DOI: 10.34641/ctestem.2022.451

The Effect of Unplugged Programming and Visual Programming on Computational Thinking in Children Aged 5 to 7

Lisa BOSGOED[1*], Nardie FANCHAMPS[2]
[1]OBS De Graswinkel, Netherlands
[2]Open University, Netherlands

lisabosgoed@hotmail.com, nardie.fanchamps@ou.nl

ABSTRACT
This research focuses on the development of computational thinking (CT) among one-hundred and eight primary school pupils in the Netherlands aged five to seven years. It compares the use of unplugged programming and visual programming with on-screen output. In addition to the effect of using different programming environments, this research also establishes whether age differences and prior knowledge of programming have an additional influence. By means of a pretest-posttest design, using the validated quantitative instrument TechCheck, possible differences between the development of CT in both experimental groups and a control group could be objectively determined. To this end, pupils from both experimental groups have applied during five programming sessions of forty-five minutes each either unplugged story introduced smart games or used the plugged-in programming environment ScratchJr. Our results show a significant difference in CT development between unplugged programming and visual programming with on-screen output. Moreover, unplugged programming had a more positive effect on the development of CT compared to the control group than visual programming with on-screen output. A moderating effect could be attributed to age differences and prior knowledge of programming. This may provide an additional explanation regarding the identified impact and significant differences found.

KEYWORDS
Unplugged programming, computational thinking, smart games, primary education

2. INTRODUCTION
In recent decades, society has changed through various technological developments from an industrially oriented society to a mostly digitally focused knowledge community (Organization for Economic Co-operation and Development [OECD], 2008). To cope with this change, 21st century skills provide educational direction so that people can continue to develop in a focused way in order to function optimally. Computational thinking (CT) is an essential skill for making this transition. CT can be described as a set of problem-solving skills based on fundamental concepts from computer science and can be seen as a fundamental skill that is required in many everyday activities (Wing, 2006). The skill of CT can be promoted by different programming environments. However, little is known about the extent to which the differences and deviating characteristics of various programming environments can contribute to the development of CT skills (Brackman et al., 2017; Rose et al., 2017). We distinguish between a) plugged-in programming in which programming skills can be acquired by entering instructions and commands into a computer via graphical or tactile user interfaces using textual, visual or tactile programming languages resulting in on-screen output or tactile output; and b) unplugged programming where skills related to programming can be acquired without the use of a computer or digital processing agent. Results from previous research show that different design aspects of learning environments can have an effect on learning outcomes. For example, the extent to which the working memory is strained depends on prior knowledge and the way information is represented (concrete, iconic or symbolic) (Paas & Van Merriënboer, 2020). In addition, the children's development in each successive phase also plays a prominent role, from learning by physically manipulating perceptible objects to mental manipulation of more abstract or visual information (Sigelman & Rider, 2012).

3. PURPOSE OF THE STUDY
The aim of this study is to explore the effect that the type of programming environment and the associated characteristic differences have on the development of CT in young children. The research question is as follows: "Is there a measurable difference in effect on the development of computational thinking between unplugged programming and visual programming with on-screen output in children aged 5 to 7, controlling for age and prior knowledge of programming?"

4. METHOD
A quantitative, quasi-experimental study was conducted to determine the potential effects of the type of programming environment on the development of CT. Various schools were approached to participate in the study.

To determine the effect, a pretest-posttest design was applied. Children were non-randomly assigned into three research groups: unplugged programming, visual programming with on-screen output and a control group. Due to the COVID-19 pandemic, one school participated as the control group to reduce the number of contacts. As a pre- and posttest measurement, TechCheck was used as a validated instrument to determine the level of CT. As an intervention, children from both experimental groups were offered five programming lessons. Children from the control group participated in programming lessons after the study.

5. MATERIALS
To answer the research question, various unplugged smart games and ScratchJr, a plugged-in programming environment, were used to promote programming skills

(such as algorithms, loops and conditionals). All programming activities and games were carried out in collaboration. Children were offered one programming activity or game per lesson. Three-dimensional board games (e.g. Robot Turtles, Little Red Riding Hood and Sleeping Beauty) were used as unplugged smart games, where problems must be solved by applying sequential, manual steps (Brackman et al., 2017). In Robot Turtles, for example, players first need to arrange cards, which are included in the game, with written or pictographic commands such as "forward", "backward", "left", "right" and "jump". Then they have to move their turtle manually, according to the instruction, to receive a diamond. ScratchJr, as a plugged-in environment, is designed to teach young children programming within a two-dimensional environment (Rose et al., 2017). Instructions on the screen are created via graphical user interfaces by the drag-and-drop method. Instructions are created using blocks, which can be dragged from a library, that are pictographically displayed and represent commands. In the main program, these can be structured sequentially and in parallel. To apply a constructed instruction, the play button is pressed. ScratchJr offers various design aspects that allow children to create interactive animations, games and storylines. To determine the level of CT in the pretest and posttest, TechCheck was used. TechCheck has been validated in a group of 5- to 9-year-olds who participated in a study of visual programming with tangible output (Relkin et al., 2020). Results from the classical theory test and item response test show reliability and validity (α = .69). TechCheck measures CT as one construct using 15 multiple choice questions, which have a strong pictographic character. Furthermore, TechCheck do not distinguish between CT skills such as algorithmic thinking, problem decomposition or pattern recognition.

6. FINDINGS

Table 1 displays the results from the pretest and posttest. From this data it can be deduced that the posttest measurements show a higher average score and a lower standard deviation than the pretest measurements. Children from all experimental groups answered more questions correctly in the posttest than in the pretest. However, no significant differences were found between any groups $F(2.105) = 1.863$; $p = .160$. Comparing the averages (M) regarding the development of CT, the unplugged programming group had a higher mean score than the group that programmed using a visual environment with on-screen output and the control group.

Table 1. Means and Standard Deviations of CT

	Pretest	Posttest
Unplugged programming (n = 33)	11.48 (1.91)	12.21 (1.90)
Visual programming (n = 37)	9.05 (3.15)	9.08 (2.99)
Control group (n = 38)	11.32 (3.04)	11.42 (2.46)

After correcting means, significant differences were found between unplugged programming and visual programming with on-screen output, controlling for age (p = .008) and prior knowledge of programming (p = .042), as shown in Table 2.

Table 2. Means for Development of CT

	Before correction	Covariate age	Covariate prior knowledge
Unplugged programming	.73	.98	.90
Visual programming	.03	-.20	-.22
Control group	.11	.11	.20

Note. Covariate age groups unplugged: 5 years (n = 2), 6 years (n = 17), 7 years (n = 14); visual: 5 years (n = 12), 6 years (n = 18), 7 years (n = 7); control: 5 years (n = 7), 6 years (n = 20), 7 years (n = 11). Covariate prior knowledge unplugged: none (n = 0), few (n = 2), many (n = 31); visual: none (n = 9), few (n = 14), many (n = 14); control: none (n = 3), few (n = 2), many (n = 33).

7. CONCLUSION

Our research indicated that unplugged programming can play a prominent role in the development of CT, where age differences and prior knowledge of programming are of characteristic influence. In total, age has a moderate effect on the development of CT (η^2 = .09) and prior knowledge has a small-to-moderate effect (η^2 = .06). To generalise from our findings, more research is needed with larger groups.

8. REFERENCES

Brackman, 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. *WiPSCE '17: Proceedings of the 12th workshop on primary and secondary computer education* (pp. 65-71). Association for Computing Machinery. https://dl.acm.org/doi/10.1145/3137065.3137069

Organisation for Economic Co-operation and Development. (2008). *21st century learning: Research, innovation and policy: Directions from recent OECD analyses.*

Sigelman, C. K., & Rider, E. A. (2012). *Life-span human development* (7th ed.). Belmont: Wadsworth.

Paas, F., & Van Merriënboer, J. J. G. (2020). Cognitive-load theory: Methods to manage working memory load in the learning of complex tasks. *Current directions in Psychological Science, 29*(4), 394-398. https://doi.org/10.1177/0963721420922183

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*(4), 482-498. https://doi.org/10.1007/s10956-020-09831-x

Rose, S. P., Habgood, J. M. P., & Jay, T. (2017). An exploration of the role of visual programming tools in the development of young children's computational thinking. *Electronic Journal of e-Learning, 15*(4), 297-309. https://doi.org/10.34190/ejel.15.4.2368

Wing, J. M. (2006). Computational thinking. *Communications of the ACM*, 49(3), 33-35. https://doi.org/10.1145/1118178.111825