British Journal of Educational Technology *Vol 00 No 00 2016* 00–00
doi:10.1111/bjet.12496

Relation between Alice software and programming learning: a systematic review of the literature and meta-analysis

Joana M. Costa and Guilhermina L. Miranda

Joana Martinho Costa has a Computer Education master and is now doing a PhD in ICT and Education at the Institute of Education, University of Lisbon. Guilhermina Lobato Miranda is a psychologist and assistant professor at the Institute of Education, University of Lisbon. She teaches and researches in the domains of educational technology and instructional psychology. Address for correspondence: Joana Martinho Costa, Institute of Education, University of Lisbon, Alameda da Universidade, 1649-013 Lisboa, Portugal. Email: joana.martinho.costa@campus.ul.pt

Abstract

This paper presents the results of a systematic review of the literature, including a meta-analysis, about the effectiveness of the use of Alice software in programming learning when compared to the use of a conventional programming language. Our research included studies published between the years 2000 and 2014 in the main databases. We gathered 232 papers. Taking into account the selection criteria to make the meta-analysis, we retained six papers with a quasi-experimental design, with 464 participants in total. To combine the results we used the random effect model. It resulted in an effect size of 0.54 (Cohen's *d*) with a confidence interval between 0.34 and 0.74.

We concluded that until now there have been few experimental results on the effectiveness of Alice programming language to introduce students in learning how to program. The results we found were the expression of different experimental treatments and distinguished teaching methods which made the comparison of the results obtained more subtle. However, the existing experimental results that were submitted to the meta-analysis allowed us to assume with a certain margin of safety that a teaching strategy that uses Alice should obtain more effective results than the use of a conventional programming language.

Introduction

The initial learning of programming a computer has attracted the attention of researchers mainly due to the difficulties that the students present. The specialized literature points out some explanations to these difficulties, including: (1) the complex syntax of the first programming language that the students contact with, which, usually, is more appropriated in a professional context than in the initial learning stage (Jenkins, 2002); (2) the students' lack of skills in problem solving, logical and abstract reasoning, which are needed in this area (Bierre & Phelps, 2004; Dann, Cooper & Pausch, 2000); (3) the strategy of "pure" memorization often used by the students to achieve success in other subjects but which is not sufficient in this area (Salcedo e Idrobo, 2011) since programming is not only syntax and has a dynamic behavior (Dann *et al.*, 2000; Moons & Backer, 2013); some studies even refer that (4) the teaching methods could not be the most adequate, as they tend to value abstraction (Gomes, Henriques & Mendes, 2008; Moons & Backer, 2013).

Practitioner Notes

What is already known about this topic

- The students present many difficulties in initial programming learning.
- Alice software was created to support beginners but so far there has been no combination of the results obtained.

What this paper adds

- A systematic review and meta-analysis about the impact of the Alice software in initial programming learning.
- It combines studies from 2000 to 2014 and uses achievement as dependent variable.

Implications for practice and/or policy

- A teaching strategy that uses Alice should obtain more effective results than the use of a conventional programming language.
- The results we found were the expression of different experimental treatments and backgrounds which made the comparison of the results obtained more subtle.

Several authors propose a set of solutions to overcome some of these problems and facilitate the learning process. Moons and Backer (2013) present four proposals in their research. The first approach is highlighted with the use of more adequate educational methodologies, ie, a specific order to approach the subject-matters. However, it is not yet clear which one is the best methodology.

In a second approach, the authors present as a possible solution the more frequent use of learning techniques based on the constructivist perspective, as role-playing, storytelling and workshops. Liu, Cheng and Huang (2011), and Robertson (2012) present the same opinion, underlying the simulation and games construction as having a positive effect on the development of skills and problem solving.

Moons and Backer (2013) also presented as a solution the application of programming languages more appropriated to learning, as Pascal or Python. However, this solution only solves the problem of the complex syntax of the first language, mentioned by Jenkins (2002).

None of these suggested solutions converges to the proposed solution made by Gomes *et al* (2008). They affirm that programming learning should be first acquired by the programming's logic learning and only after that by the syntax. A very similar idea was first proposed by Fay and Mayer (1994) in the context of the initial learning Logo programming language. These sort of solutions may be related with the use of a type of software environments: the micro world (Moons & Backer, 2013). The micro world concept was created by Papert (1985) and consists of a programming environment that allows objects to move through commands. In this subject, Karel the robot, Scratch and Alice stand out.

Alice's language allows the students to create videos by animating characters or video games while they learn basic programming concepts. Programs are built by dragging the instructions to the edition area, also allowing the construction of interactions between objects. According to the Alice project authors and Salcedo and Idrobo (2011), programming by dragging instructions to the edition area prevents many syntax basic mistakes, one of the common problems in the initial learning of any programming language.

Table 1: Number of papers identified by the research, inclusion and exclusion criteria

Database	*Number of papers identified in search*	*Number of papers meeting inclusion criteria*	*Exclusion criteria*			*Selected*
			A	*B*	*C*	
ACM digital library	94	15	5	4	3	2[a]
ERIC	9	3	1	-	-	2
IEEE Xplore	39	6	3	1	-	2[a]
ISI web of knowledge	79	13	3	7	-	3[a]
ScienceDirect	9	1	-	1	-	0
SpringerOpen	0	-	-	-	-	0
DOAJ	2	0	-	-	-	0
Total	232	38	12	13	3	6

[a]One study repeated.

There are several studies that try to demonstrate the effectiveness of the Alice programming language, but so far was not yet carried out any systematic literature review that aggregate and compare the different results. Another feature to point out is that most of the developed studies have small dimensions and present different scientific methods (Bishop-Clark, Courte & Howard, 2006; Garlick & Çelikel, 2010; Salcedo & Idrobo, 2011). Consequently, it hinders the understanding of the true impact of this software in programming learning. That is why we think a systematic review of the literature is needed, as well as a meta-analysis which present the effect size of the studies and combine them to generate new data. We do not intend to obtain conclusive results due to the samples' size of most of the studies, but to combine the information needed for future research.

Methods

In order to reduce subjectivity, bias and the difficulty to present voluminous quantities of information inherent in the narrative review of the literature, our methodological option relied on the systematic review and meta-analysis (Borenstein, Hedges, Higgins & Rothstein, 2009; Cooper & Hedges, 2009). The clear, systematic and precise method and the evidence synthesis skill are characteristics of this type of review that constitute added values in the reaching of the goals here proposed (Gough, Oliver & Thomas, 2012; Hedges & Pigott, 2001). Furthermore, the application of the meta-analysis to combine the results allows us to obtain a clearer perspective about the impact of each study, once the analysis is done directly from the effect size (Borenstein *et al.*, 2009; Cooper & Hedges, 2009).

We developed the systematic review according to the steps suggested by Khan, Kunz, Kleijnen and Antes (2003) and Cooper and Hedges (2009). Finally we analyze the statistical significance of the results and calculate the heterogeneity to validate the studies combination (Borenstein *et al.*, 2009).

Data sources

At this phase we conducted a research in the main databases related with Computer Science and Education (Table 1). This search was done throughout November 2014 and had as restrictions: (1) studies written in English, Spanish or Portuguese; (2) from the year 2000 until 2014 (because the Alice tool was created in 1999); and (3) each paper must present abstract and full text.

Attending to the purpose of this review, the terms used for searching (that should also be presented in the abstract) were "Alice" and "programming." In the specific case of the database ISI

Web of Knowledge, the studies were also restricted to the research area "Computer Science" or "Education—Educational Research" since "Alice" is also a concept used in the Artificial Intelligence area. We obtained 232 papers in all databases (Table 1).

Inclusion and exclusion criteria

After the results of the first phase, we selected the papers which contained the following (Table 1):

1. Use of Alice software as a programming learning tool;
2. Participants belonging to high school or above, mainly the first years of college;
3. Presentation of quantitative empirical results.

In the following phase, the selected studies were object of a deep analysis (Table 1). The ones with at least one of the following characteristics have been excluded (Khan *et al*., 2003):

1. Results derived from the participants self-evaluations;
2. Results deriving from the counting of programming constructs to ascertain learning;
3. The presented data does not reflect and does not allow the effect size calculus.

In the effect size calculus we used the Cohen's *d*. This measure, when represented by a proportion, estimates the quantity of subjects of the experimental group (EG) that is expected to exceed the medium value of the control group (CG) (Conboy, 2003).

We also analyzed the references in the official website of the Alice project (http://www.alice.org/). After the analysis of the studies from the inclusion and exclusion criteria, it was possible to obtain one more study.

Coding procedures

The codification of the studies data (Table S1) was based in the meta-analysis codification of Hedges, Shymansky and Woodworth (1989) and subsequently adapted to this work's context.

Results

The results presented in Table 2 illustrate the selected studies compilation from the codification in Table 3.

In each study the EG was the group that used the Alice tool and the CG only had contact with traditional programming languages such as C, C++ or Java. These studies have in common the evaluation of the students' achievement (success) as dependent variable and a quasi-experimental design as research method. The data collection instruments (DCI) were of two types: The Sykes (2007) and Wang, Mei, Lin, Chiu & Lin (2009) studies analyze the variable "success" from questionnaires and the remaining ones analyze it from the students' final results in the Programming I subject.

The Wang *et al* (2009) study was implemented among participants that attended high school; the remaining studies were carried out among students from the first years of college (University). The number of participants per study range from 21 to 166 and the time span of each study ranges from 2 months to 4 years.

In the procedures we noticed that only the EG from the Wang *et al* (2009) and Zhang, Liu and Pablos (2014) studies used the Alice tool and were not exposed to other programming languages. In the remaining studies, the Alice tool was used concurrently or before the traditional programming classes. That is, in Cooper, Dann and Pausch (2003), Moskal, Lurie and Cooper (2004), Sykes (2007) and Johnsgard and McDonald (2008) studies, the EG had the same programming classes as the CG but was exposed for some additional time to the Alice tool previously or simultaneously (depending on the semester). We verified that the Wang *et al* (2009) study, which also

Table 2: Characteristics of selected studies

Study	Country	Procedures	Grade	DCI	N	Time	d
Cooper, Dann and Pausch (2003)	EUA	EG had classes with Alice during the first half of the semester, concurrently or before Programming I. CG did not have classes with Alice	University	Students' final results in Programming I	21	1 year	1.104
Moskal, Lurie and Cooper (2004)	EUA	EG had classes with Alice during the first half of the semester, concurrently or before Programming I. CG did not have classes with Alice	University	Students' final results in Programming I	36	1 year	0.988
Sykes (2007)	Canada	EG and CG had traditional classes in the C language. In each class the EG had an extra half an hour more with Alice	University	Questionnaire with the combination of knowledge-based, skill-set-based and problem solving-based-programming problems	83	1 year and a half	0.664
Wang, Mei, Lin, Chiu and Lin (2009)	Thailand	EG had classes with Alice and CG had classes with C++ for 8 weeks, 100 minutes per week	High	Questionnaire with 20 multiple choice questions, having focused in one specific programming concept in each of them	166	2 months	0.358
Zhang, Liu and Pablos (2014)	China	In Programming I every student had 2 hours of theoretical classes and 2 hours of practical classes per week. In the practical classes, EG used Alice and CG used Java language	University	Students' final results in Programming I	56	6 months	0.660
Johnsgard and McDonald (2008)	EUA	The students had Programming I classes in C++. EG previously had a subject with Alice. CG didn't have classes with Alice	University	Students' final results in Programming I	106	4 years	0.486

Table 3: Effect sizes

					95% CI		
Study	*Cohen's d*	*EG*	*CG*	*ES*	*LL*	*UL*	*Weight (%)*
Cooper *et al.* (2003)	1.104	11	10	0.20492	0.207	2.067	4.9
Moskal *et al.* (2004)	0.988	12	24	0.13304	0.270	1.736	7.6
Sykes (2007)	0.664	72	11	0.10551	0.025	1.311	9.5
Wang *et al.* (2009)	0.358	81	85	0.02428	0.053	0.666	41.4
Zhang *et al.* (2014)	0.660	26	26	0.7881	0.108	1.226	12.7
Johnsgard and McDonald (2008)	0.486	37	69	0.04204	0.084	0.894	23.9
Overall	0.541	241	223	0.01005	0.344	0.737	100

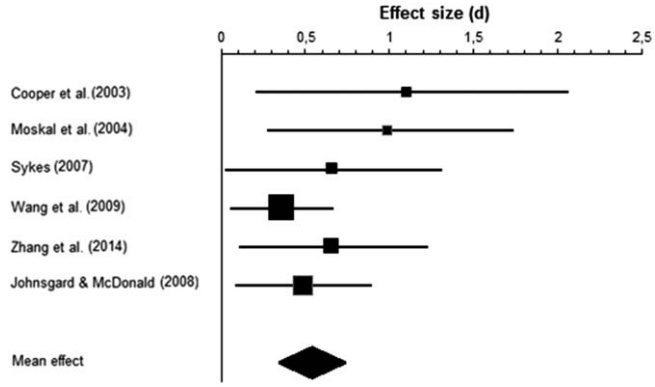

Figure 1: Forest plot with selected studies

presented the lowest effect size, is the only one that did not use the Alice tool while learning other programming language. Besides, Cooper *et al* (2003), Moskal *et al* (2004), Sykes (2007) and Johnsgard and McDonald (2008) studies used the Alice tool beyond the Programming I subject's normal schedule or previously to its beginning. From these data it is possible to conclude that in these studies the EG were exposed to programming for more time, a feature factor that may have had an impact in the obtained results.

The use of the Alice tool inside the programming curriculum and not merely as a complement should be analyzed in further research, as we did not find enough studies to understand this relation.

The weight of each study, the number of participants in the EG and CG, the obtained data from the confidence intervals at 95%, lower limit (LL) and upper limit (UL) and the standard deviation (ES) are represented in Table 3. Figure 1 represents the graphic of meta-analysis in the forest plot format.

From Table 3 it is possible to observe that the Wang *et al* (2009) study is the heavier in the calculation of the average effect size and the one that presents the lowest effect size. The Cooper *et al* (2003) study presented the highest effect size but the least significant weight, representing only about 5% of the total sample.

Although there is a large discrepancy in the size of the experimental and CGs between the Sykes (2007) and Johnsgard and McDonald (2008) studies, the total of participants present in experimental and CGs are very similar.

The weighted combination of the studies resulted in the effect size of 0.54. According to Cohen (1988) this value is considered intermediate but is not possible to categorize it accurately as it is dependent on the context in which it is inserted. Hattie (2009) in his meta-analysis works considers that this value is high in the educational area. Specifically in this case, considering that around 69% of the individuals from de EG exceed the intermediate result of the CG, we can confirm that the use of Alice shows a significant increase in the students' success in learning how to program.

From the forest plot we verified that there is a high variability in the Cooper *et al* (2003), Moskal *et al* (2004) and Sykes (2007) studies' results compared to the remaining, as the confidence interval's limits of these studies are numerically expressive. Although the minimum limit of the confidence interval of the Sykes (2007) study presents a great proximity to the value of null effect, not a single study actually crosses this value. Therefore we can conclude that each study presents results that are statistically significant.

The Wang *et al* (2009) study has the most relevant area in the graphic and therefore the biggest influence in the calculation of the weight of the final effect size.

According to Borenstein *et al* (2009), a small number of studies increases the calculus error of the variance estimate between the studies, standard error and confidence interval, which may cause false evidence. Hedges and Pigott (2011) also state that the use of studies with small samples may reduce the potentialities of the application of the random effect model used here. It is therefore beneficial to conduct a more detailed analysis of each study here presented than to focus in the calculated average. For these reasons, the number of studies derived from the systematic review does not allow us to report a conclusive effect. There is a need to deepen the use of this tool in programming learning since not all studies are clear in the type of teaching and learning strategies used in the classroom. We also believe that the differences in exposure time to learning programming is a weakness of this meta-analysis, particularly in the research developed by Johnsgard and McDonald (2008), where the "Cohort effect" may have influenced the results.

The number of participants in the six studies were 464 students (241 in EGs and 223 in CGs) which may be considered as a small sample. We present a comparative analysis between the studies with the final average of the effect size and of the calculus of heterogeneity so that the reader has a clearer perspective of the effects that Alice program language has in students' achievement. However by pointing out that, with this meta-analysis we aimed to provide an in-depth look at the effects of the Alice programming language in students learning to organize a new research and to give clues for further researchers.

Heterogeneity test

We used the random effect model as a statistic model in this meta-analysis because we considered that the effect size of each presented study can vary given the characteristics of the samples, influencing the final value of the effect size (Borenstein *et al.*, 2009). The heterogeneity was calculated from the excel file provided by Cummings (2012), from the Q and I2 test. The heterogeneity test presented the $Q = 4.81$, $df = 5$, $p = 0.44$ and $I2 = 0.0\%$ data. Therefore we can confirm that the combination of the studies is homogenous.

Conclusions and future work

We made a systematic literature review and a meta-analysis on the use of the Alice tool as a facilitator of learning to program. The need to combine the existing data in order to understand the real impact of the use of Alice in programming learning created the starting point of this paper.

Through the systematic review of the literature and the meta-analysis we verified that the use of the tool revealed a positive impact in students' achievement in the majority of the studies we analyzed and that the average of the effect size was high, considering that these studies were done in the field of education. However, the generalization of these results lacks comparative research, because we suspect that these results may be related to the teaching strategies used during the learning process of Alice programming language and also to the exposure time to other programming languages. We think that these results are not only due to the unique characteristics of this programming language. Similarly to what happened with the Logo programming language and all the huge research effort that was developed during the 1980s and the 1990s, it is possible that these results derive from the association between the language characteristics and specific teaching methods (cf. Clements, 1985; De Corte, 1993; Liao & Bright, 1991; Littlefield *et al.*, 1988; Mayer, 1988; Mendelsohn, Green & Brna, 1990; Pea & Kurland, 1984; Resnick, 1990; Valcke, 1991). From the available research results we cannot clarify this doubt because the teaching methods were not analyzed in the papers included in our systematic review of the literature and meta-analysis.

Another problem that we found in the literature review was that we came across with several studies of great potential to be included in the meta-analysis. However, the presented results did not allow us to calculate the Cohen's *d*. In some studies there is a lack of data while in others the DCIs were not adequate to ascertain learning (they used the counting of programming constructs or participants' self-assessment). Furthermore, the studies we analyzed differ considerably in the length of the treatment, a constraint that will only be overcome with a deeper experimental work.

In the future, in order to try to overcome some of the weaknesses found in the research conducted until now, we consider that researchers should use a more rigorous experimental methodology, with control and EGs, making only variations in the programming language used in each group and using valid measuring instruments. It will also be necessary to describe, with precision, the concepts and variables studied and to make a clear and parsimonious description of the adopted teaching methods. We know that this statement may generate some controversy among certain researchers who sympathize with other scientific methodologies, in particular among ones who use qualitative methods. However, we still believe that there is enough scientific room for both approaches, but we can only determine the effectiveness of a program or a teaching method using the experimental methodology (cf. Sweller, Aryes & Kalyuga, 2011).

Another relevant data in the studies analyzed consists in the use of the Alice tool as a complement of the programming curriculum and not as an alternative inside the curriculum of the Programming I subject. As future work, we intend to analyze in which way the Alice tool may be adapted to the programming curriculum in high school.

Finally, we point out the fact that the majority of the studies have been developed in a university environment, although the Alice tool has as target audience the students from high school. It will be interesting to understand if the use of this software will be more effective in high school or in university.

The studies presented in our meta-analysis only allow us to preview, with a certain margin of safety, that the Alice tool will bring positive results in the success of the students in the Programming subject. We plan to develop, during the next school year (2015/2016), an empirical research with high school students, with the experimental characteristics previously described. After that, we intend to insert the results of our empirical research in this meta-analysis and understand to what extent the Alice tool has contributed to the quality and success in programming learning in high school, specifically in the learning of basic concepts, like variables, logical and arithmetic operators, declarations, control structures and vectors. We also hope that other researchers in this field will develop experimental research that can be included in a new meta-analysis.

Statements on open data, ethics and conflict of interest

The data is available through the contact to institutional email or address.

This study was carried out under the ethical guidelines.

We declare have no conflicts of interest with respect to the research, authorship and/or publication of this work.

References

Bierre, K., & Phelps, A. (2004). The use of MUPPETS in an introductory java programming course. In Proceedings of the *Fifth Conference on Information Technology Education* (pp. 122–127). New York: ACM.

Bishop-Clark, C., Courte, J., & Howard, E. (2006). Programming in pairs with Alice to improve confidence, enjoyment, and achievement. *Journal of Educational Computing Research*, *34*(2), 213–228.

Borenstein, M., Hedges, L., Higgins, J., & Rothstein, H. (2009). *Introduction to meta-analysis*. Chichester, UK: Wiley.

Clements, D. H. (1985). *Effects of Logo programming on cognition, metacognition, skills, and achievement*. Paper presented at the Annual Meeting of the American Educational Research Association, Chicago.

Cohen, J. (1988). *Statistical power analysis for the behavioral sciences*. Hillsdale, NJ: Erlbaum.

Conboy, J. (2003). Algumas medidas típicas univariadas da magnitude do efeito. *Análise Psicológica*, *2*(21), 145–158.

Cooper, H., & Hedges, L. (2009). Research synthesis as a scientific process. In H. Cooper, L. Hedges & J. Valentine (Eds), *The handbook of research synthesis and meta-analysis* (pp. 4–15). New York: Russel Sage Foundation.

Cooper, S., Dann, W., & Pausch, R. (2003). Teaching objects-first in introductory computer science. In Proceedings of the *34th Technical Symposium on Computer Science Education* (pp. 191–195). New York: ACM.

Cummings, G. (2012). *Understanding the new statistics: effect sizes, confidence intervals, and meta-analysis*. London, UK: Routledge.

Dann, W., Cooper, S., & Pausch, R. (2000). Making the connection: programming with animated small world. In Proceedings of the *Fifth Annual Conference on Innovation and Technology in Computer Science Education* (pp. 41–44). New York: ACM.

De Corte, E. (1993). Toward embedding enriched Logo-based learning environments in the school curriculum: retrospect and prospect. In P. Georgiadis, G. Gyftodimos, Y. Kotsanis, & C. Kunigos (Eds), *Proceedings of the Fourth European Logo Conference* (pp. 335–349). Greece: University of Athens.

Fay, A., & Mayer, R. (1994). Benefits of teaching design skills before teaching Logo computer programming: evidence for syntaxe-independent learning. *Journal of Educational Computing Research*, *11*(3), 187–210.

Garlick, R., & Çelikel, E. (2010). Using Alice in CS1: a quantitative experiment. In Proceedings of the *Fifteenth Annual Conference on Innovation and Technology in Computer Science Education* (pp. 165–168). New York: ACM.

Gomes, A., Henriques, J., & Mendes, A. (2008). Uma proposta para ajudar alunos com dificuldades na aprendizagem inicial de programação de computadores. *Educação, Formação & Tecnologias*, *1*(1), 93–103.

Gough, D., Oliver, S., & Thomas, J. (2012). *An introduction to systematic reviews*. London, UK: Sage Publications.

Hattie, J. (2009). *Visible learning: a synthesis of over 800 meta-analyses relating to achievement* London, UK: Routledge.

Hedges, L., & Pigott, T. (2001). The power of statistical tests for moderators in meta-analysis. *Psychological Methods*, *4*, 426–445.

Hedges, L., Shymansky, J., & Woodworth, G. (1989). *A practical guide to modern methods of meta-analysis*. Washington, DC: National Science Teachers Association.

Jenkins, T. (2002). On the difficulty of learning to program. In Proceedings of *Third Annual Conference of the LTSN-ICS* (pp. 53–58). UK: Loughborough University.

Johnsgard, K., & McDonald, J. (2008). Using Alice in overview courses to improve success rates in programming I. In Proceedings of the *21st Conference on Software Engineering Education and Training* (pp. 129–136). Charleston, SC: IEEE.

Khan, K., Kunz, R., Kleijnen, J., & Antes, G. (2003). Five steps to conducting a systematic review. *Journal of the Royal Society of Medicine*, *96*, 118–121.

Liao, Y.-K., & Bright, G. (1991). Effects of computer programming on cognitive outcomes: a meta-analysis. *Journal of Educational Computing Research*, *7*(3), 251–268.

Littlefield, F., Delclos, V. R., Lever, S., Clayton, K. N., Bransford, J. D., & Franks J. J. (1988). Learning logo: method of teaching, transfer of general skills, and attitudes toward school and computers. In R. E. Mayer (Ed.), *Teaching and learning computer programming: multiple research perspectives* (pp. 111–135). Hillsdale, NJ: Erlbaum.

Liu, C., Cheng, Y., & Huang, C. (2011). The effect of simulation games on the learning of computational problem solving. *Computers & Education*, *57*, 1907–1918.

Mayer, R. (Ed.) (1988). *Teaching and learning computer programming: multiple research perspective*. Hillsdale, NJ: Erlbaum.

Mendelsohn, P., Green, T., & Brna, P. (1990). Programming languages in education: the search for an easy start. In J. M. Hoc, T. R. Green & D. J. Gilmore (Eds), *Psychology of programming* (pp. 175–200). London: Academic Press.

Moons, J., & Backer, C. (2013). The design and pilot evaluation of an interactive learning environment for introductory programming influenced by cognitive load theory and constructivism. *Computers & Education, 60*, 368–384.

Moskal, B., Lurie, D., & Cooper, S. (2004). Evaluating the effectiveness of a new instructional approach. In Proceedings of the *35th Technical Symposium on Computer Science Education* (pp. 75–79). New York: ACM.

Papert, S. (1985). Different visions of Logo. *Computers in the Schools: Interdisciplinary Journal of Practice, Theory, and Applied Research, 2*(2–3), 3–8.

Pea, R., & Kurland, D. M. (1984). On the cognitive effects of learning computer programming. *New Ideas in Psychology, 2*(2), 137–168.

Resnick, M. (1990). MultiLogo: a study of children and concurrent programming. In I. Harel (Ed.), *Constructionist learning* (pp. 185–216). Cambridge, MA: MIT Media Lab.

Robertson, J. (2012). Making games in the classroom: benefits and gender concerns. *Computers & Education, 59*, 385–398.

Salcedo, S., & Idrobo, A. M. (2011). New tools and methodologies for programming languages learning using the Scribbler Robot and Alice. In Proceedings of the *41st ASEE/IEEE Frontiers in Education Conference* (pp. F4G-1–F4G-6). Rapid City: IEEE.

Sweller, J., Ayres, P., & Kalyuga, S. (2011). *Cognitive load theory*. New York: Springer.

Sykes, E. R. (2007). Determining the effectiveness of the 3D Alice programming environment at the computer science I level. *Journal of Educational Computing Research, 36*(2), 223–244.

Valcke, M. (1991). Méta-analyse des recherches consacreés à Logo. In L. Gurtner et J. Retschitzki. *Logo et apprentissages* (pp. 79–91). Neuchâtel: Delachaux et Niestlé.

Wang, T., Mei, W., Lin, S., Chiu, S., & Lin, J. (2009). Teaching programming concepts to high school students with Alice. In Proceedings of the *39th ASEE/IEEE Frontiers in Education Conference* (pp. 1–6). San Antonio, TX: IEEE.

Zhang, J., Liu, L., & Pablos, P. (2014). The auxiliary role of information technology in teaching enhancing programming course using Alice. *International Journal of Engineering Education, 30*(3), 1–6.

Supporting Information

Additional supporting information may be found in the online version of this article at the publisher's website.

Table S1. Codification of the studies data