulation, and amotivation. The analysis was performed on the subsample of 84 students who filled the pre-test survey, numbering 46 males and 38 females. As measured by the SIMS scale, the mean of student motivational aspects were calculated. Figure 13 presents student motivational aspects in participating to lab sessions during week 10 to 12. The highest motivational aspect was identified regulation ($M = 3.93, SD = 0.73$), followed by the external regulation ($M = 3.54, SD = 0.76$) and then the intrinsic motivation ($M = 3.37, SD = 0.82$), with only little overall amotivation ($M = 2.24, SD = 0.73$). To evaluate the outcome of this research question, we relied on our path model analysis (Figure 14). The path model analysis allows us to evaluate how the constructs of intrinsic motivation, identified regulation, external regulation, and amotivation [41] influence the students' behavior on computational notebook activities.

This article has been accepted for publication in IEEE Transactions on Learning Technologies. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TLT.2022.3180588

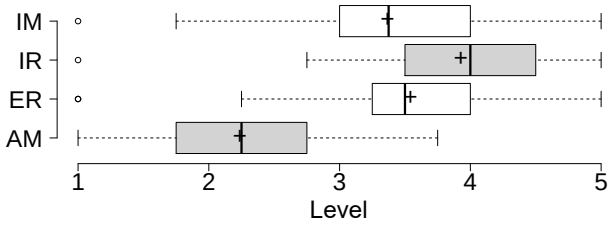

Fig. 13. Student mean intrinsic motivation (IM), identified regulation (IR), external regulation (ER), and amotivation (AM), as measured by the SIMS scale, to use computational notebooks in the context of Weeks 10 to 12 lab sessions.

More precisely, we investigate how the students' behavior is influenced by motivational aspects during the lab sessions. We were particularly interested in the influence on student lab performance and on students' need for help. As illustrated in Figure 14, coefficients of our path analysis indicate that intrinsic motivation had a significant effect $(p < 0.05)$ on lab session performance (0.272). Intrinsic motivation did not have any significant influence on the need for help (i.e., hints requested). Identified regulation influenced (0.323) significantly $(p < 0.05)$ students' lab session performance, but not the amount of hints requested. While external regulation influenced (0.347) significantly $(p < 0.01)$ the students' quantity of hints requested but not the final lab session performance. Amotivation does not have any influence on students' lab session performance or on the number of hints requested.

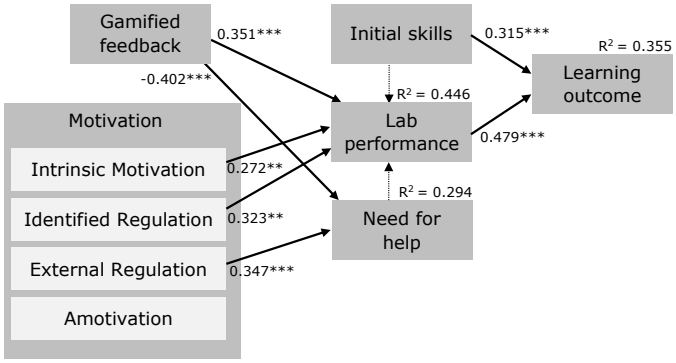

Fig. 14. PLS Model results. *** $p < 0.01$, ** $p < 0.05$

C. How can gamification contribute to increased engagement with a computational notebook? (RQ3)

To answer this research question we evaluate if the gamification of the lab sessions contributed to increasing the lab session performance and allowed a decrease of undesired behavior, in this case need for help. The analysis was performed on the same subsample and the same path model as for RQ2 (Figure 14). The gamified feedback functionality was implemented as shown in Figure 3, rewarding students for accurate answers and penalizing them for hints revealed. A control group ($N = 29$) was defined with no visual gamified feedback. The aim was two-fold: (1) to increase lab session engagement through the scoring system and engagement, and (2) to decrease the need for help (hints requests).

1) Increasing lab session engagement: Figure 14 shows that the link between gamified feedback and lab session performance is significant $(p < 0.01)$ and positive (0.351). Furthermore, gamified feedback has been found to be the most influencing construct on the lab performance, more influential than identified regulation or intrinsic motivation. Overall lab performance was predicted at 44.6%. As depicted in Figures 15 and 16, students receiving gamified feedback engaged more ($M = 14473.24, SD = 13862.91$) and performed better ($M = 87.6, SD = 61.06$) in the lab sessions than students from the control group, with respectively ($M = 13128.83, SD = 13329.82$) and ($M = 63.24, SD = 51.74$).

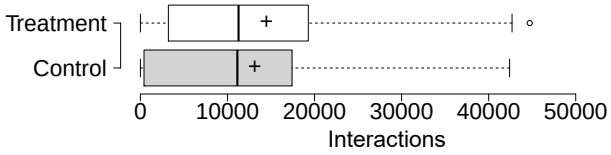

Fig. 15. Lab session engagement, resulting from activity tracking on Graasp, for students with gamified feedback (treatment) versus students without gamified feedback (control).

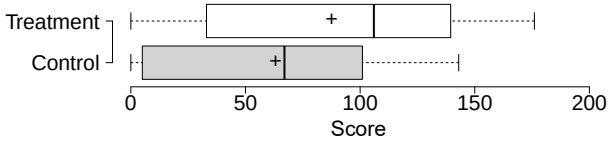

Fig. 16. Lab session scores for students with gamified feedback (treatment) versus students without gamified feedback (control).

2) Decreasing need for help: Figure 14 shows that the link between gamified feedback and need for help is significant $(p < 0.01)$ and negative (-0.402). During the lab sessions, students receiving gamified feedback requested fewer hints ($M = 4.78, SD = 7.73$) than students from the control group ($M = 14.62, SD = 20.82$) (Figure 17).

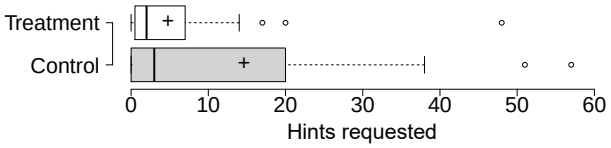

Fig. 17. Need for help (i.e., hints requested) for students with gamified feedback (treatment) versus students without gamified feedback (control).

VI. Discussion

This study provides the results of a semester-long field study with 115 non-CS students on the use, and the gamification, of an innovative computational notebook environment aimed at developing CT skills. The field study was conducted in an introductory course on information technology for first-year undergraduates in business and economics, where CT may not be seen as necessary by the students. This research assessed computational notebooks and CT skills making use of pre- and post-test surveys, learning analytics, and student-generated data from lab sessions. We showed that it is feasible

and valuable to teach CT competence to non-CS students, and that computational notebooks are an appropriate tool for introducing CT and programming to students with less technical academic backgrounds. Furthermore, our results convey the fact that gamification can increase engagement with these notebooks. A detailed discussion is presented below, with limitations and potential further work.

A. Computational notebooks in a distance learning context

The pre-post approach and the multifaceted pool of assessment tools that we used, allowed us to find that students gained CT skills both in terms of general problem-solving (CT score) and programming skills (Python score), as well as in terms of adopting more analytical problem-solving strategies (CT concepts). The integrated aspect of the computational notebooks allowed, among other things, students to avoid having multiple opened files and programs on their computers, which at the same time avoids installation issues. Still, the nature of such notebooks enabled dynamic class activities.

As such, this study endorses the notion that computational notebooks can support active learning scenarios for promoting CT skills in non-CS students (RQ1). Unsurprisingly, we found that initial knowledge—operationalized with the programming scores of the pre-test—was strongly linked to the learning outcomes. Yet, this study demonstrated that self-directed lab performance was an even more important predictor of the learning outcomes. The better a student performed in the online labs, the better their final grades. Furthermore, initial knowledge was not a significant predictor of self-directed lab performance, which indicated that students were not put off by the technology.

Our analyses showed that participating in the real-time lecture was associated with an increased pass rate compared to students who chose not to attend or only had limited engagement in the live online lectures. This finding supports previous research, which shows that active learning is more effective for knowledge acquisition [17]. However, this finding is not completely aligned with literature claiming the importance of active learning in both live and remote learning context [69], [70]. In fact, we admit some reservations about remote offline active teaching scenarios which did not, in this context and in contrast with real-time scenarios, have links with learning outcomes. These distance-teaching observations have been made in the particular context of COVID-19 in which this study took place, where in-class teaching was at that time not possible. Students could either follow the course in real-time online or watch recorded videos of the course at home. This study showed that the pass rate of the more active students was 76%, whereas the pass rate of the less active was only 47%. It should be noted, however, that around 32% of the students who took the exam were not engaged at all with the real-time lectures. This result should be seen in perspective with a previous preliminary study, which showed around 90% of students actively following the lecture in a *physical* in-class setting [10].

This work is not without limitations. Indeed, although this study covered one course during a full semester with 115 students, the conclusions could be more generalizable if the results integrated more courses with more instructors. In fact, our results are valid for our own class, but not necessarily for all other non-CS classes, as we do not know if this class would be representative for all other classes. From the point of view of remote learning, this study was able to demonstrate the strengths of tools such as Graasp, but the measure of remote engagement is always subject to variables that cannot be easily controlled. For instance, remote engagement via videos replayed offline is potentially subject to overcounting or undercounting as a video can be viewed by a student without them paying attention to it. Finer-grained learning analytics and links between several dissociated learning systems (LMS, computational notebook, and video repository) could inform about such differences. It would be interesting to conduct a similar study over a longer period of time with a larger number of participants allowing the positive long-term results to be verified. Future work could investigate the added value of lectures viewed on replay and on what kind of scenarios, the student's learning can be supported. This is especially important with remote teaching becoming more prevalent.

Another limitation lays in the simple but innovative approach used to measure the evolution of students' CT concepts. The approach was to ask participants to adopt a kind of think-aloud approach to describe how they approached the problem. The strategy used for the analysis was rudimentary (counting the frequency of keywords) and could be extended and improved in future work. This syntactic analysis of the cognitive approach to solving CT problems seemed to be an interesting line of investigation and also deserves to be explored further. It should be noted that we first conducted an analysis using advanced lexical analysis tools, such as the software called Linguistic Inquiry and Word Count (LIWC) to extract linguistic features [71], more precisely the 2015 version of the LIWC dictionary [72]. However, such tools did not appear to generate valid results as a simple and inconsistent problem-solving strategy description such as "Yes" received a higher score than some obviously more appropriate ones such as "By trial and error". Future research making use of problem-solving reflections and interpretation of these advanced tools such as LIWC deserve, in our opinion, to be conducted further.

B. Computational notebooks motivational aspects and gamification

This study has found motivation and gamified feedback to be strong predictors of lab performance. This finding supports previous research that finds that technological platforms can provide scaffolding for CT skills acquisition in gamified settings [45]. This study has shown how various motivational aspects may influence student performance and behavior in gamified settings. The results of this research have shown that the engagement with computational notebooks is associated with student situational motivation (RQ2). Specifically, we found that the influences of intrinsic and extrinsic motivation were completely different: while intrinsic motivation led to better lab performance and better learning outcomes, extrinsic motivation (as understood by external regulation) decreased

a student's self-regulation, making them look for more hints to quickly solve the task. Moreover, this study demonstrated that gamification can contribute to increased engagement with the computational notebook (RQ3). This study showed that gamified feedback influenced positively self-directed lab performance (score and engagement), and it also showed that gamified feedback influenced negatively unwanted behavior, such as the hints requested to complete the exercises in a quick and possibly less thoughtful way. We believe that the use of computational notebook environments such as Graasp, integrated with other applications, especially those introducing gamification, can open doors to other interesting avenues of research. The results presented in this article have already demonstrated the benefits of such a gamification approach, aligning with previous studies [44], [45] showing that the technological environment itself can successfully encourage feedback to drive CT skills learning in gamified settings. These results demonstrate that learning activity designers can encourage certain desired behaviors and at the same time discourage certain undesirable ones by using a well-designed gamification mechanism. Computational notebook environments—when used in combination with multiple instruments allowing us to assess students' CT expertise and covering different facets of students' knowledge acquisition—are clearly opening a new perspective, but it still needs to be investigated further.

Unfortunately, even though the sample size was adequate for the results presented above, it was too small to assess more fine-grained group differences and interaction effects (e.g., female vs. male, advanced vs. beginners). Future research should assess whether the effects of gamification play out in a similar direction for such subgroups. It is particularly important to confirm that students with less initial knowledge or who are less inclined toward CT are not left behind or negatively affected by certain gamification features. Based on such finer grained analysis, personalized gamification mechanisms could be deployed to adequately motivate students if no one-size-fits-all mechanism is found. We believe that future research and development on more integrated computational notebook environments will be encouraged by our results. The combination of learning analytics with real-time coding support and gamification features was central to our study. Nevertheless, most computational notebooks do not yet allow the composition of such rich learning activities. Open format and the availability of tracking data on student activities should be encouraged across the various computational notebook environments; this would potentially open the doors to further improvements in knowledge acquisition of complex skills such as CT.

VII. Conclusion

This paper addressed the issue of teaching CT to non-CS students. We conducted a field study in a real classroom with 115 students using a computational notebook app as support. This study evaluated computational notebook support for non-CS students from multiple perspectives. In order to evaluate the progress of the students in terms of competence in CT, we carried out a pre- and a post-test composed of problem-solving and programming questions. Students were also monitored during live lectures and self-directed lab sessions, allowing us to observe not only differences between real-time and replayed student engagement, but also influences of motivational aspects and gamified feedback on lab performance and learning outcome. We conclude by noting that computational notebooks can support active learning scenarios for promoting CT skills in non-CS students, that engagement with computational notebooks is associated with student motivation, and that gamification can contribute to increased engagement with the computational notebook. Finally, this study underlines the importance of continuing to investigate methods to engage people with little apparent interest in CT with active learning, computational notebooks and gamification mechanisms.

Acknowledgments

This research was partially funded by Swissuniversities through the P8 project titles Transversal CT—supporting responsible computational problem-solving across domains.

References

[1] J. C. Neubert, J. Mainert, A. Kretzschmar, S. Greiff *et al.*, "The assessment of 21st century skills in industrial and organizational psychology: Complex and collaborative problem solving," *Industrial and Organizational Psychology*, vol. 8, no. 2, pp. 238–268, 2015.

[2] D. Barr, J. Harrison, and L. Conery, "Computational Thinking: A Digital Age Skill for Everyone," *Learning & Leading with Technology*, vol. 38, no. 6, pp. 20–23, 2011.

[3] A. Yadav, N. Zhou, C. Mayfield, S. Hambrusch, and J. T. Korb, "Introducing Computational Thinking in Education Courses," in *Proceedings of the 42nd ACM Technical Symposium on Computer Science Education*. ACM, 2011, pp. 465–470.

[4] A. Juškevičienė and V. DagienĖ, "Computational thinking relationship with digital competence," *Informatics in Education*, vol. 17, no. 2, pp. 265–284, 2018.

[5] X. Tang, Y. Yin, Q. Lin, R. Hadad, and X. Zhai, "Assessing computational thinking: A systematic review of empirical studies," *Computers & Education*, vol. 148, p. 103798, 2020.

[6] A. Peterson, H. Dumont, M. Lafuente, and N. Law, "Understanding innovative pedagogies: Key themes to analyse new approaches to teaching and learning," 2018.

[7] A. Domínguez, J. Saenz-de Navarrete, L. De-Marcos, L. Fernández-Sanz, C. Pagés, and J.-J. Martínez-Herráiz, "Gamifying learning experiences: Practical implications and outcomes," *Computers & education*, vol. 63, pp. 380–392, 2013.

[8] J. Hamari, J. Koivisto, and H. Sarsa, "Does gamification work?–a literature review of empirical studies on gamification," in *2014 47th Hawaii international conference on system sciences*. Ieee, 2014, pp. 3025–3034.

[9] L. Leifheit *et al.*, "The role of self-concept and motivation within the" computational thinking" approach to early computer science education," Ph.D. dissertation, Eberhard Karls Universität Tübingen, 2021.

[10] J. C. Farah, A. Moro, K. Bergram, A. K. Purohit, D. Gillet, and A. Holzer, "Bringing Computational Thinking to non-STEM Undergraduates through an Integrated Notebook Application," in *Proceedings of the Impact Papers at the 15th European Conference on Technology-Enhanced Learning (EC-TEL 2020)*, T. Broos and T. Farrell, Eds., 2020.

[11] E. Eriksson, O. S. Iversen, G. E. Baykal, M. Van Mechelen, R. Smith, M.-L. Wagner, B. V. Fog, C. Klokmose, B. Cumbo, A. Hjorth *et al.*, "Widening the scope of fablearn research: Integrating computational thinking, design and making," in *Proceedings of the FabLearn Europe 2019 Conference*, 2019, pp. 1–9.

[12] D. Gillet, A. Vozniuk, M. J. Rodríguez-Triana, and A. Holzer, "Agile, Versatile, and Comprehensive Social Media Platform for Creating, Sharing, Exploiting, and Archiving Personal Learning Spaces, Artifacts, and Traces," in *The World Engineering Education Forum*, 2016.

[13] S. Papert, *Mindstorms: Children, computers, and powerful ideas*. Basic books, 1980.

[14] J. Wing, "Research notebook: Computational thinking—what and why," *The link magazine*, vol. 6, 2011.

[15] C. Wang, J. Shen, and J. Chao, "Integrating computational thinking in stem education: A literature review," *International Journal of Science and Mathematics Education*, pp. 1–24, 2021.

[16] C. Lu, R. Macdonald, B. Odell, V. Kokhan, C. Demmans Epp, and M. Cutumisu, "A scoping review of computational thinking assessments in higher education," *Journal of Computing in Higher Education*, pp. 1–46, 2022.

[17] R. DiYanni and A. Borst, "Active learning," in *The Craft of College Teaching*. Princeton University Press, 2020, pp. 42–60.

[18] L. Johnson, S. Adams Becker, M. Cummins, V. Estrada, A. Freeman, and H. Ludgate, *NMC Horizon Report: 2013 Higher Education Edition*, 2013.

[19] M. Oliver and K. Trigwell, "Can 'Blended Learning' Be Redeemed?" *E-learning and Digital Media*, vol. 2, no. 1, pp. 17–26, 2005.

[20] M. J. Rodríguez-Triana, L. P. Prieto, A. Vozniuk, M. S. Boroujeni, B. A. Schwendimann, A. Holzer, and D. Gillet, "Monitoring, Awareness and Reflection in Blended Technology Enhanced Learning: A Systematic Review," *International Journal of Technology Enhanced Learning*, vol. 9, no. 2-3, pp. 126–150, 2017.

[21] S. Van Goidsenhoven, D. Bogdanova, G. Deeva, S. vanden Broucke, J. De Weerdt, and M. Snoeck, "Predicting Student Success in a Blended Learning Environment," in *Proceedings of the 10th International Conference on Learning Analytics & Knowledge*, 2020, pp. 17–25.

[22] R. Collopy and J. M. Arnold, "To Blend or Not to Blend: Online-Only and Blended Learning Environments," *Issues in Teacher Education*, vol. 18, no. 2, 2009.

[23] Q. Chu, X. Yu, Y. Jiang, and H. Wang, "Data Analysis of Blended Learning in Python Programming," in *International Conference on Algorithms and Architectures for Parallel Processing*. Springer, 2018, pp. 209–217.

[24] S. Kross and P. J. Guo, "Practitioners teaching data science in industry and academia: Expectations, workflows, and challenges," in *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, 2019, pp. 1–14.

[25] A. Y. Wang, A. Mittal, C. Brooks, and S. Oney, "How Data Scientists Use Computational Notebooks for Real-Time Collaboration," *Proceedings of the ACM on Human-Computer Interaction*, vol. 3, no. CSCW, pp. 1–30, 2019.

[26] K. O'Hara, D. Blank, and J. Marshall, "Computational Notebooks for AI Education," in *The Twenty-Eighth International Flairs Conference*, 2015.

[27] J. M. Perkel, "Why Jupyter is Data Scientists' Computational Notebook of Choice," *Nature*, vol. 563, no. 7732, pp. 145–147, 2018.

[28] A. Radenski, ""Python First": A Lab-Based Digital Introduction to Computer Science," *SIGCSE Bulletin*, vol. 38, no. 3, pp. 197–201, 2006.

[29] A. Cardoso, J. Leitão, and C. Teixeira, "Using the Jupyter Notebook as a Tool to Support the Teaching and Learning Processes in Engineering Courses," in *Intl. Conference on Interactive Collaborative Learning*. Springer, 2018, pp. 227–236.

[30] M. Zastre, "Jupyter Notebook in CS1: An Experience Report," in *Proceedings of the Western Canadian Conference on Computing Education*, 2019, pp. 1–6.

[31] S. H. Edwards, D. S. Tilden, and A. Allevato, "Pythy: Improving the Introductory Python Programming Experience," in *Proceedings of the 45th ACM Technical Symposium on Computer Science Education*, 2014, pp. 641–646.

[32] P. J. Guo, "Online Python Tutor: Embeddable Web-Based Program Visualization for CS Education," in *Proceeding of the 44th ACM Technical Symposium on Computer Science Education*, 2013, pp. 579–584.

[33] Project Jupyter, D. Blank, D. Bourgin, A. Brown, M. Bussonnier, J. Frederic, B. Granger, T. Griffiths, J. Hamrick, K. Kelley, M. Pacer, L. Page, F. Pérez, B. Ragan-Kelley, J. Suchow, and C. Willing, "nbgrader: A Tool for Creating and Grading Assignments in the Jupyter Notebook," *Journal of Open Source Education*, vol. 2, no. 16, p. 32, 2019.

[34] A. Rule, A. Tabard, and J. D. Hollan, "Exploration and Explanation in Computational Notebooks," in *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, 2018, pp. 1–12.

[35] M. Borowski, J. Zagermann, C. N. Klokmose, H. Reiterer, and R. Rädle, "Exploring the Benefits and Barriers of Using Computational Notebooks for Collaborative Programming Assignments," in *Proceedings of the 51st ACM Technical Symposium on Computer Science Education*, 2020, pp. 468–474.

[36] M. Gagné and E. L. Deci, "The history of self-determination theory in psychology and management." 2014.

[37] R. J. Vallerand, "Toward a hierarchical model of intrinsic and extrinsic motivation," *Advances in experimental social psychology*, vol. 29, pp. 271–360, 1997.

[38] ——, "A hierarchical model of intrinsic and extrinsic motivation in sport and exercise," *Advances in motivation in sport and exercise*, vol. 2, pp. 263–319, 2001.

[39] E. L. Deci and R. M. Ryan, "Motivation and self-determination in human behavior," *NY: Plenum Publishing Co*, 1985.

[40] ——, "A motivational approach to self: Integration in personality." 1991.

[41] F. Guay, R. J. Vallerand, and C. Blanchard, "On the assessment of situational intrinsic and extrinsic motivation: The situational motivation scale (sims)," *Motivation and emotion*, vol. 24, no. 3, pp. 175–213, 2000.

[42] I. Kotini and S. Tzelepi, "A gamification-based framework for developing learning activities of computational thinking," in *Gamification in education and business*. Springer, 2015, pp. 219–252.

[43] D. L. Butler and P. H. Winne, "Feedback and self-regulated learning: A theoretical synthesis," *Review of educational research*, vol. 65, no. 3, pp. 245–281, 1995.

[44] L.-K. Lee, T.-K. Cheung, L.-T. Ho, W.-H. Yiu, and N.-I. Wu, "Learning computational thinking through gamification and collaborative learning," in *International Conference on Blended Learning*. Springer, 2019, pp. 339–349.

[45] C. W. Tan, P.-D. Yu, and L. Lin, "Teaching computational thinking using mathematics gamification in computer science game tournaments," in *Computational Thinking Education*. Springer, Singapore, 2019, pp. 167–181.

[46] F. Pires, F. M. M. Lima, R. Melo, J. R. S. Bernardo, and R. de Freitas, "Gamification and engagement: Development of computational thinking and the implications in mathematical learning," in *2019 IEEE 19th International Conference on Advanced Learning Technologies (ICALT)*, vol. 2161. IEEE, 2019, pp. 362–366.

[47] L. Gouws, K. Bradshaw, and P. Wentworth, "First year student performance in a test for computational thinking," in *Proceedings of the South African Institute for Computer Scientists and Information Technologists Conference*, 2013, pp. 271–277.

[48] V. Roadrangka, R. H. Yeany, and M. J. Padilla, "The construction and validation of group assessment of logical thinking (galt)," in *Paper Presented at the National Association for Research in Science Teaching Annual Meeting, Dallas, TX*, 1983.

[49] B. Kim, T. Kim, and J. Kim, "and-pencil programming strategy toward computational thinking for non-majors: Design your solution," *Journal of Educational Computing Research*, vol. 49, no. 4, pp. 437–459, 2013.

[50] M. Lafuente, O. Lévêque, I. Benítez, C. Hardebolle, and J. Dehler-Zufferey, "Assessing computational thinking: Development and validation of the algorithmic thinking test for adults," *Journal of Educational Computing Research*, in press.

[51] Ö. Korkmaz, R. Çakir, and M. Y. Özden, "A validity and reliability study of the computational thinking scales (cts)," *Computers in human behavior*, vol. 72, pp. 558–569, 2017.

[52] M. Yağcı, "A valid and reliable tool for examining computational thinking skills," *Education and Information Technologies*, vol. 24, no. 1, pp. 929–951, 2019.

[53] S. Herzog and N. A. Bowman, *Validity and Limitations of College Student Self-Report Data: New Directions for Institutional Research, Number 150*. John Wiley & Sons, 2011, vol. 110.

[54] T. T. Yuen and K. A. Robbins, "A qualitative study of students' computational thinking skills in a data-driven computing class," *ACM Transactions on Computing Education (TOCE)*, vol. 14, no. 4, pp. 1–19, 2014.

[55] S. Atmatzidou and S. Demetriadis, "Advancing students' computational thinking skills through educational robotics: A study on age and gender relevant differences," *Robotics and Autonomous Systems*, vol. 75, pp. 661–670, 2016.

[56] Y.-H. Lai, S.-Y. Chen, C.-F. Lai, Y.-C. Chang, and Y.-S. Su, "Study on enhancing aiot computational thinking skills by plot image-based vr," *Interactive Learning Environments*, pp. 1–14, 2019.

[57] T.-C. Hsu, S.-C. Chang, and Y.-T. Hung, "How to learn and how to teach computational thinking: Suggestions based on a review of the literature," *Computers & Education*, vol. 126, pp. 296–310, 2018.

[58] J. D. Hunter, "Matplotlib: A 2D Graphics Environment," *Computing in Science & Engineering*, vol. 9, no. 3, pp. 90–95, 2007.

[59] A. L. Strauss, *Qualitative analysis for social scientists*. Cambridge university press, 1987.

[60] A. Strauss and J. Corbin, *Basics of qualitative research*. Sage publications, 1990.

[61] J. Brooke, "Sus: a 'quick and dirty' usability scale," *Usability evaluation in industry*, vol. 189, 1996.
[62] J. Hair, C. L. Hollingsworth, A. B. Randolph, and A. Y. L. Chong, "An updated and expanded assessment of PLS-SEM in information systems research," *Industrial Management and Data Systems*, pp. 442–458, 2017.
[63] K. A. Bollen, "Latent variables in psychology and the social sciences," *Annual review of psychology*, pp. 605–634, 2002.
[64] N. Gillet, R. J. Vallerand, M.-A. K. Lafreniere, and J. S. Bureau, "The mediating role of positive and negative affect in the situational motivation-performance relationship," *Motivation and Emotion*, vol. 37, no. 3, pp. 465–479, 2013.
[65] S. Subhash and E. A. Cudney, "Gamified learning in higher education: A systematic review of the literature," *Computers in human behavior*, vol. 87, pp. 192–206, 2018.
[66] I. J. Quitadamo and M. J. Kurtz, "Learning to improve: using writing to increase critical thinking performance in general education biology," *CBE—Life Sciences Education*, vol. 6, no. 2, pp. 140–154, 2007.
[67] J. F. Hair Jr, G. T. M. Hult, C. Ringle, and M. Sarstedt, *A primer on partial least squares structural equation modeling (PLS-SEM)*. Sage publications, 2016.
[68] A. Bangor, P. T. Kortum, and J. T. Miller, "An empirical evaluation of the system usability scale," *Intl. Journal of Human–Computer Interaction*, vol. 24, no. 6, pp. 574–594, 2008.
[69] A. S. Kim, S. A. Khan, A. Carolli, and L. Park, "Investigating teaching and learning during the coronavirus disease 2019 pandemic." *Scholarship of Teaching and Learning in Psychology*, 2021.
[70] O. A. Pilkington, "Active learning for an online composition classroom: Blogging as an enhancement of online curriculum," *Journal of Educational Technology Systems*, vol. 47, no. 2, pp. 213–226, 2018.
[71] Y. R. Tausczik and J. W. Pennebaker, "The psychological meaning of words: Liwc and computerized text analysis methods," *Journal of language and social psychology*, vol. 29, no. 1, pp. 24–54, 2010.
[72] J. W. Pennebaker, R. L. Boyd, K. Jordan, and K. Blackburn, "The development and psychometric properties of liwc2015," Tech. Rep., 2015.

Alessio De Santo is a PhD candidate in Information Systems at the University of Neuchâtel. He holds a Bachelor of Science in Business Technology and a Master of Science in Information Systems. His current research interests cover digital support in learning and healthcare contexts. Recently, he joined the University of Applied Sciences of Western Switzerland HES-SO, Neuchâtel, as an assistant professor.

Juan Carlos Farah is a PhD candidate in Robotics, Control, and Intelligent Systems at the École Polytechnique Fédérale de Lausanne. He received a Bachelor of Arts in Economics from Harvard University and a Master of Science in Computing from Imperial College London. His current research focuses on human–computer interaction and the perception of anthropomorphic traits in intelligent conversational agents.

Marc Lafuente Martínez is a researcher, consultant, and project manager who focuses on new technologies and educational innovation. He earned his PhD in Educational Psychology at the University of Barcelona, where he also was a lecturer. He's been a policy analyst at the Organisation for Economic Cooperation and Development (OECD) and he collaborates with a number of government agencies and think-tanks. Recently, he has been a researcher at the École Polytechnique Fédérale de Lausanne (EPFL).

Arielle Moro is a Postdoctoral fellow at the University of Neuchâtel. She holds a PhD in Information Systems from the University of Lausanne. Her research thesis explored the tradeoff between predicting mobility and preserving location data privacy. Her research interests focus on analyzing human behavior in various domains (e.g., mobility, sustainability, learning) using machine learning techniques.

Kristoffer Bergram is a PhD candidate in Computer Science at the University of Neuchâtel. He holds Bachelor of Science degrees in Psychology and Statistics from Lund University and a Master of Science in Information Systems from the same university. His current research is focused on guiding human behavior in human–computer interaction contexts by using digital nudging.

Aditya Purohit is a PhD candidate in Information Systems at the University of Neuchâtel. He holds a Bachelor of Engineering in Computer Science and a Master of Science in Marketing Research from University Grenoble Alpes. His current research interests cover digital health, digital addiction, gamification, and building interventions to support digital wellbeing.

Pascal Felber is a Professor of Computer Science at the University of Neuchâtel. He received his PhD degree in Computer Science from the Swiss Federal Institute of Technology in 1998. He has since worked at Oracle Corporation and Bell-Labs (Lucent Technologies) in the USA, as well as at Institut EURECOM in France, before joining the University of Neuchâtel in 2004. His current research interests include digital education with a focus on the fields of dependable, concurrent, and distributed computing.

Denis Gillet received his PhD degree in Information Systems from the École Polytechnique Fédérale de Lausanne (EPFL) in 1995. Currently, he is a Faculty Member at the EPFL School of Engineering. He has been the technical coordinator for large-scale European innovation actions for STEM education in schools, and associate editor of the IEEE TLT and IJET. His current research interests include digital education, human–computer interaction, humanitarian technology, and ICT for development.

Adrian Holzer is a Professor of Management Information Systems at the University of Neuchâtel. He holds a PhD in Information Systems from the University of Lausanne. He was a research associate at École Polytechnique Fédérale de Lausanne, the co-head of the interdisciplinary platform at the University of Lausanne, and an SNF research fellow at Polytechnique Montréal. His research interests cover digitalization in learning and humanitarian contexts.