Education and Information Technologies (2024) 29:2999–3024
https://doi.org/10.1007/s10639-023-11956-6

The effect on computational thinking and identified learning aspects: Comparing unplugged smartGames with SRA-Programming with tangible or On-screen output

Nardie Fanchamps[1] · Emily van Gool[2] · Lou Slangen[2] · Paul Hennissen[2,3]

Received: 17 January 2023 / Accepted: 5 June 2023 / Published online: 16 June 2023

Abstract
Learning basic concepts of programming resulting in a development on computational thinking (CT) can be reached by means of digital programming environments. As a counterpart, the application of unplugged programming activities seems also to have promising potential regarding the impact on CT. The main characteristic of unplugged programming is that it comprises activities without the use of information and communication technologies (ICT). Since previous research has shown that the application of sense-reason-act (SRA) programming with different types of output demonstrated a better understanding of underlying complex programming concepts with an impact on CT, our research investigates whether the application of unplugged programming, offered via SmartGames, can also generate such a distinctive impact on developing CT. To capture the effects of the different interventions applied, a mixed-methods study was conducted among primary school students aged ten to twelve. Research data were obtained by means of a pretest–posttest questionnaire survey using the validated Computational Thinking Test (CTt), and by conducting interviews to determine the effects of CT and to ascertain identifiable learning effects. Our research indicates that unplugged programming by applying SmartGames can be a consummate regarding the development of CT, similar to SRA-programming using either robotics with tangible output or robot simulations with on-screen output. The research findings identified support our claim that the application of unplugged SmartGames shows equivalent development on CT subcharacteristics in comparison with plugged-in SRA-programming with tangible or on-screen output. A better understanding of complex programming concepts and positive identified learning effects could only be partly demonstrated.

Keywords Unplugged SmartGames · SRA-programming · Computational thinking · Visual programming · Tangible output · On-screen output

✉ Nardie Fanchamps
nardie.fanchamps@ou.nl

Extended author information available on the last page of the article

1 Introduction

Nowadays it is essential to be familiar with the competencies that are needed to function in a digital world (Iivari et al., 2020). These competencies can be acquired as early as primary school through learning programming (Stamati, 2020). Learning to program enables pupils to develop concepts from the world of computer science (Hogenboom et al., 2021). Primary education can provide a designated environment where programming can be taught, learned and practised (Sáez-López et al., 2019). Applications of programming can enhance pupils' problem-solving abilities (Sprankle & Hubbard, 2012). The ability to solve complex problems by making use of computer science concepts facilitates the development of computational thinking (CT) (Fanchamps, 2021; Nouri et al., 2020; Tedre & Denning, 2016).

Although developing computer-based programming skills is the most common approach to developing CT in schools, a systematic literature review of 125 studies focused on CT reveals that educators and teachers also deploy unplugged variants and alternatives in their prevailing teaching (Kalelioglu et al., 2016). However, while the effectiveness of computer programming for promoting CT skills is frequently studied (Lye & Koh, 2014), this is still only marginally applicable regarding the unplugged approach. Most studies with unplugged activities focused on promoting students' interest in computer science (Battal et al., 2021; Taub et al., 2012; Yildiz & Karal, 2021).

Contrary to common belief, it is not always necessary to apply digital tools and programming to develop skills associated with CT (Brackmann et al., 2017). Unplugged programming refers to learning computer science concepts without the use of digital technology (del Olmo-Muñoz et al., 2020). This is often called computer science unplugged (CSU) (Kalelioglu et al., 2016). A specific category of unplugged applications for learning computer science concepts are SmartGames that can train the memory, require and stimulate logical thinking, and can improve spatial awareness and problem-solving skills (Curzon & McOwan, 2017; Tsarava et al., 2018). By their nature, unplugged SmartGames contain characteristics that can be attributed to CT (Sharma et al., 2019; Sun et al., 2021). Therefore, the application of SmartGames to learn the principles of programming with a development on CT appears to be a promising concept for primary school practice. Furthermore, determining whether unplugged SmartGames can act as a trigger for learning appears to be essential (Prieta et al., 2013). In this regard, from previous research in CT learning, it is already known that in the application of plugged-in programming environments like on-screen simulations and robotics the perception, motivation and involvement of pupils is high (Slangen, 2016; Zapata-Cáceres & Fanchamps, 2021). Given the potentially expected results on CT, it is valuable to determine the effects on learning experiences when using unplugged SmartGames and whether a comparison with such plugged-in programming environments can be made.

Apart from unplugged activities by application of SmartGames, sense-reason-act (SRA) programming in visual environments with tangible or on-screen output

facilitates the development of CT by means of complex programming concepts (Fanchamps et al., 2022a). Previous research has shown that the type of task design, the programming environment used and the type of output can be of characteristic importance (Fanchamps et al., 2022b). SRA-programming appears to be of distinctive importance regarding the acquisition of a deeper understanding of complex programming concepts, resulting in a higher level of CT (Fanchamps, 2021). SRA-programming can be seen as an anticipatory method of solving programming problems by applying complex programming concepts, using visual programming environments with tangible or on-screen output. Given previous research indicated that SRA-programming allows for a better understanding of complex programming concepts that subsequently leads to an increased development of computational thinking (Rodriguez et al., 2017), this research therefore investigates whether unplugged programming applied via SmartGames has the same impact on gaining an understanding of complex programming concepts, resulting in a higher level of computational thinking, compared to visual SRA-programming with either a tangible or visual, on-screen output. It also examines the characteristics of learning experiences created by applying different task environments (unplugged—plugged-in).

2 Theoretical framework

Computational thinking (CT) encompasses more than merely solving problems using concepts derived from the world of computer science. It refers to thinking processes in which problem-oriented issues and their solutions can be reformulated and re-arranged in a form that they can be effectively executed by an information-processing digital agent (Dummer, 2017). Computational processes are a way of understanding natural and social phenomena and for explaining and interpreting the world as a composition of information processes (Denning & Tedre, 2019). Computing, as an ancient human practice, required computational thinking to design procedures and artefacts to automate them. Many aspects of computational thinking existed before the electronic computer and have been refined over time (Tedre & Denning, 2016). As the use of computers became increasingly common in the middle of the last century, the use of CT became a prerequisite for proper handling them. This implied the necessity to find a manner to define and to teach CT. From this perspective Denning and Tedre (2019) have addressed six dimensions (methods, machines, computing education, software engineering, design, computational science) that describe the construct of CT.

From a philosophical perspective, CT seems elaborated to the essence of what we can achieve by interacting with computers, as extensions of our minds, to create and discover (Knuth, 1980). A similar conceptual approach was also taken by Papert (1980), who focused on two aspects of computation: 1) how to use computation to create new knowledge, and 2) how to use computers to improve thinking and patterns of access to knowledge. After computational thinking was relegated to the background, Wing (2006) generated new attention to computational thinking and formulated a modified approach, considering CT as the fundamental ability to solve

challenging problems using skills abstracted from the world of computer science. From then on, CT has attracted renewed and prominent interest from scholars, and a multitude of definitions appear in the literature. In summary, these definitions for CT can be represented as "*the mental skills and practices for: a) designing computations that let computers do tasks for us, and b) explaining and interpreting the world as a complex of information processes"* (Denning & Tedre, 2019). In parallel with a further refinement of the definition of CT, the characteristics that define CT are also reducible from the literature, namely: problem decomposition, pattern recognition, data representation, generalisation, abstraction, and algorithms (Shute et al., 2017).

There are various ways to develop CT by means of educational activities (Hsu et al., 2018). On the one hand, this development can be achieved through the use of ICT-based activities. On the other hand, a development on CT can also be achieved without the application of digital technology using unplugged programming activities (Brackmann et al., 2017; Kuo & Hsu, 2020; Rodriguez et al., 2017).

Unplugged programming activities can be defined as learning to understand concepts derived from computer science without using digital technology (Bell et al., 1998). Unplugged programming has its origins in the movement of computer science unplugged (CSU) (Wahyudin et al., 2021). The rationale of CSUnplugged includes learning activities that teach computer science thinking without using digital technology. With this in mind, an approach emerged that introduced primary school pupils and teachers to off-line activities such as board games (Bell et al., 1998), and to help them place computer science in a broader perspective so that it encompasses more than just programming (Bell & Vahrenhold, 2018). In the studies undertaken by Lambert and Guiffre (2009), several positive results were obtained from CSUnplugged activities. Studies conducted by Rodriguez et al. (2017) and Thies and Vahrenhold (2013) with secondary school students showed that learning revenue on CT through unplugged activities indicated to be at least as effective as learning through the use of a digital approach. Furthermore, the study by Brackmann et al. (2017) found that primary school pupils who had participated in unplugged activities showed significant improvement in CT compared to pupils in the control group. Research conducted by del Olmo-Muñoz et al. (2020) shows that pupils can learn CT-skills by means of unplugged activities and that this form of unplugged learning is at least as effective as learning based on a computer aided approach. Moreover, studies conducted by Looi et al. (2018) and Relkin et al. (2020) show that primary school pupils, who had participated in unplugged activities, showed significant improvement on their level of CT compared to a group that used plugged programming activities.

In education, games can be used as a teaching aid in helping to explain or to reinforce the learning of computational concepts (Alvarez & Djaouti, 2011; Miller, 2008). The difference between 'normal games' and SmartGames is in the initial design process. Elements are added with the intention of evoking a variety of approaches and strategies to be employed or for acquiring specific content (De Freitas, 2018). SmartGames are logical thinking games which can stimulate the use of human memory, can improve spatial understanding and appeal to problem-solving thinking (Tsarava et al., 2018). SmartGames prompt thought processes among users for solving problems where the application goes beyond pure entertainment.

SmartGames can be used in order to propose and understand computer concepts, such as algorithms, parallel processing or data transfer. According to (Bakan & Bakan, 2018), playing a SmartGame enhances the process of inquiry based learning because it can put users in the required learning situations. Moreover, it stimulates interaction among learners, and can facilitate mastering complex tasks (Hung et al., 2014). In education, the use of games can be implemented to teach particular aspects like cleverness, insightfulness, speed, automation, knowledge and for cognitive challenges (Alvarez & Djaouti, 2011). Most games contain a core of required dexterity, insight and skills and can therefore be used for this purpose. Such elements characterize the difference between "regular games" and SmartGames. In addition, elements can be added that can evoke the user's cognitive challenge.

Robotic and virtual programming environments are considered powerful tools for learning the basic programming concepts (Caci et al., 2013; López et al., 2021). When applying sense-reason-act (SRA) programming, the more complex programming concepts such as iterations, conditionals and functions with parameters are used (Basu et al., 2016; Werner et al., 2012). SRA-programming asks for a more thorough understanding of the underlying operating principles rather than restricted linear, sequential programming structures (Martinez et al., 2015). The application of SRA requires logical reasoning of the ''if … then …'', "if … then … else, ''wait … until …'', ''repeat … until …'', "while …" kind. The complexity is rooted in the ability to think in terms of scenarios, and to understand and apply the more complex concepts of programming (Popat & Starkey, 2019). Programming according to the SRA approach is characterised by a multi stage process in which information obtained from sensor readings (*sense*) is fed into a reasoning component which compares these external conditions with pre-set values (*reason*), and the subsequent process of interferences that lead to initiated actions (*act*) (Lith, 2006). From previous research (Fanchamps, 2021) we know that a variety of ICT-based programming environments can ensure a growth in computational thinking, and that primary school pupils achieve higher levels of CT when applying SRA-programming either using robots or using simulations of reality with visual, on-screen output. We also know that the application of SRA-programming with an impact on CT depends on the type of task design and the environmental conditions in which programming are to be performed (Slangen, 2016). In addition we know that the application of SRA-programming enables a thorough understanding of the more complex concepts of programming resulting in a significant impact on computational thinking (Fanchamps, 2021).

Building on the theoretical exploration above, This research aims to explore opportunities to establish educational improvement, and specifically focuses on comparing applications of plugged-in and unplugged learning materials in order to identify characteristics and features from a didactic perspective regarding the development on CT. Although thorough examination of the literature shows that unplugged programming in a comparison with ICT-based activities indicates to be at least as effective concerning a development on CT, it is questionable whether this is also demonstrable when SRA-programming is applied. Moreover, previous research has shown that SRA-programming can achieve enhanced development on CT through an increased understanding of complex concepts of programming such as

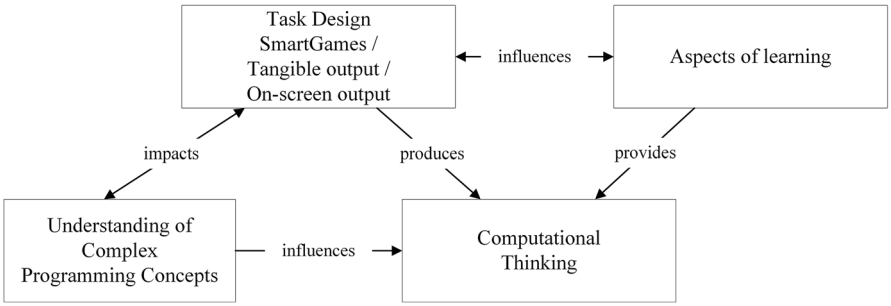

Fig. 1 Schematic representation of the conceptual model

the application of iterations, conditionals and functions. Therefore, we want to know whether the application of SmartGames can achieve similar effects. We hypothesise that the application of unplugged programming activities offered through SmartGames can have an equal impact on gaining insight into underlying complex programming concepts, and generate identifiable development in CT compared to visual plugged-in SRA-programming with tangible or on-screen output. In addition, we conjecture that through the application of unplugged SmartGames there is an influence on acquired learning experiences similar to the application of plugged-in SRA-programming. Our conceptual model, as displayed in Fig. 1, summarizes the relationships and interconnections between the independent and dependent variables, with some relationships being reciprocal.

2.1 Research questions

Grounded on the previous exploration, our main research question is: Whether and to what extent can unplugged programming activities applied via SmartGames, in comparison to SRA-programming tasks with tangible or visual output, show a similar development on complex programming concepts resulting in a development on computational thinking, and what aspects of learning emerge by means of such different task environments?

2.2 Sub-questions:

- Can unplugged SmartGames provide a full-fledged substitute for developing computational thinking?
- Does unplugged programming via SmartGames promote identifiable development on CT compared to SRA-programming with tangible or on-screen output?
- Do unplugged activities contain more aspects of CT development than just programming components?
- What indications regarding different aspects of learning (knowledge, experiences, added value, motivation, collaboration) can applications of various CT programming environments provide?

2.3 Hypothesis

- Unplugged programming applied by SmartGames shows equivalent development on computational thinking in comparison with plugged-in SRA-programming.
- Programming via unplugged SmartGames activities provide a better understanding of complex programming concepts compared to visual SRA-programming with tangible or on-screen output.
- The application of unplugged SmartGames activities produces more positive identified learning effects comparing SRA-programming with tangible or on-screen output.

3 Method

This comparative study should be seen as an exploratory approach to gain a better understanding of the effects of unplugged and plugged-in learning environments on CT development. As illustrated in Fig. 2, an exploratory mixed-methods design was conducted in which, by application of a pre-/post-test questionnaire survey and semi-structured interviews, both quantitative and qualitative data were obtained to (a) determine the effect of the interventions and comparing the impact; (b) to assess the associated hypotheses; and (c) to investigate the research questions. For the dependent variables, this includes among primary school pupils 1) pre-assessment of computational thinking skills (CTt), 2) for the intervention either the application of unplugged SmartGames activities, or two different visual programming environments with either tangible output (Lego EV-3) or on-screen output (Bomberbot), and 3) post-assessment of computational thinking skills (CTt) combined with semi-structured interviews.

In addition to the data obtained by applying the CTt, we also collected qualitative data by means of semi-structured interviews (open ended interview / one-to-one) (Creswell, 2008) with 4 standard questions (*'What would you like to share about working with SmartGames / Lego / Bomberbot?', 'How would you feel about*

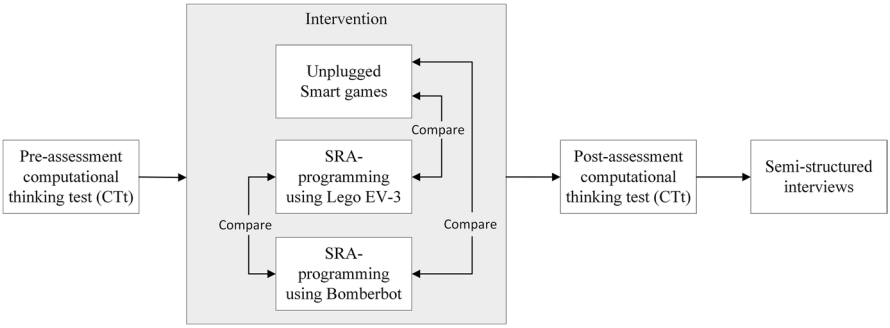

Fig. 2 Research design

working with SmartGames / Lego / Bomberbot at school?', 'What have you gained from working with SmartGames / Lego / Bomberbot?', 'How did you experience working together on the tasks with your buddy?'), and subsequent follow-up questions (e.g. *'Can you explain your answers a little more precisely?', 'Can you give an example?', 'Why do you think so?', 'What would you like to do differently yourself now that you have performed the tasks?', 'And why?', 'Would you like to do the tasks more often?', 'What did you gain the most and/or the least from?', 'What do you know now?', 'What can you do better now?', 'How do you feel about doing the tasks in dyads?', 'Could you have done this task alone and by yourself?', 'How would you prefer to do it?', 'Is there anything you would have liked to change?'*), asking pupils about their experiences at the end of the programming sessions. These questions where upfront benchmarked with a focus group of educational research experts. The aim of these interviews is to find out a) whether the application of unplugged SmartGame activities and/or plugged-in SRA-programming can facilitate pupils in their opinion about content development and knowledge gained, b) to find out how pupils stand in relation to the task and working with either SmartGames, Lego EV-3 and Bomberbot, and c) to determine whether pupils have different interesting learning experiences in different settings, and what these experiences contribute to supporting the hypotheses posed in this study.

3.1 Participants

This study was conducted with primary school pupils ranging in age from 8 to 13 years, from grade 5 and 6[1] ($N=35$), from a primary school in the Netherlands, and of which three experimental groups (SmartGames $n=12$ / Lego EV-3 $n=10$ / Bomberbot $n=13$) were randomly composed. None of the participating pupils were familiar with programming beyond the use of basic computer programs such as Word, PowerPoint and the Internet, and none of them had previously taken a CT test. All pupils from the primary school involved were arbitrarily assigned to one of three experimental conditions (SmartGames, Lego Ev-3, Bomberbot). This ensured that each experimental group was a realistic reflection of the population concerned.

3.2 Materials

To answer our research questions, we used three different programming intervention setups which vary in characteristics. These consisted of the application of 1) unplugged SmartGames, 2) Lego EV-3 Mindstorms© robots with tactile output, and 3) Bomberbot© with visual on-screen output.

[1] In this publication we use the UK grade level system to indicate the research population. Grade 5 and 6 in the UK corresponds with the Dutch "group 7 and 8".

Fig. 3 Selected unplugged SmartGames

3.2.1 Unplugged smartGames

As an application for unplugged programming, several different SmartGames are used. These games can be played without the use of a computer and are characterised by required logical thinking and strategic action taking. As a prerequisite, the selected SmartGames should have an increasing level of difficulty and ascending level of abstraction, and should be playable with two players against each other. First of all, we selected a number of educational SmartGames varying in structure and complexity, which are appropriate for pupils in the age group of 9–12 years. Subsequently, in order to determine relevant characteristics with respect to logic, critical and creative thinking, as addressed by Jonassen (2000) (evaluating, imagining, elaborating, evaluating, analysing, connecting), of the pre-selected SmartGames, an educational games expert was consulted who, specifically with the population concerned, determined what the build-up in difficulty and level of abstraction of each game should be, and what transfer could be made from each game to which content-related link to other school subjects. In addition, the findings and recommendations of the game expert were benchmarked with a focus group of experts who helped determine what the structure in application should be. Figure 3 shows the selected unplugged SmartGames.

Based on these three starting points, we determined the selection and sequence of the SmartGames applied. These are games that support learning regarding 1) how to recognise a problem, 2) how to identify the most important details of a problem, 3) how to divide a problem into small, logical steps in order to create a problem-solving process, and 4) to evaluate the learning process afterwards. To promote the development of CT skills, a logical increase of difficulty is a prerequisite and the 'luck element', as often used in games, should be avoided. Also, these games should enable collaborative learning processes. Since pupils play two games in one hour, an additional criterion was that the explanation of a game should not exceed three minutes. Based on these criteria, a game specialist was subsequently consulted to arrive at a final selection of eight games. Subsequently, the finally selected games

Table 1 Allocated Computational Thinking characteristics per SmartGame

	Selected unplugged SmartGames							
CT-skill	*Froggit*	*Checkers*	*4 in a Row*	*Teblo*	*Sea Battle*	*Cube Duel*	*Mastermind*	*Topspot*
(Re)formulating problems	x	x–	x	x	x–	x	x	x
Collecting data								
Analysing data	x	x –	x	x	x	x	x	x
Visualisation data					x			
Problem decomposition					x			x
Abstraction	x	x	x	x	x	x	x	x
Algorithms and procedures								
Automation								
Simulation and modelling		x	x	x	x	x	x	
Parallelisation							x	

were played beforehand by an expert-focus team in order to determine which characteristics of computational thinking skills can be promoted by playing each game. Table 1 shows the selected games and renders the allocated computational thinking characteristics for each game.

3.2.2 Plugged-in lego mindstorms EV-3

Lego EV-3 Mindstorms robots are used as a tangible programming environment. Characteristic for this visual oriented programming environment is that a program to be executed by a robot is constructed by using the "drag and drop method" by which programming commands must be arranged onto a worksheet in the correct sequence. Each robot is equipped with various sensors (tactile sensor, ultrasonic sensor, colour sensor) that allows the robot to react to changing conditions in the task environment by means of its program to be constructed based on sensor input (e.g. manoeuvring through various labyrinth setups, avoid obstacles, follow a line, etc.). Applied in this way, reactive SRA-programming enables the application of thinking in parallel routines. Figure 4 shows a SRA-programming solution devised using Lego Mindstorms software, and Fig. 5 shows the Lego EV-3 robots with sensors mounted used.

3.2.3 Plugged-in bomberbot

Bomberbot is a purely visually oriented programming environment that is a simulation of reality. This environment also uses the “drag and drop” programming method to construct a program to be executed by a simulated robot mounted with virtual sensors. This robot must manoeuvre through a labyrinth, solve tasks, smash rubies

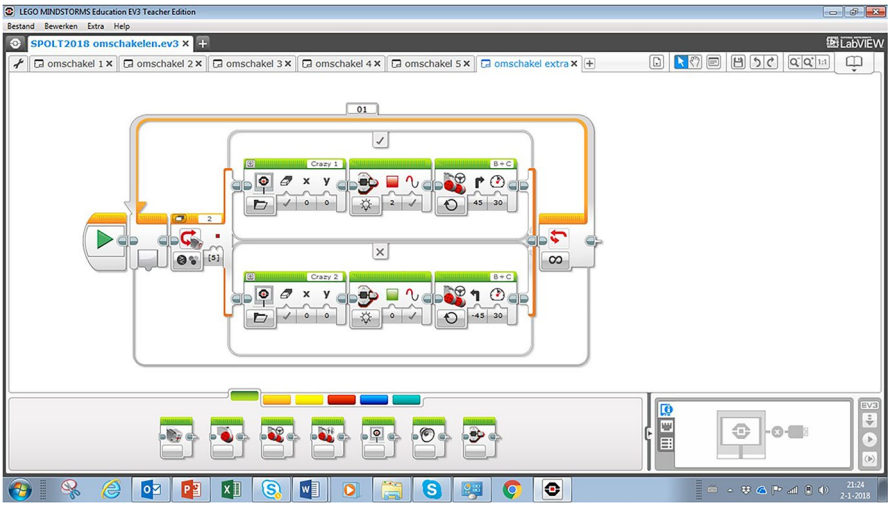

Fig. 4 SRA-programming Lego EV-3 Mindstorms©

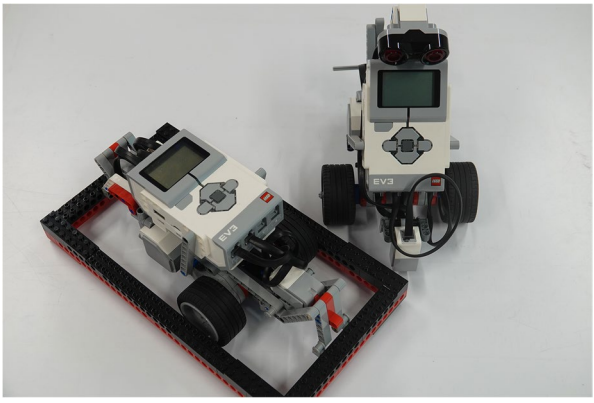

Fig. 5 Used robots Lego EV-3 Mindstorms©

and open treasure chests. This programming environment also enables the application of SRA-programming by means of thinking in parallel programming routines. Figure 6 shows an example of SRA-programming in Bomberbot.

3.2.4 Computational thinking assessment

To determine the level of CT between the pre- and post-assessments, we used the validated Computational Thinking test (CTt) (Román-González et al., 2017). This test makes it possible to generate information on the level of solving CT tasks, the understanding of the computational concepts involved, and the understanding of complex programming concepts (hypotheses 1 and hypotheses 2). All pupils involved in this research completed this questionnaire individually. The questionnaire

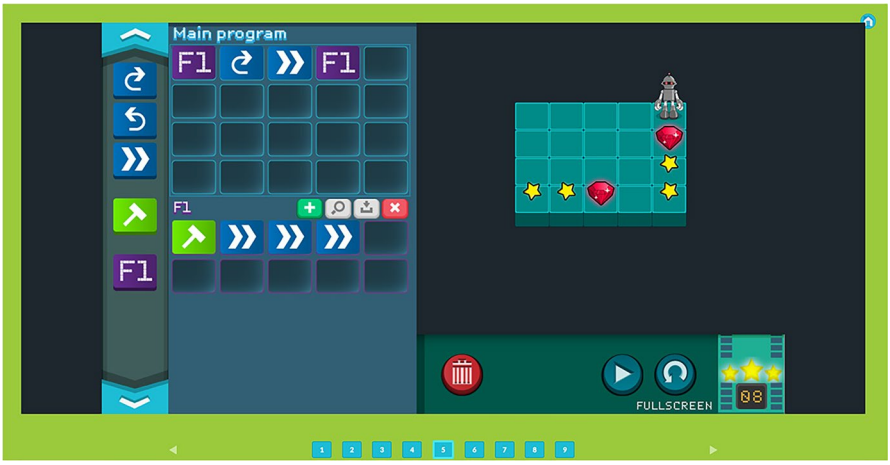

Fig. 6 SRA-programming in Bomberbot©

contained a total 28 items that relate to the various computational concepts involved, i.e. basic directions (28 items), loops (repeat times: 13 items, repeat until: 12 items), conditionals (if-simple: six items, if/else-complex: four items, while: four items) and functions (simple functions: four items, functions with parameters: zero items). The existence or nonexistence of nesting can be derived (19 items). The computational task required (completion: nine items, debugging: five items, sequencing: 14 items) to provide the right solution for each of the 28 questionnaire items can be inferred. To determine the reliability of the scale, we calculated Cronbach's alpha. It should be noted that a value for Cronbach's alpha of 0.70 is considered an acceptable reliability factor (Santos, 1999). The developers of the CTt indicate that for fifth and sixth graders ($N=176$), Cronbach's alpha should be $\alpha=0.721$ (Román-González et al., 2017). We identified a value of Cronbach's alpha of $\alpha=0.686$, which almost complies with the required level of internal consistency for our scale in this particular sample. An explanation for this slightly lower value for Cronbach's alpha can be found in the fact that the research sample size was smaller than that for which the designers of the CTt validated the test. In addition, the age category of the research population included in this study was towards the lower limit of the test, which may be the reason for the lower reliability. Taking this into account, the CTt performed almost as reported by the original authors, and the measurement results obtained were therefore used as such.

3.3 Procedure

As a pre-test assessment (Fig. 2), all three groups completed the CTt. Subsequently, the group that was to use unplugged SmartGames first received a basic explanation regarding the game approach. Furthermore, the application of SmartGames consisted of five one-hour sessions. In each hour, pupils played two games against each other in dyads. After half an hour, pupils switched to a new, more difficult game. A

high form of motivation is a prerequisite for playing the games. To stimulate this motivation, an additional competition element was built in. Each pupil noted their own score for each game and gave an indication of how they experienced each game ("a lot of fun", "boring", "difficult"). At the beginning of each session, pupils were shown the score everyone had achieved. The fifth session was a final challenge in which pupils, according to the achieved score-ranking, played against each other with the game that had received the highest score. The group that were to program with Lego EV-3 Mindstorms robots received basic instruction and then completed 15 programming tasks, including five final challenge assignments in five one-hour sessions, by applying SRA-programming. The group that was to use Bomberbot completed, after a short introduction, 10 programming missions in five one-hour sessions using SRA-programming, each consisting of 15 programming tasks including challenge assignments.

To analyse the quantitative data collected from the questionnaire survey, a variance analysis (Anova) and Levene's test were initially conducted for all variables to distil preliminary indications and to assess whether equal variances should be assumed. Since specific hypotheses were formulated in this study, a subsequent contrast analysis was carried out to demonstrate possible significant effects and to confirm or reject these hypotheses. To make the magnitude of the effects visible, Hedges g was calculated.

The pre- and post-measurement results from the CTt were entered into SPSS for quantitative data analysis, and the effects of the independent variables on the dependent variables were assessed. The differences in values were determined by comparing the means. In all our statistical analyses, a significance level of 5% ($p = \leq 0.05$) was assumed. The nature of the data met the conditions for the assumption of normality, indicating that the distribution of sample means (across independent samples) was normal. We tested whether assumptions of the homogeneity of variances were violated ($p \leq 0.05$). Degrees of freedom were calculated, and a bootstrapping procedure was applied to re-estimate the standard error of the mean difference. Based on statistical analysis, differences between the averages (means) were studied. Subsequently, it was determined whether the 0-value was within the confidence interval. To assess the normality of the variables, the Shapiro–Wilk test was applied ($p = 0.766$), which was above the chosen alpha level of $\alpha = 0.05$. Furthermore, the null hypothesis could not be rejected, as there was evidence that the tested data were normally distributed. The histograms also indicated a normal distribution to which could be assumed that all variables were normally distributed. The value for the extent of the effect size (Hedges g) for different sample sizes was calculated (it should be noted that $g = 0.2$ can be considered a small effect size, $g = 0.5$ represents a medium effect size, and $g = 0.8$ indicates a large effect size) (Field, 2013).

4 Results and data analysis

Our main research objective, to determine whether and to what extent unplugged programming activities applied via SmartGames, in comparison to SRA-programming tasks with different types of output (using Lego EV-3 with tangible output and Bomberbot with on-screen output), show a similar development on complex

programming concepts resulting in a development on computational thinking, and what aspects of learning emerge by means of such different task environments, is answered by analysing the means for all the variables measured in this research. We aimed to explore whether; (i) unplugged programming applied by SmartGames shows equivalent development on computational thinking in comparison with plugged-in SRA-programming; (ii) programming via unplugged SmartGames activities provide a better understanding of complex programming concepts compared to visual SRA-programming with tangible or on-screen output; and (iii) the application of unplugged SmartGame activities produces more positive identified learning effects comparing SRA-programming with tangible or on-screen output.

4.1 Differences computational thinking development

To provide a structured overview regarding differences in CT development pertaining to the three experimental groups, our study uses a subdivision into CT subcategories. These are derived from the same subdivision established by (Román-González et al., 2017) in the validated CTt used. Table 2 displays the data for each of the CT subcategories regarding the three intervention groups.

The analysis of the averages obtained from pretest–posttest assessment (Table 2) shows that applying unplugged SmartGames, in comparison to applying plugged in SRA-programming using Bomberbot or Lego EV-3 showed (a) increased values for solving CT tasks successfully, (b) increased values regarding control over loops, conditionals and functions, (c) increased values for the use of nesting, and (d) increased values regarding sequencing, completion and debugging for the required task application. Both the group that applied unplugged SmartGames as well as the groups that applied SRA-programming with Bomberbot or Lego EV-3 showed higher average scores (*M*) for all identified variables in the post-assessment. The mean values should be interpreted as being the average number of correctly answered questions per respondent, standardised to a value of zero to one. This progression can also be derived from the calculated percentage values for each separately measured variable, calculated for the three different intervention groups. This percentage calculation was included as such because the three different intervention groups differed in respondent size (SmartGames: $n=12$; Lego EV-3: $n=10$; Bomberbot: $n=13$). The percentage values per sub-category were calculated for the number of items per sub-category divided to the total number of items in the questionnaire, multiplied by the calculated mean (*M*) (for example calculation pre-test loops “repeat times”: $13/28\times0.47=21.82\%$). By illuminating characteristic differences between intervention groups by calculating percentages, an objective comparison could be made regarding the effect and impact on CT by applying either unplugged SmartGames or both plugged-in SRA programming environments. For all combined sub-categories, percentages were calculated based on the weighted averages. The category "loops: combined" is an amalgamation of the subcategories "loops: repeat times" (13 items) and "loops: repeat until" (12 items). Since both sub-categories have an overlap of three items, the percentage calculation for the category "loops: combined" is based on a total of 22 items. The category "conditionals: combined" is an amalgamation of the sub-categories "conditionals: if-simple" (6

Table 2 Differences computational thinking development

Variable	SmartGames				Lego EV-3				Bomberbot			
	M	*SD*	*range*	%	*M*	*SD*	*range*	%	*M*	*SD*	*range*	%
Pre-test: CT tasks correctly solved (28 items)	0.56	0.07	0.43–0.68	56.00	0.58	0.18	0.21–0.79	58.00	0.48	0.11	0.32–0.68	48.00
Post-test: CT tasks correctly solved (28 items)	0.64	0.10	0.50–0.82	64.00	0.71	0.15	0.50–0.82	71.00	0.58	0.13	0.39–0.79	58.00
Pre-test loops: repeat times (13 items)	0.47	0.16	0.15–0.69	21.82	0.49	0.19	0.08–0.77	22.75	0.45	0.16	0.23–0.85	20.89
Post-test loops: repeat times (13 items)	0.62	0.13	0.38–0.85	28.79	0.68	0.16	0.54–1.00	31.57	0.54	0.18	0.31–0.85	25.07
Pre-test loops: repeat until (12 items)	0.51	0.12	0.33–0.75	21.86	0.58	0.16	0.33–0.83	24.86	0.43	0.12	0.25–0.58	18.43
Post-test loops: repeat until (12 items)	0.60	0.14	0.42–0.83	25.71	0.63	0.17	0.25–0.83	27.00	0.55	0.16	0.33–0.83	23.57
Pre-test loops: combined (22 items)	0.49	0.09	0.29–0.64	38.50	0.54	0.17	0.21–0.76	42.43	0.44	0.11	0.32–0.67	34.57
Post-test loops: combined (22 items)	0.61	0.11	0.44–0.80	47.93	0.66	0.16	0.39–0.92	51.86	0.55	0.15	0.32–0.84	43.21
Pre-test conditionals: if-simple (6 items)	0.43	0.13	0.17–0.67	9.21	0.48	0.23	0.00–0.83	10.29	0.38	0.25	0.00–0.83	8.14
Post-test conditionals: if-simple (6 items)	0.50	0.21	0.00–0.67	10.71	0.58	0.24	0.17–0.83	12.43	0.44	0.14	0.17–0.67	9.43
Pre-test conditionals: if/else (4 items)	0.54	0.28	0.00–1.00	7.71	0.55	0.11	0.50–0.75	7.86	0.34	0.19	0.00–0.50	4.86
Post-test conditionals: if/else (4 items)	0.58	0.22	0.25–1.00	8.29	0.65	0.27	0.25–1.00	9.29	0.45	0.28	0.00–1.00	6.43
Pre-test conditionals: while (4 items)	0.44	0.28	0.00–0.75	6.29	0.40	0.17	0.25–0.75	5.71	0.45	0.28	0.00–1.00	6.43
Post-test conditionals: while (4 items)	0.46	0.21	0.25–0.75	6.57	0.55	0.23	0.25–1.00	7.86	0.46	0.26	0.00–0.75	6.57
Pre-test conditionals: combined (12 items)	0.47	0.10	0.25–0.58	20.14	0.48	0.15	0.25–0.72	20.57	0.39	0.14	0.14–0.58	16.71
Post-test conditionals: combined (12 items)	0.51	0.18	0.17–0.81	21.86	0.59	0.19	0.22–0.94	25.29	0.45	0.17	0.14–0.67	19.29
Pre-test functions: simple (4 items)	0.46	0.23	0.00–0.75	6.57	0.48	0.22	0.25–1.00	6.86	0.34	0.25	0.00–0.75	4.86
Post-test functions: simple (4 items)	0.71	0.18	0.50–1.00	10.14	0.65	0.24	0.25–1.00	9.29	0.48	0.30	0.00–1.00	6.86
Pre-test use of nesting (19 items)	0.45	0.08	0.26–0.58	30.54	0.48	0.14	0.32–0.74	32.57	0.37	0.12	0.21–0.63	25.11
Post-test use of nesting (19 items)	0.56	0.13	0.32–0.74	38.00	0.56	0.16	0.32–0.84	38.00	0.47	0.16	0.21–0.68	31.89
Pre-test required task completion (9 items)	0.55	0.17	0.33–0.89	17.68	0.52	0.21	0.22–0.89	16.71	0.52	0.15	0.22–0.78	16.71
Post-test required task completion (9 items)	0.66	0.11	0.44–0.78	21.21	0.69	0.16	0.44–0.89	22.18	0.60	0.14	0.33–0.78	19.29
Pre-test debugging (5 items)	0.52	0.22	0.00–0.80	9.29	0.52	0.32	0.00–0.80	9.29	0.37	0.27	0.00–0.80	6.61
Post-test debugging (5 items)	0.62	0.22	0.20–1.00	11.07	0.72	0.23	0.20–1.00	12.86	0.49	0.24	0.00–0.80	8.75

Table 2 (continued)

Variable	SmartGames				Lego EV-3				Bomberbot			
	M	*SD*	*range*	%	*M*	*SD*	*range*	%	*M*	*SD*	*range*	%
Pre-test sequencing (14 items)	0.53	0.11	0.36–0.71	26.50	0.61	0.20	0.14–0.86	30.50	0.51	0.13	0.36–0.71	25.50
Post-test sequencing (14 items)	0.63	0.15	0.36–0.86	31.50	0.71	0.16	0.57–1.00	25.50	0.59	0.15	0.36–0.86	29.50

M = average; SD = standard deviation; range = spread in measurement; % = percentage values related to the total number of items in the questionnaire multiplied by the calculated mean (M)

items), "conditionals: if/else" (4 items) and "conditionals: while" (4 items). Since there are two items that overlap between these three sub-categories, the percentage calculation for the category "loops: combined" is based on a total of 12 items. Grounded on the data obtained, it can be claimed that the application of unplugged SmartGames and plugged-in SRA-programming with Bomberbot or Lego EV-3, equally cause this identified increase. This is despite the fact that for some CT sub-characteristics in the post-assessment only a slight increase in measured values could be observed. Proportionally, all three intervention groups performed better on all measured CT sub-characteristics comparing the pre- and post-assessment of computational thinking.

4.2 Understanding of complex programming concepts

To better understand the impact of the different interventions applied, concerning unplugged SmartGames or plugged-in SRA programming, and whether this led to an increased understanding of complex programming concepts, a contrast analysis on all variables was performed (Table 3).

Table 3 Contrast analysis with a comparison of unplugged SmartGames and SRA-programming for all groups

Group	SmartGames compared to Lego EV-3			SmartGames compared to Bomberbot			Lego EV-3 compared to Bomberbot		
Variable	*t*	*p*	*g*	*t*	*p*	*g*	*t*	*p*	*g*
Total (28)	–1.215	0.233–	0.560	–1.308	0.200–	0.514	2.499	0.018*–	0.936
Loops: repeat times	–0.817	0.420 –	0.416	–1.344	0.188–	0.506	2.122	0.041*–	0.815
Loops: repeat until	–0.433	0.668	0.195	–0.817	0.420–	0.332	1.223	0.230–	0.487
Loops: combined	–0.705	0.486–	0.371	–1.219	0.231–	0.453	1.887	0.068–	0.713
Conditionals: if-simple	–0.992	0.328–	0.357	–0.771	0.446–	0.339	1.759	0.088–	0.739
Conditionals: if/else	–0.601	0.552–	0.287	–1.344	0.188–	0.514	1.898	0.066–	0.725
Conditionals: while	–0.914	0.367–	0.411	–0.065	0.949–	0.003	0.884	0.383–	0.364
Conditionals: combined	–1.048	0.302–	0.433	–0.899	0.375–	0.343	1.938	0.061–	0.783
Functions	–0.545	0.589–	0.287	–2.300	0.028*	0.920	1.622	0.114–	0.616
Nesting	–0.027	0.978–	0.014	–1.485	0.147–	0.621	1.439	–0.160—	0.563
CT task: completion	–0.538	0.594–	0.223	–1.009	0.321–	0.474	1.514	–1.139—	0.604
CT task: debugging	–1.036	0.308–	0.445	–1.429	0.162–	0.564	2.430	–0.021*–	0.976
CT task: sequencing	–1.270	0.213–	0.518	–0.644	0.524–	0.267	1.926	–0.063—	0.777

Variable = measurable value; total = number of questions correct CT questionnaire; computational concept addressed = loops, conditionals, functions, nesting; completion = completed by CT; debugging = reformulating of problems; sequencing = sequence; t = t-value contrast analysis; p = p-value contrast analysis; g = effect size based on Hedges g for different sample sizes; * = significant effect measured

The contrast analysis of the total number of correctly solved CT tasks (28) shows that there was a significant difference, with a large measurable effect, between the groups that applied SRA-programming using Lego EV-3 compared to Bomberbot t (33) = 2.499, p = 0.018, g = 0.936. No significant difference could be measured between the groups that used unplugged SmartGames and applied SRA-programming using either Lego EV-3 (p = 0.233) or Bomberbot (p = 0.200).

A contrast analysis of the differences in the application of "loops: repeat times" shows that there was significant difference with a large measurable effect, between the groups that applied SRA-programming using Lego EV-3 compared to Bomberbot t (33) = 2.122, p = 0.041, g = 0.815. No significant difference could be measured between the groups that used unplugged SmartGames and applied SRA-programming using either Lego EV-3 (p = 0.420) or Bomberbot (p = 0.188).

A contrast analysis of the differences in the application of "functions" shows that there was a significant difference with a large effect between the group that used unplugged SmartGames and SRA-programming using Bomberbot t (33) = –2.300, p = 0.028, g = 0.920, No significant difference could be measured between the group that used unplugged SmartGames and SRA-programming using Lego EV-3 (p = 0.589), and no significant difference could be measured between the group that applied SRA-programming using Lego EV-3 or Bomberbot (p = 0.114).

A contrast analysis of the differences in the required task "debugging" shows that there was a significant difference, with a large measurable effect, between the groups that applied SRA-programming using Lego EV-3 compared to Bomberbot t (33) = 2.430, p = 0.021, g = 0.976. No significant difference could be measured between the groups that used unplugged SmartGames and applied SRA-programming using either Lego EV-3 (p = 0.308) or Bomberbot (p = 0.162).

4.3 Identified learning effects attributed

A qualitative analysis by means of the semi-structured interviews conducted provided additions and further explanations to what our indications from the quantitative data showed. These interviews were conducted after the final work session in which pupils applied either unplugged SmartGames or SRA programming by usage of Lego EV-3 or Bomberbot. Pupils were asked about a) what they would like to share about working with SmartGames / Lego / Bomberbot, b, how they feld about working with SmartGames / Lego / Bomberbot at school, c) what their gain was from working with SmartGames / Lego / Bomberbot, and d) how they experienced working together on the tasks in dyads. Regardless the programming environment used, pupils showed a very high motivation to participate with each of the programming environments offered. The most common feelings pupils expressed when talking about their experiences were those of happiness and joy, and that solving problems made them feel smart. All pupils who applied SRA-programming by using Lego EV-3 robotics or by application of unplugged SmartGames described these as enjoyable and fun. Of the pupils who applied SRA-programming by using Bomberbot, 50% agreed with this opinion, while the rest of this group found it sometimes difficult or boring. Pupils who applied unplugged SmartGames specifically

expressed their growth in knowledge and skills, namely: "being able to think about strategies better" (25%), "targeted application of skills in the classroom" (25%), and "being better able to predict what the other person is doing" (25%). The answers of the pupils who applied Lego EV-3 robotics were more general in nature: 70% indicated that they could handle the robot controller better, 20% felt that they could handle programming better. Pupils who applied Bomberbot had the most difficulty putting into words what they had learned. Although 55% said they had learned more about programming in general, however, 29% said they had no idea where or how they had developed. Regarding the question posed to pupils about their experience of working together in dyads, all pupils who applied unplugged SmartGames (100%) perceived it as pleasant and enjoyable, followed by the group that applied Bomberbot (86%). Remarkably, only 50% of the pupils who applied Lego EV-3 robotics expressed the same experience.

5 Discussion

The fundamental premise of our research was to investigate whether and to what extent unplugged programming activities applied via SmartGames, in comparison to SRA-programming tasks with tangible or visual output, show a similar development on complex programming concepts resulting in a development on computational thinking, and what aspects of learning emerge by means of such different task environments.

An examination of the contrast analysis performed shows that the influence of different types of programming environments applied, either unplugged SmartGames or plugged-in SRA programming, can have significant effects on the development of multiple sub-characteristics of CT. Noteworthy are the significant values we have indicated with an * in Table 3.

Further interpretation of the reported results reveals that, depending on the characteristics of the different programming environment applied, a significant development in sub-characteristics of CT can be observed. A plausible explanation can be found in the fact that Bomberbot, characterised by its visual, on-screen output, promotes the use of iterative, cause-and-effect reasoning more effectively by means of its applied task design, and that unplugged SmartGames, through the strategic and logical thinking required as game elements to ultimately be smarter than an opponent in order to win, facilitates the understanding and application of functions. Also, when playing a game, forward-thinking is always required to anticipate what the opponent might do and what steps in the thinking scenario the opponent might have included. This strengthens our conviction that the application of unplugged SmartGames develops more aspects of CT than just programming components. The difference in focus to more directly perceive the impact of the programming intervention in Bomberbot can also explain the significant results found. Bomberbot appears to be more targeted towards handling parallel programming interventions, whereas Lego EV-3 still leaves much room for finding programming solutions by means of deterministic programming. It is also noticeable that Bomberbot incorporates incentives that encourage pupils to keep searching for the most optimal and efficient

programming solution, which seems to have a determining effect in the development of sub-characteristics of CT.

From our qualitative findings obtained distilled from the semi-structured interviews, it can be inferred that participants who applied unplugged SmartGames or Lego EV-3 were highly motivated and enthusiastic during all the programming sessions. The level of motivation among participants who applied Bomberbot was lower for half of the participants. From the participants who applied unplugged SmartGames could 75% indicate which (CT) skills they had developed. From the participants that applied Bomberbot could 55% indicate a development on CT, and from the participants that applied Lego EV-3 could 50% indicate this development. In this regard, it is valuable to mention that the formulation of what was learned by the groups that either applied Bomberbot or Lego EV-3 was more general in nature ('learnt more about programming'), than the group that applied unplugged SmartGames who were able to articulate more specifically their development gone through. Furthermore, participants were mostly very positive about working in pairs. Of the group that applied Bomberbot, 86% said they enjoyed working together, while of the group that applied Lego EV-3, 50% said they positively valued working together. Working together in pairs is a prerequisite for playing the SmartGames. All participants in this group (100%) reported enjoying this collaboration. As a valuable side effect, respondents in this group also noted that they got to know their classmate better.

Our research includes explanations for the observed results on CT that could be established by applying unplugged SmartGames and SRA-programming. Examples include SmartGames to solve complex problems more strategically, tangible robotics to understand iterations better and for debugging CT tasks, and on-screen simulations to generically learn more about programming. Pupils working with these kind of interventions develop demonstrable cognitive skills/tools that enable them to tackle issues transversely.

Only a limited number of studies have been conducted on the effects of SRA-programming and the impact on CT through the application of different types of interventions (Fanchamps, 2021; Slangen, 2016). Consequently, little is still known about the added value of SRA-programming regarding development on CT. Our study seeks to shed more light on this. In part, hypotheses and questions are results of theoretical exercises.

Our findings indicate that the power and possibilities regarding the application of unplugged SmartGames in education should not be underestimated in terms of their contribution to a development focused on computational thinking as claimed by Rodriguez et al. (2017) and Thies and Vahrenhold (2013). The exploratory research conducted shows that growth in a multitude of characteristics related to CT is perfectly possible without the application of digital computer technology as also stated by Brackmann et al. (2017) and del Olmo-Muñoz et al. (2020). Significant results were found on specific sub-characteristics of CT even by applying unplugged SmartGames in a direct comparison with SRA-programming by applying programmable robots or on-screen simulations of reality. This paves the way for learning environments where no digital technology or computer equipment are available, or where it is required to adapt to learners' needs.

Similar to SRA-programming, the application of unplugged SmartGames requires anticipating unforeseen situations which trigger up following actions, and the characteristics of specific tasks in which circumstances can change/vary which asks for more strategically handling. This involves an appeal to imagining what could possibly happen in a SmartGame, how to react to such changing circumstances and to be able to devise generic and specific approaches and solutions in advance. This manifests itself in strategies called for while playing a SmartGame. Herein, the explanation can be found regarding a demonstrable development on all CT characteristics that is closely aligned with the SRA approach as included in this study. In fact, as also stated by Brennan and Resnick (2012) and Denning and Tedre (2019), this goes in parts even beyond developing aspects of CT that relate only to programming and computing.

In addition, we noticed and obtained indications that discussing the qualitative questions asked in the semi-structured interviews in this study may have led to an increased and progressive awareness of the pupils. In our view, besides retrieving research data, this also harbours a pedagogical function of encouraging pupils to articulate from awareness the progression of their own development with increase in skill. Something we successively register as an increase in CT. It would be valuable in follow-up research to further investigate this assumed awareness effect for CT learning.

6 Conclusions

From an examination of the quantitative data obtained, it can be inferred that the group that applied unplugged SmartGames (based on the differences identified between pre- and post-assessment) showed equivalent growth for all variables compared to the application of the two other experimental SRA-programming conditions (Bomberbot or Lego EV-3). This supports our assumption that the application of unplugged SmartGames can also lead to an over-all increase in CT. A further interpretation of the available data shows that there was a notable increase on specific sub-characteristics of CT by application of either unplugged SmartGames, or by application of SRA-programming using the two different visual programming environments.

Comparing the application of SRA-programming with Bomberbot or Lego EV-3, it can be inferred that the application of Lego EV-3 causes a significant and large effect on the development on CT in general. Furthermore, the comparison between the application of SRA-programming with Bomberbot or Lego EV-3 shows that the application of Lego EV-3 is the main culprit in terms of the significant effect found on the application of “loops: repeat times” and for the CT task “debugging”. The application of unplugged SmartGames is found to have a significant effect on the application of "functions" in a comparison with the application of SRA-programming with Bomberbot. It can therefore be assumed that by applying unplugged SmartGames and both visual SRA-programming environments with different types of output (tangible – on-screen), pupils (1) solved more CT tasks correctly; (2) solved more questions correctly

which required the application of loops, conditionals, functions and nesting; and (3) showed an increase in the application of required CT tasks (e.g., completion, debugging, and sequencing). This suggests that, like the application of SRA-programming, the application of unplugged SmartGames also contributes to a better understanding of complex programming concepts.

The hypothesis that unplugged programming applied by SmartGames shows equivalent development on computational thinking in comparison with plugged-in SRA-programming can be confirmed. This can be inferred from the increase in the total number of correctly solved CT-tasks and from the substantial progression visible on all measured CT sub-characteristics. It should be noted that, despite an increase on all measured values, a significant development could only be identified for the application of "functions" by application of unplugged SmartGames in a direct comparison to the application of SRA-programming with either Bomberbot or Lego EV-3.

The hypothesis that programming via unplugged SmartGame activities provide a better understanding of complex programming concepts compared to visual SRA-programming with tangible or on-screen output can be confirmed for the value "functions" compared to Bomberbot. The values for "loops: repeat times" and the CT-task "debugging" did indicate increased values for the application of unplugged SmartGames, but proved significant only in the comparison between SRA programming with Bomberbot and Lego EV-3.

The hypothesis that the application of unplugged SmartGame activities produces more positive identified learning effects comparing SRA-programming with tangible or on-screen output can only partly be proven. Our analysis shows that the application of unplugged SmartGames contributes to the (further) development of CT on the basis of their characteristics (strategic thinking, logical reasoning) and ensures a high level of engagement, and working in dyads is also highly valued. Our qualitative analysis also confirms that the pupils' perceptions as distilled from semi-structured interviews corresponded most closely to the development of their CT skills and the opportunities embedded in them as established in the quantitative assessment. It can be inferred that pupils show evidence of having become more aware of the skills we classify as CT skills and which are the meaningful steps for pupils to develop themselves further. Noting that an explanation for this can also be found in the language pupils use when applying SmartGames because it fits games, while the language used when applying Lego-EV3 and Bomberbot differs from everyday language in terms of CT.

7 Limitations and follow-up research

Several limitations and further considerations can be identified regarding the results of this research. Due to these, a certain lack of generalisability of the results obtained should be taken into account.

This study involved a relatively small target group population. Despite being aware of this limited research sample, this study has produced interesting findings

that can provide valuable leads for further research. In order to increase the reliability and probative value of the qualitative data, it is recommended to repeat this study with larger groups of respondents.

Due to the age range of the respondents included in this study and the use of the validated computational thinking test (CTt), it is valuable to repeat the study with pupils older than the population included in this study who are towards the lower limit of the test. This is to further increase the validity and reliability of the results obtained.

It may be the case that pupils were already familiar with some of the selected and applied unplugged SmartGames, or may have worked with Lego EV-3 robotics or Bomberbot before. This may have undesirably affected the results identified.

This study used the application of visual programming environments that differ in type of output. It is recommended to investigate whether other types of programming environments generate similar outputs to those currently observed.

The in this research applied quantitative assessment instrument was more attuned to the use of digital programming environments. It would be worthwhile to replicate this research with other CT assessment tools which also adapt more to unplugged programming environments.

It would also be interesting in subsequent research to compare a development on CT by applying unplugged SmartGames with non-SRA-based programming environments.

Acknowledgements The authors would like to thank Bomberbot Netherlands and Heutink Netherlands for making the programming environments available and for their cooperation. We would also like to thank the primary school and pupils involved for participating in this study.

Authors' contributions All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Nardie Fanchamps. The first draft of the manuscript was written by Nardie Fanchamps and all authors commented on previous versions of the manuscript. All authors read and approved the revised and final manuscript.

Data availability Can be requested from the first author.

Code availability Retrievable on request from Bomberbot© and Heutink with personal login code.

Declarations

Ethics approval The Ethical research board (cETO) of the Open University of the Netherlands has assessed the proposed research and concluded that this research is in line with the rules and regulations and the ethical codes for research in Human Subjects (reference: U2019/01324/SVW).

Conflicts of interest/Competing interests The authors declare that they have no conflict of interest/no competing interests.

Consent to participate Written informed consent was obtained from the parents of the individual participants.

Consent to publish The parents of the individual participants consented the data obtained to be published.

References

Alvarez, J., & Djaouti, D. (2011). An introduction to Serious game Definitions and concepts. Serious games & simulation for risks management, Paris, France.

Bakan, U., & Bakan, U. (2018). Game-based learning studies in education journals: A systematic review of recent trends. *Actualidades Pedagógicas, 72*(72), 119–145. https://doi.org/10.19052/ap.5245

Basu, S., Biswas, G., Sengupta, P., Dickes, A., Kinnebrew, J. S., & Clark, D. (2016). Identifying middle school students' challenges in computational thinking-based science learning. *Research and Practice in Technology Enhanced Learning, 11*(1), 13. https://doi.org/10.1186/s41039-016-0036-2

Battal, A., Afacan Adanır, G., & Gülbahar, Y. (2021). Computer Science Unplugged: A Systematic Literature Review. *Journal of Educational Technology Systems, 50*(1), 24–47. https://doi.org/10.1177/00472395211018801

Bell, T., & Vahrenhold, J. (2018). CS Unplugged—How Is It Used, and Does It Work? In H.-J. Böckenhauer, D. Komm, & W. Unger (Eds.), *Adventures Between Lower Bounds and Higher Altitudes: Essays Dedicated to Juraj Hromkovič on the Occasion of His 60th Birthday* (pp. 497–521). Springer International Publishing. https://doi.org/10.1007/978-3-319-98355-4_29

Bell, T., Witten, I. H., & Fellows, M. (1998). *Computer Science Unplugged: Off-line activities and games for all ages*. Citeseer.

Brackmann, C. P., Román-González, M., Robles, G., Moreno-León, J., Casali, A., & Barone, D. (2017). Development of computational thinking skills through unplugged activities in primary school. Proceedings of the 12th Workshop on Primary and Secondary Computing Education, Nijmegen, Netherlands.

Brennan, K., & Resnick, M. (2012). New frameworks for studying and assessing the development of computational thinking. *AERA*, 25.

Caci, B., D'Amico, A., & Chiazzese, G. (2013). Robotics and Virtual Worlds: An experiential learning lab. In *Biologically Inspired Cognitive Architectures 2012* (pp. 83–87). Springer. https://doi.org/10.1016/j.sbspro.2013.10.070

Creswell, J. W. (2008). *Educational Research: Planning, conducting and evaluating quantitative ad quantitative research* (3e ed.). Pearson Education Inc.

Curzon, P., & McOwan, P. W. (2017). The power of computational thinking: Games, magic and puzzles to help you become a computational thinker. *World Scientific*. https://doi.org/10.1142/9781786341853_0001

De Freitas, S. (2018). Are games effective learning tools? A review of educational games. *Journal of Educational Technology & Society, 21*(2), 74–84.

del Olmo-Muñoz, J., Cózar-Gutiérrez, R., & González-Calero, J. A. (2020). Computational thinking through unplugged activities in early years of Primary Education. *Computers & Education, 150*, 19. https://doi.org/10.1016/j.compedu.2020.103832

Denning, P. J., & Tedre, M. (2019). *Computational thinking*. MIT Press. Retrieved March 16, 2021, from https://books.google.nl/books?hl=nl&lr=&id=WzyUDwAAQBAJ&oi=fnd&pg=PR7&dq=computational+thinking+denning+tedre&ots=VD_muWPYJq&sig=MLpBQ9RKnvndMYyF9G-3scGHcU8&redir_esc=y#v=onepage&q=computational%20thinking%20denning%20tedre&f=false

Fanchamps, N. (2021). *The influence of sense-reason-act programming on computational thinking Open University]*. Heerlen.

Fanchamps, N., Slangen, L., Specht, M., & Hennissen, P. (2022a). Effect of SRA-programming on computational thinking through different output modalities. *Journal of Computers in Education*, 1–30. https://doi.org/10.1007/s40692-022-00236-w

Fanchamps, N., Specht, M., & Hennissen, P. (2022b). The Effect on Computational Thinking Using SRA-Programming: Anticipating Changes in a Dynamic Problem Environment. *IEEE Transactions on Learning Technologies, 15*(2), 213–222. https://doi.org/10.1109/TLT.2022.3162895

Hogenboom, S. A., Hermans, F. F., & Van der Maas, H. L. (2021). Computerized adaptive assessment of understanding of programming concepts in primary school children. *Computer Science Education*, 30. Retrieved September 22, 2022, from https://www.tandfonline.com/action/showCitFormats?doi=10.1080/08993408.2021.1914461

Hsu, T.-C., Chang, S.-C., & Hung, Y.-T. (2018). How to learn and how to teach computational thinking: Suggestions based on a review of the literature. *Computers & Education, 126*, 296–310. https://doi.org/10.1016/j.compedu.2018.07.004

Hung, C.-M., Huang, I., & Hwang, G.-J. (2014). Effects of digital game-based learning on students' self-efficacy, motivation, anxiety, and achievements in learning mathematics. *Journal of Computers in Education, 1*(2), 151–166. https://doi.org/10.1007/s40692-014-0008-8

Iivari, N., Sharma, S., & Ventä-Olkkonen, L. (2020). Digital transformation of everyday life–How COVID-19 pandemic transformed the basic education of the young generation and why information management research should care? *International Journal of Information Management, 55*(102183), 1–6. https://doi.org/10.1016/j.ijinfomgt.2020.102183

Kalelioglu, F., Gulbahar, Y., & Kukul, V. (2016). A framework for computational thinking based on a systematic research review. *Baltic Journal of Modern Computing*, 14.

Knuth, D. E. (1980). *Algorithms in Modern Mathematics and Computer Science*.

Kuo, W.-C., & Hsu, T.-C. (2020). Learning computational thinking without a computer: How computational participation happens in a computational thinking board game. *The Asia-Pacific Education Researcher, 29*(1), 67–83. https://doi.org/10.1007/s40299-019-00479-9

Lambert, L., & Guiffre, H. (2009). Computer science outreach in an elementary school. *Journal of Computing Sciences in Colleges, 24*(3), 118–124.

Lith, P. V. (2006). *Masterclass Robotica*

Looi, C.-K., How, M.-L., Longkai, W., Seow, P., & Liu, L. (2018). Analysis of linkages between an unplugged activity and the development of computational thinking. *Computer Science Education, 28*(3), 255–279. https://doi.org/10.1080/08993408.2018.1533297

López, J. M. S., Otero, R. B., & García-Cervigón, S. D. L. (2021). Introducing robotics and block programming in elementary education. *Revista Iberoamericana De Educación a Distancia, 24*(1), 95–113. https://doi.org/10.5944/ried.24.1.27649

Lye, S. Y., & Koh, J. H. L. (2014). Review on teaching and learning of computational thinking through programming: What is next for K-12? *Computers in Human Behavior, 41*, 51–61. https://doi.org/10.1016/j.chb.2014.09.012

Martinez, C., Gomez, M. J., & Benotti, L. (2015). A comparison of preschool and elementary school children learning computer science concepts through a multilanguage robot programming platform. Proceedings of the 2015 ACM Conference on Innovation and Technology in Computer Science Education,

Miller, C. T. (2008). *Games: Purpose and potential in education*. Springer Science & Business Media.

Nouri, J., Zhang, L., Mannila, L., & Norén, E. (2020). Development of computational thinking, digital competence and 21st century skills when learning programming in K-9. *Education Inquiry, 11*(1), 1–17. https://doi.org/10.1080/20004508.2019.1627844

Papert, S. (1980). *Mindstorms, children, computers and powerful ideas*. Basic Books, inc.

Popat, S., & Starkey, L. (2019). Learning to code or coding to learn? A systematic review. *Computers & Education, 128*, 365–376. https://doi.org/10.1016/j.compedu.2018.10.005

Prieta, F. D. l., Mascio, T. D., Marenzi, I., & Vittorini, P. (2013). Pedagogy-Driven Smart Games for Primary School Children. 2nd International Workshop on Evidence-based Technology Enhanced Learning,

Relkin, E., de Ruiter, L., & Bers, M. U. (2020). TechCheck: Development and validation of an unplugged assessment of computational thinking in early childhood education. *Journal of Science Education and Technology, 29*, 482–498. https://doi.org/10.1007/s10956-020-09831-x

Rodriguez, B., Kennicutt, S., Rader, C., & Camp, T. (2017). Assessing computational thinking in CS unplugged activities. Proceedings of the 2017 ACM SIGCSE Technical Symposium on Computer Science Education,

Román-González, M., Pérez-González, J.-C., & Jiménez-Fernández, C. (2017). Which cognitive abilities underlie computational thinking? Criterion validity of the Computational Thinking Test. *Computers in Human Behavior, 72*, 678–691. https://doi.org/10.1016/j.chb.2016.08.047

Sáez-López, J.-M., Sevillano-García, M.-L., & Vazquez-Cano, E. (2019). The effect of programming on primary school students' mathematical and scientific understanding: Educational use of mBot. *Educational Technology Research and Development, 67*(6), 1405–1425. https://doi.org/10.1007/s11423-019-09648-5

Sharma, K., Papavlasopoulou, S., & Giannakos, M. (2019). Coding games and robots to enhance computational thinking: How collaboration and engagement moderate children's attitudes? *International Journal of Child-Computer Interaction, 21*, 65–76. https://doi.org/10.1016/j.ijcci.2019.04.004

Shute, V. J., Sun, C., & Asbell-Clarke, J. (2017). Demystifying computational thinking. *Educational Research Review, 22*, 142–158.

Slangen, L. (2016). *Teaching robotics in primary school*. Eindhoven University of Technology. Retrieved October 3, 2019, from https://pure.tue.nl/ws/files/25754482/20160630_CO_Slangen.pdf

Sprankle, M., & Hubbard, J. (2012). *Problem solving and programming concepts* (M. Hirsch, Ed.). Pearson Education Inc.

Stamati, M. (2020). The Importance of ICT in Primary Education: Interpretive Schemes and Practices in the Island of Lesvos. *Multilingual Academic Journal of Education and Social Sciences, 8*(1), 168–182. https://doi.org/10.46886/MAJESS/v8-i1/7276

Sun, L., Guo, Z., & Hu, L. (2021). Educational games promote the development of students' computational thinking: a meta-analytic review. *Interactive Learning Environments*, 15. https://doi.org/10.1080/10494820.2021.1931891

Taub, R., Armoni, M., & Ben-Ari, M. (2012). CS unplugged and middle-school students' views, attitudes, and intentions regarding CS. *ACM Transactions on Computing Education (TOCE), 12*(2), 1–29. https://doi.org/10.1145/2160547.2160551

Tedre, M., & Denning, P. J. (2016). The long quest for computational thinking. Proceedings of the 16th Koli Calling International Conference on Computing Education Research, Koli, Finland.

Thies, R., & Vahrenhold, J. (2013). On plugging" unplugged" into CS classes. Proceeding of the 44th ACM technical symposium on Computer science education.

Tsarava, K., Moeller, K., & Ninaus, M. (2018). Training computational thinking through board games: The case of Crabs & Turtles. *International Journal of Serious Games, 5*(2), 25–44. https://doi.org/10.17083/ijsg.v5i2.248

Wahyudin, W., Rishanty, A. M., Nursalman, M., Nazir, S., & Riza, L. S. (2021). Learning through computer science unplugged on team assisted individualization on the computational thinking ability. *Linguistics and Culture Review, 5*(S3), 1442–1452. https://doi.org/10.21744/lingcure.v5nS3.1841

Werner, L., Campe, S., & Denner, J. (2012). Children learning computer science concepts via Alice game-programming. Proceedings of the 43rd ACM technical symposium on Computer Science Education.

Wing, J. M. (2006). Computational Thinking. *Communications of the ACM, 49*(3), 33–35. https://doi.org/10.1145/1118178.1118215

Yildiz, M., & Karal, H. (2021). A Computer Science Unplugged Activity: CityMap. *International Journal of Computer Science Education in Schools, 5*(2), 14–27. https://doi.org/10.21585/ijcses.v5i2.110

Zapata-Cáceres, M., & Fanchamps, N. (2021). Using the Beginners Computational Thinking Test to Measure Development on Computational Concepts Among Preschoolers. 5th APSCE International Computational Thinking and STEM in Education Conference 2021, Singapore.

Authors and Affiliations

Nardie Fanchamps[1] · Emily van Gool[2] · Lou Slangen[2] · Paul Hennissen[2,3]

✉ Nardie Fanchamps
nardie.fanchamps@ou.nl

[1] Open University, Valkenburgerweg 177, 6419 AT Heerlen, Netherlands

[2] Fontys University of Applied Science, Mgr. Claessenstraat 4, 6131 AJ Sittard, Netherlands

[3] Zuyd University of Applied Science, Nieuw Eyckholt 300, 6419 DJ Heerlen, Netherlands