The effects of video game making within science content on student computational thinking skills and performance

Varvara Garneli and Konstantinos Chorianopoulos
Department of Informatics, Ionio Panepistemio, Kerkyra (Corfu), Greece

Received 26 November 2018
Revised 14 March 2019
Accepted 16 March 2019

Abstract

Purpose – This study aims to explore the effects of an alternative learning environment, such as the video game making (VGM) within science content, on computational thinking (CT) skills development and student performance.

Design/methodology/approach – A didactic intervention was performed for five weeks. Two student groups were taught the same computational concepts in two ways. One group was taught by constructing a video game within science content to practice science and computing curriculum while the other group constructed appropriately designed projects to practice only the computing curriculum. Additionally, the students constructed a pretest project before the beginning of the intervention and a post-test project after its end. Results were based on quantitative and qualitative code analysis and interviews from the students.

Findings – VGM within science content resulted in projects with more CT skills and also supported students to effectively apply their acquired coding skills, after the end of the intervention.

Practical implications – The results of this study suggest an interdisciplinary environment, such as the VGM within science content, which can effectively support CT skills development and computing curriculum.

Originality/value – Although VGM has been successfully applied to teach science content, this study explored the potential influence of this learning environment on CT skills development and coding fluency. Such interdisciplinary educational environments could be applied in the typical school settings to promote a plethora of skills and academic contents.

Keywords Science education, Computational thinking, Computer programming, Computing education, Video game making,

Paper type Research paper

Introduction

Video game making (VGM) for learning could be considered as an alternative educational environment which brings the educational potential of games into the typical school settings (Kafai and Burke, 2015). Computer programing, for example, has been successfully presented to novices through the VGM process (Navarrete, 2013). However,

> Encouraging programming as an activity meant to be good in itself is far removed, in its nature, from working at identifying ideas that have been disempowered and seeking ways to re-empower them (Papert, 2000).

Actually, VGM provides a powerful medium which supports student learning in various academic contents, e.g. mathematics or science by integrating the various contents within

The authors would like to thank all the students and the schoolteachers for their participation in the didactic intervention.

Interactive Technology and Smart Education

1741-5659
DOI 10.1108/ITSE-11-2018-0097

the video games structure (Kafai, 1995; Schanzer *et al.*, 2015). The potential influence of VGM for learning could be further explored, in a variety of ways. Computational thinking (CT), for example which has been defined by Wing (2008) as the "thought processes involved in formulating problems and their solutions so that the solutions are represented in a form that can be effectively carried out by an information-processing agent", has been received great attention the past years. Empirical research could examine and clearly identify the effects of VGM for learning purposes on CT skills development.

Therefore, the purpose of this empirical investigation is to explore the effects of VGM within science content on CT skills development and student performance. In particular, two middle school student groups were taught the same computational concepts using a visual programing tool, in two different ways. One group (VGM) was taught by slowly constructing a video game to learn physics and computing curriculum while the other group attended computing curriculum-focused courses (CCFC) in which theory presentations and examples were followed by appropriately designed projects, aiming at putting computing theory into practice. The above intervention lasted two school hours per week for five weeks. The experimental procedure included code analysis and interviews from the students. Therefore, this study explores VGM within science content as an interdisciplinary learning environment aiming at supporting science and computing curriculum and the main research question of this study is:

RQ1. Which are the effects of VGM for learning within science content on CT skills development and student performance?

The paper is structured as follows: In the next section, the relevant literature is reviewed; subsequently, the evaluation methodology used is presented; following this, the results are discussed; and finally, the findings are summarized.

Background work and research hypotheses

Many students dislike school activities when they emphasize on memorizing facts and acquiring skills. Papert (2000) described such activities as "a prison for a mind that wants to fly". In a constructionist perspective, computer programing could be used as an educational tool aiming at enriching the learning process, supporting exploratory learning, experimentation and creativity (Papert, 1972; Harel and Papert, 1990; Goldberg and Kay, 1977). According to Kafai and Burke (2015), activities which integrate the learning of both, coding and content, through collaborative game making could be applied into the regular curriculum. However, the most studies which used computing to teach another academic content have mostly focused on the learning of the other content (Guzdial, 2015). More research could be conducted to explore the effects of this learning environment in a variety of ways.

Video game making within science content and computational thinking skills development

CT could be considered as the ability to apply computing ideas to facilitate computing work in various disciplines and even in the daily life, and thus, CT skills should be accessible to a broad number of people (Guzdial, 2015). According to Wing (2008), CT could foster problem solving skills, using activities such as, problem decomposition, abstraction, pattern recognition, and algorithm design. Programing is an important tool for the development of CT skills but CT skills are much more than programing skills (Voogt *et al.*, 2015). There is an increasing effort to embed CT into high school science and mathematics curricular materials (Weintrop *et al.*, 2016). However, training students to acquire such skills could be a

demanding effort. Teachers believe that such approaches should start as early as the primary school years and through interdisciplinary approaches (Yılmaz *et al.*, 2018). Nevertheless, there is evidence that CT skills could be developed by constructing interactive artifacts, such as video games (Werner *et al.*, 2012; Lee *et al.*, 2011). This research is aiming to explore the effects of VGM within science content on CT skills development. Hence, we could hypothesize that:

H1. VGM within science content could support the development of projects with equal or better CT skills compared to CCFC.

Influence of video game making within science content on future computing activities

Liu *et al.* (2016) define motivation as the "force that activates, directs, and sustains goal-directed behavior". Educational interventions, such as the computing ones, need not only to motivate student interest in the respective academic subjects but furthermore sustain and develop this interest in a way that makes possible the re-engagement with similar activities and subjects (Hidi and Renninger, 2006). It really seems important for students to develop positive attitudes and interests towards CS, and to understand basic computational concepts which could be transferred to future computing experiences (Grover and Pea, 2016). As a result, the following hypothesis, regarding learners' intention to be reengaged with programing activities applying CT skills, is proposed:

H2. VGM within science content could motivate potential re-engagement with programing activities, resulting in projects with equal or better CT skills compared to CCFC.

Video game making within science content and computing curriculum

In a project-based perspective, students construct meaningful artifacts, applying knowledge and skills which represent their learning (Grant, 2002). However, learning to code could be a quite challenging task, especially for novices (Saeli *et al.*, 2011). Several programing languages have been designed closer to novices' natural ways of thinking about specifying computation (Kelleher and Pausch, 2005). Scratch, for example, is a constructionist programing tool which encourages free exploration and experimentation through an interactive interface and a set of graphical "programing blocks" which could be snapped together, resulting in directly executable programs (Resnick *et al.*, 2009). Appropriate designed tools could support students to develop a different relationship with learning, according the constructionist principles (Papert and Harel, 1991). Moreover, computing curriculum could be introduced not sequentially but as needed (Meerbaum-Salant *et al.*, 2013). In this viewpoint, students could practice the various computational concepts by making several projects, or alternatively, they could work in a project framework, creating one project (Richards, 2009), e.g. an educational video game. Such educational environments follow the constructionist perspective; learners come to respond in educational tasks which have been designed for certain pedagogical purposes discovering their knowledge through exploration and experimentation, and using mediating tools, e.g. video games (Wu and Wang, 2012). Moreover, scaffolding and peer to peer collaboration support student learning according to different needs and backgrounds (Hmelo-Silver *et al.*, 2007). More research could empirically explore the effects of diverse educational environments on student performance. Hence, we hypothesize that:

H3. VGM designed to promote science learning could support students in making functionable without coding errors projects in a potential re-engagement with computing activities compared to CCFC.

Method

Setting

In this study, the effects of a VGM approach within science content on CT skills development and learning performance were explored. This teaching intervention was designed to encourage student learning through the construction of meaningful artifacts and had a focus on project outcomes rather than test results, according to the constructionist perspective (Ackermann, 2001). In particular, a between groups experiment was designed. Two student groups were taught the same computing curriculum within two different learning environments. The VGM group students made a video game within physics content to learn computing and physics curriculum. The computational concepts were introduced according to the needs of the game construction process (Meerbaum-Salant *et al.*, 2013). At the same time, the CCFC group students were introduced to the same computing curriculum through computing theory presentations which were followed by the construction of appropriately designed projects, aiming at putting computing theory into practice. Therefore, this study examines the potential differences between these two learning environments; the VGM one which is aiming at teaching computing and physics content and the CCFC which focuses only in the computing curriculum.

The empirical study was conducted in the context of secondary education, at a Greek public middle school. The school is located in an urban area and may be considered typical in terms of the number of students, their reason for attending, and the school's infrastructure. The real classroom conditions, in which the intervention was conducted, could give useful information to educators and course designers.

Participants

A between-group teaching intervention was performed with 35 students, 18 boys and 17 girls. All students attended the third grade of middle school. The average age of participants was 15.06 years old (SD = 0.24) for the CCFC group and 15.00 years old (SD = 0.00) for the VGM one (Table I). According to the formal school curriculum for students of that age, participants were attending computing lessons at school, for 2 h per week. They had already been introduced to coding activities by making storytelling scratch projects, the previous year.

For the needs, of this study, students formed two groups and participated in the study, working in two different learning environments. The first group (VGM) was taught computer programing by constructing a video game within physics content. The second group (CCFC) was taught the same programing curriculum by constructing several projects aiming at putting the computing theory into practice. Both groups were instructed the same computing curriculum by the same two teachers, while one of them was also the researcher who kept notes during the intervention. Students were working in pairs, except three of

Table I. Participants of the study

Study groups	Boys	Girls	N
CCFC	13	5	18
VGM	6	11	17

them who decided to work independently. Students were divided into groups based on the alphabetical order of their names, in the same way classes are normally distributed. No other criterion was used for this distribution.

Procedure

At first, a meeting was conducted to make decisions related to the curriculum which will be taught and the projects which will be assigned to both groups. This meeting was composed of the two computing teachers who taught both student groups, the science teacher for the VGM group needs, and the researchers. Meeting members decided to use the Scratch Programing Environment [1]for both groups to teach more and in depth programing concepts (Table II). Then, students were informed that programing lessons would be provided. The VGM group students were additionally informed that they will construct a video game to learn computing and physics curriculum. Before the beginning of the intervention, students of both groups were asked to individually create a Scratch project (pretest), in one school hour.

The intervention lasted five sessions of two school hours each. All students were encouraged to decide the design and coding of their projects. Additionally, teachers supported all students during the construction of their projects, depending on their needs. The programing curriculum was the same and with the same order for both groups and involved the concepts of coordination and synchronization, the loops and pen commands, the conditionals, the variables, the event handlers, the operators and the random values. Students practiced these programing concepts through the construction of their projects. As a result the VGM group constructed an educational video game within physics content while the CCFC group constructed a small storytelling with two screens, geometrical shapes, a voting simulation for elections, a traffic simulation, and a small video game in which the player collects randomly appeared objects to increase his score (Table II).

After the end of the intervention, students were asked to create individually another Scratch project (post-test), in one school hour. The pre and post projects were used to explore potential differences on knowledge and skills which were applied by the students on their projects.

Programing curriculum	VGM group project	CCFC group projects
	Pre scratch project	
Coordination and Synchronization (e.g. Broadcast, When I receive, Wait)	Construction of two screens: an introductory and the main screen of the project	Construction of two screens to digitally describe a story
Loops and pen commands for designing	Electric circuit design with pen commands	Geometrical shapes design with pen commands
Conditionals, variables, and event handlers	Battery and switch to be turned on/off	Elections' simulation based on the voter's age
Operators for numerical (and Boolean) Values	Lamp, electrons and ions controlled by the logic value of battery and switch	Traffic simulation based on the logic value of traffic signals
Random values	Game features: score, avatar, and game-play Project completion	Randomly appeared figures which should be collected by the user
	Post scratch project	

Table II. Computing curricula and projects' description/group

Measures

This study results were based on student projects which were constructed during the didactic intervention. Students had to make several decisions, implementing various computational concepts and practices, while working on their projects. The collaborative interaction among them resulted in projects (Stahl *et al.*, 2006) which were used to capture student knowledge and skills (Bell, 2010). Additionally, the students individually constructed a pretest project before the beginning of the intervention and a post-test project after its end. The pre and post projects were used to explore potential differences due to the different treatments. Mastery or performance could affect the students' outcomes but it might be more critical to assess those outcomes which were achieved under student relative autonomy (Deci and Ryan, 2016). Under this perspective, the pre and post test projects were those projects that students freely decided to create, in the limited time of one school hour. Student projects were quantitatively examined regarding CT skills, using "Dr Scratch[2]", an online scratch project analyze tool. This tool provides a grading system from 1 to 3 for the following CT categories: abstraction – problem decomposition, parallelism, logical thinking, synchronization, flow control, user interactivity and data representation. The grades from each CT category are summarized, resulting in the CT skills score. Researchers repeated the work manually to cross check the results. In both ways, CT skills' grading was based on Dr Scratch methodology (Moreno-León and Robles, 2015; code examples of CT skills categories and grading can be found in Appendix 2; Table AI). An example from the CT measurement could be found in the Appendix 3 (Table AII).

In addition, a qualitative examination of student pre and post projects and a semi-structured interview from the students were used to provide a more complementary picture of the study's results. Educators kept notes about student opinion regarding computer programing, the respective teaching practice, and their projects. The semi-structured interviews guide could be found in the Appendix 1.

Data analysis

In all, 35 middle-school students were involved in this intervention and were divided into two groups, the CCFC group and the VGM one. Student projects which had been constructed during the treatments were quantitatively examined. A non-parametric Mann and Whitney (1947) test was run to determine potential differences on CT score between the groups. In addition, the size effect was calculated by Eta-squared measure (η^2) (Tomczak and Tomczak, 2014). Student pre and post projects were also explored. In particular, a non-parametric Wilcoxon signed-rank test (z), which could be applied to no normally distributed data, was used to determine whether there was a significant difference between the pre and post projects (Conover, 1999). The size effect was examined by the correlation coefficient (r) (Tomczak and Tomczak, 2014). Finally, potential differences on CT scores between groups based on the pre and post projects were investigated by using the non-parametric Mann and Whitney (1947) test. Additionally, the size effect was calculated by Eta-squared measure (η^2) (Tomczak and Tomczak, 2014).

The study also gathered information from the various informal conversations with students and observations during the intervention. Conversations were conducted randomly with those students who wanted to participate. Researchers guided the conversations to probe different aspects of student motivation and learning performance throughout the treatments. Educators encouraged students to talk about their experiences. Informal handwritten notes of student answers were made by the researchers during these conversations. Moreover, an extended qualitative study of the student projects was performed. After projects were examined and interviews were conducted, all personal

information was removed from the collected data before digitizing them. Finally, an inductive content analysis was conducted to systematically identify properties, attributes, and embedded patterns (Maguire and Bevan, 2002). This analysis consisted of two phases: in the first phase, all the interesting phrases within the informal notes were underlined. The second phase included an extended discussion to code the study's results and develop the study's coding schema. In particular, this study explored coding categories, such as student opinion and perceptions about computer programing, the educational environment and the number and type of execution errors in student projects (Table III).

Results

Student projects which were constructed during the intervention were quantitatively and qualitatively examined. The VGM group students constructed their project in several steps, practicing the physics and computing curriculum (Table II). The aforementioned steps could be seen in Figure 1. At the same time, the CCFC group students were taught the same computing curriculum, following the same order by constructing various appropriately designed projects, such as storytelling with two screens, geometrical shapes design, a traffic simulation, video game etc. (Figure 1).

Moreover, students constructed two projects, the pre and post projects in the beginning and after the end of the intervention. These projects' content was decided by the students, in a relative autonomy and were also used to capture differences between treatments on CT and student performance.

Differences on student projects during the intervention. First, we quantitatively examined student projects which were constructed during the intervention for both groups. In particular, we examined the CT score for the CCFC group (M = 16.22, SD = 2.68) and the VGM one (M = 8.6, SD= 0.7). The Mann and Whitney (1947) test indicated that the CT score was greater for the VGM group (Mdn = 18.5) than the CCFC group (Mdn = 17), U = 73.5, p= 0.017 with a medium size effect $\eta^2 = 0.30$. The above results were based on the exact sampling distribution for *U* (Dineen and Blakesley, 1973) (Table IV).

Differences between the pre and post projects for each group

Computing curriculum-focused courses group. CCFC group student projects were examined regarding the CT score. In particular, we examined the CT score of the pre projects (M = 5.61, SD = 1.42) and the post ones (M = 9.5, SD = 3.22). A Wilcoxon signed-rank test (z) indicated that the median post projects CT ranks were statistically significant

Coding categories	Coding rules/Examples	
Computer programing	Student opinion concerning computer programing and their intention to be re engaged in computing activities in the future	«I like coding», «It is a hard activity" or "It is a boring one» - "I would like to learn how to make android apps or video games"
Educational environment	Whether the instructional approach met student needs and expectations as well as the easiness or the difficulty of it	"I have learned many things and now I am able to create my own projects" or "I could easily construct my project"
Number of Errors	How many errors were made in each student project	each project had execution errors >=0
Type of Errors	How many errors were made for each programing concept	e.g. errors on the use of variables, loops etc. which result in no execution

Table III.
Qualitative study coding schema

Figure 1. Representative student projects of both groups

Table IV. Differences between groups during the intervention

Measures	Group medians CCFC	VGM	U	p	η^2
CT	17	18.5	73.5	0.017*	0.30

Note: *$p < 05$

higher than the median pre projects CT ranks, Z = 3.497, p= 0.000 (Conover, 1999) with a large size effect r= 0.47 (See Table V).

In particular, the CT score was improved in 16 post projects, decreased in 1 post project and there was no change in 1 post project.

Video game making group. VGM group student projects were also examined regarding the CT score. In particular, we examined the CT score of the pre projects (M = 5.47, SD = 1.18) and the post ones (M = 8.05, SD = 4.66). A Wilcoxon signed-rank test (z) indicated that the median post projects CT ranks were statistically significant higher than the median pre project CT ranks, $z = 2.298$, $p = 0.022$ (Conover, 1999) with a medium size effect $r = 0.31$ (Table VI).

In particular, from the 17 participants of the VGM group, the CT score was increased in eight post projects, decreased in one post project and had no change in eight post projects: differences between treatments based on the pre and post projects.

Student pre and post projects were additionally examined to identify potential differences between treatments. For this purpose, we calculated the differences on the CT score for the VGM group (M = 2.76, SD = 4.35) and the CCFC group (M = 3.89, SD = 2.99). The Mann and Whitney (1947) test indicated that there was no statistically significant difference on the CT score for any group, U = 96, $p = 0.062$. The above results were based on the exact sampling distribution for U (Dineen and Blakesley, 1973).

Qualitative analysis of the study. Student pre and post projects were qualitatively examined by the researchers. The CCFC resulted in post projects with improved CT score, except two cases, but 11 post projects included blocks of primitives which could not be executed correctly. Particularly, the observed errors concerned the usage of variables (in seven projects), the usage of sensors (in three projects), the usage of loops (in two projects) and the existence of unfinished blocks (in one project) (Figure 2). These errors were observed in different types and with diverse complexity projects.

On the other hand, the VGM approach influenced students differently; only eight post projects were improved in the CT score. Moreover, CT score was the same in both phases for seven cases. It seems that students did not decide to construct post projects with advanced CT skills. However, all students except one did not make any execution errors in the post phase of the intervention.

Student opinions concerning computer programing varied similarly, in both groups. In particular, some students of both groups considered computer programing as an enjoyable activity. For example, they mentioned that they like programing "using Scratch which is an easy tool" and "it is fun to make things to move around". The creative process of

Table V. CCFC group results based on pre and post projects

	CCFC group ($N = 18$)							
Measures	Negative ranks	Positive ranks	Z	p	Post > Pre	Pre> Post	Pre = Post	r
CT	3	9.38	−3.497	0.000*	16	1	1	0.47

Note: *$p < 05$

Table VI. VGM group results based on pre and post projects

	Video game making group ($N = 17$)							
Measures	Negative ranks	Positive ranks	Z	p	Post > Pre	Pre > Post	Pre = Post	r
CT	*2.5*	*6.25*	*−2.298*	0.022*	*8*	*2*	*7*	*0.31*

Note: *$p < 05$

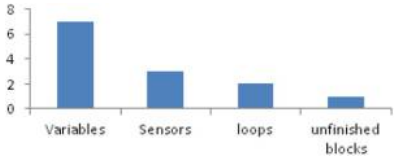

Figure 2. CCFC group errors

programing inspired many students and most of them "wanted to share their work with their classmates". At the same time, some other students had different opinion saying "that's enough with programming, I would like to make something more interesting now" or "we like programming but we'd like to use another programming tool with better graphics". Moreover, the difficulties in coding "which need so much thought in order to make things work" were underlined by many students.

The VGM approach was considered challenging and confusing, at least in the beginning. Many students needed help and advices by the teachers. Moreover, some of them were complaining for the difficulties they were dealing with. According to the students' interviews, only the 41 per cent of them considered the instructional approach easy. Despite the challenges, most programing pairs worked effectively, successfully completing the required tasks and even suggesting their own innovative solutions. At the same time, CCFC programing pairs worked more independently applying innovative ideas and giving their own perspectives. The most participants said that they could easily complete their projects (59 per cent) according to their teachers' instructions. They seemed to enjoy the intervention underlying that "we like experimentation". In general, most CCFC group students created interesting projects, using programing as a means of self-expression.

Another interesting outcome concerned student choices for future computing lessons. Although CCFC participants were more positive to coding activities, only the 50 per cent of them chose to continue in attending activities which include coding. On the contrary, VGM participants were more tired by the instructional approach but the 65 per cent of them chose a computing activity for future engagement, and mostly android apps making. Many students mentioned that they were "really interesting in coding, but they would prefer to make things similar to the ones they use in their everyday life, e.g. games and apps with better graphics".

VGM group students were doubtful about the physics content, at least in the beginning of the intervention. For example, some participants were asking "what kind of video-game we will create within physics content?" or wondering "why the physics content was chosen". A boy was really disappointing underlying that "I do not like physics and if I would decide to create another project, it will certainly be in another context". Moreover, they felt surprised with the idea of integrating game features in an electric circuit. They said that this action "will destroy their work". Despite this first reaction, the whole effort was transformed to fun and many of them seemed quite satisfied after projects' completion. "Despite the several difficulties we faced in the beginning of the intervention, we like our work and want to make more programming based projects in the future" or "we learned many things by designing this project" and some of them "wanted to upload their projects on the internet". Moreover, some students were a little disappointing saying that the project's construction was a quite hard task for them, and in some cases, the project was completed due to the efforts of their partner. An interesting observation concerned the integration of features beyond the necessary ones from students of both groups. This resulted in many projects which reflected varied interests and ideas, e.g. sports, music, social activities, etc.

Discussion

This study explored the effects of VGM within science content on CT skills and student performance compared to CCFC. In particular, one student group, applied computer programing skills for learning science and computing curriculum (VGM) by making an educational video game while the second was taught the same computational concepts by theory presentations and examples followed by appropriately designed projects aiming at putting computing theory into practice (CCFC). Both groups were taught according to the

constructionist principles and were encouraged to freely express their ideas during the intervention. Moreover, educators followed a "scaffolding" teaching strategy which builds on prior knowledge to form the new one (Van Der Stuyf, 2002). Such methods could be beneficial for each learner, as the teaching is individualized according to his/her needs and expectations, encouraging autonomous learning. The instructional approach supported both student groups to effectively participate in the teaching intervention. However, study's qualitative results revealed that VGM participants were asking more for help from their teachers, especially in the beginning. It is clear that despite the potential benefits for the students, the scaffolding strategy could become challenging for the teachers who want to meet the needs of each individual learner. Teachers need to be very well prepared due to the various challenges.

Video game making within science content and computational thinking skills

VGM within science content resulted in projects with more CT skills compared to the CCFC (*H1*). The results were statistically significant with a medium size effect. It seems that students constructed projects which reflect their learning (Bell, 2010), applying coding skills and integrating the science content within the video game structure. The significant bigger number of the CT skills score in the VGM student projects compared to the CCFC ones could be explained by the several student decisions related to the video game's interface and features, as well as the integration of the educational content within the game-play (Kafai and Burke, 2015). Therefore, VGM within science content could be considered as an alternative educational environment which promotes CT skills development.

Influence of video game making within science content on future computing activities

Student intention to be re engaged in educational activities using coding and applying CT skills was examined through student pre and post projects, as the context and complexity were decided by the students, with relative autonomy. In the post-phase of the intervention, student post projects were significantly improved in CT skills (*H2*) with no significant difference between groups. According to the quantitative and qualitative results, CCFC encouraged almost all students to construct projects with more CT skills, with a large size effect, in the post phase of the intervention. The constructionist philosophy of Scratch supported their efforts to apply their acquired knowledge and skills, promoting experimentation and free exploration (Resnick *et al.*, 2009). At the same time, the VGM group post projects were also improved on CT skills but with a medium size effect. The quantitative and qualitative analysis revealed that some post projects had equal or less CT skills in the post phase of the intervention. It seems that some students might lose some complex computing theory components due to the pressure of "solving the problem" (Richards, 2009).

Video game making within science content and computing curriculum

Student performance was examined through their ability to write code which could be executed after the end of the didactic intervention. CCFC courses resulted in projects with more execution errors in the post phase of the intervention compared to the students who followed the VGM approach (*H3*). Students who were taught under the CCFC treatment seem to experiment more with various computational concepts in their post projects, but not always effectively. This experimentation guided them to some faulty choices, unfinished work and sometimes even a messy code. Moreover, execution errors were observed in different types and with diverse complexity projects. Meerbaum-Salant *et al.* (2011) argued that the drag and drop structure of Scratch could be responsible for a tendency to extremely

fine-grained programing. Individual instructions or fragments of scripts can be left in the script area without affecting the computation of the program that is executed. At the same time, almost all students who were taught under the VGM approach used the same programing tool and constructed projects without execution errors. The VGM process which involves design and programing activities can support the learning of CS concepts (Denner *et al.*, 2012). Moreover, VGM provided students with the ability to write effective with no execution errors code and resulted in post projects which reflect the acquired knowledge and skills. According Bell (2010), project based learning supports the students' ability to self-evaluate their own projects, efforts, motivations, interests and productivity levels. In this way, students could become aware of their own strengths and make the appropriate decisions to successfully complete a project.

Despite the challenges, more students of the VGM approach choose to continue their engagement with computing activities than the students of the CCFC. In particular, students mentioned educational activities such as mobile apps development and VGM for a potential re-engagement, according to their interviews. A project framework like the VGM could enhance motivation as long term investigations of a significant question may have the potential to motivate students and promote content's understanding (Blumenfeld *et al.*, 1991).

To carry out the didactic intervention, the constructed projects themselves were examined and became an indicator of student learning. Moreover, this study results were based not only on the completed projects during the intervention but additionally on two projects which were created by the students in the beginning and in the end of the intervention, the pre and post tests. These projects were aiming at exploring those skills students chose to include in their projects. In this way, potential differences on student intention to use and further develop the acquired skills might be identified.

Most notably, qualitatively different learning environments offer different kinds of learning experiences and thus serve different learning goals (Rosen and Salomon, 2007). Despite our efforts to base our interventions in the same learning theory, we could not avoid small differences due to study's design. Thus, some results might have been influenced by such differences.

In summary, our study provides evidence regarding CT skills development and student performance of VGM within science content approach compared to CCFC. First, the generalizability of these results must be carefully considered because the field study was conducted in a specific context (e.g. content and age). Moreover, CT skills development and students performance were mostly based on projects' code analysis. However, research on the topic is limited and thus, we used more in-depth methods, such as interviews to provide a complementary picture of the findings.

The implication of this research is to suggest VGM within science content as an interdisciplinary learning environment which could effectively support CT skills development and computing learning. The clear identification of potential benefits and limitations could support the effective implementation of this approach in the typical school settings. However, this study is limited since it did not also measure student performance in the physics curriculum. Moreover, this study explored student opinion about a potential re-engagement with computing activities through an interview in the end of the intervention. We could also repeat this interview after a longer period, e.g. some weeks to crosscheck our results. Finally, further research on more parameters, e.g. student social interactions or different educational contents, could provide valuable information to educators and course designers.

Conclusions

In this study, the effects of VGM within science content on CT skills development and student performance were explored. Research's results were based on the projects' quantitative and qualitative code analysis and triangulated by more qualitative methods to provide a complementary picture. Based on these findings, some useful guidelines could be summarized.

VGM within science content could be considered as an alternative educational environment which results in projects with advanced CT skills. Potential challenges of the approach could be dealt through student collaboration and scaffolding teaching techniques. Moreover, computing learning is also supported, as students effectively applied the acquired knowledge and skills, after the end of the didactic intervention. Although some VGM students neither could nor did not want to implement advanced CT skills after the end of the intervention, almost all of them constructed projects which could be executed correctly. Finally, VGM within science content could also support potential re-engagement with computing activities. Further research is needed to explore more parameters of such interdisciplinary environments to support different educational goals and student needs.

Notes

1. Scratch.mit.edu
2. Drscratch.org

References

Ackermann, E. (2001), "Piaget's constructivism, Papert's constructionism: what's the difference", *Future of Learning Group Publication*, Vol. 5 No. 3, p. 438.

Bell, S. (2010), "Project-based learning for the 21st century: skills for the future", *The Clearing House: A Journal of Educational Strategies, Issues and Ideas*, Vol. 83 No. 2, pp. 39-43.

Blumenfeld, P.C., Soloway, E., Marx, R.W., Krajcik, J.S., Guzdial, M. and Palincsar, A. (1991), "Motivating project-based learning: sustaining the doing, supporting the learning", *Educational Psychologist*, Vol. 26 No. 3, pp. 369-398, available at: http://dx.doi.org/10.1080/00461520.1991.9653139

Conover, W.J. (1999), *Practical Nonparametric Statistics*, 3rd ed., Wiley and Sons, Hoboken.

Deci, E.L. and Ryan, R.M. (2016), "Optimizing students' motivation in the era of testing and pressure: a Self-Determination theory perspective", in Liu, W., Wang, J. and Ryan, R. (Eds), *Building Autonomous Learners*, Springer, Singapore, pp. 9-29.

Denner, J., Werner, L. and Ortiz, E. (2012), "Computer games created by middle school girls: can they be used to measure understanding of computer science concepts?", *Computers and Education*, Vol. 58 No. 1, pp. 240-249.

Dineen, L.C. and Blakesley, B.C. (1973), "Algorithm as 62: generator for the sampling distribution of the Mann-Whitney U statistic", *Applied Statistics*, Vol. 22, pp. 269-273, doi: 10.2307/2346934.

Goldberg, A. and Kay, A. (1977), "Teaching smalltalk", *Xerox PARC SSL*, Vol. 77, available at: https://pdfs.semanticscholar.org/a910/b032d17085537f41b32eb0d1fbd57ad7f282.pdf (accessed March 2017).

Grant, M.M. (2002), "Getting a grip on project-based learning: theory, cases and recommendations", *In Meridian: A Middle School Computer Technologies Journal*, Vol. 5 No. 1, p. 83, available at: https://projects.ncsu.edu/project/meridian/win2002/514/project-based.pdf (accessed August 2017).

Grover, S. and Pea, R. (2016), "Designing a blended, middle school computer science course for deeper learning: a Design-Based research approach", in Looi, C.K., Polman, J.L., Cress, U. and Reimann, P. (Eds), *Transforming Learning, Empowering Learners: The International Conference of the Learning Sciences (ICLS) 2016, Volume 1*, *International Society of the Learning Sciences*, *Singapore*.

Guzdial, M. (2015), "Learner-centered design of computing education: research on computing for everyone", *Synthesis Lectures on Human-Centered Informatics*, Vol. 8 No. 6, pp. 1-165.

Harel, I. and Papert, S. (1990), "Software design as a learning environment", *Interactive Learning Environments*, Vol. 1 No. 1, pp. 1-32.

Hidi, S. and Renninger, K.A. (2006), "The four-phase model of interest development", *Educational Psychologist*, Vol. 41 No. 2, pp. 111-127.

Hmelo-Silver, C.E., Duncan, R.G. and Chinn, C.A. (2007), "Scaffolding and achievement in problem-based and inquiry learning: a response to Kirschner, Sweller, and Clark (2006)", *Educational Psychologist*, Vol. 42 No. 2, pp. 99-107.

Kafai, Y.B. (1995), *Minds in Play: Computer Game Design as a Context for Children's Learning*, Routledge, New York, NY.

Kafai, Y.B. and Burke, Q. (2015), "Constructionist gaming: understanding the benefits of making games for learning", *Educational Psychologist*, Vol. 50 No. 4, pp. 313-334.

Kelleher, C. and Pausch, R. (2005), "Lowering the barriers to programming: a taxonomy of programming environments and languages for novice programmers", *ACM Computing Surveys*, Vol. 37 No. 2, pp. 83-137.

Lee, F., Martin, J., Denner, B., Coulter, W.A., Erickson, J. and Werner, L. (2011), "Computational thinking for youth in practice", *Acm Inroads*, Vol. 2 No. 1, pp. 32-37.

Liu, W.C., Wang, J.C.K. and Ryan, R.M. (2016), "Understanding motivation in education: theoretical and practical considerations", in Liu, W.C., Wang, J.C.K and Ryan, R.M. (Eds), *Building Autonomous Learners*, Springer Singapore, pp. 1-7.

Maguire, M. and Bevan, N. (2002), "User requirements analysis: a review of supporting methods", in Hammond, J., Gross, T. and Wesson, J. (Eds), *Ifip Wcc 2002. Ifip*, 99, Springer, Boston, pp. 133-148.

Mann, H.B. and Whitney, D.R. (1947), "On a test of whether one of two-random variables is stochastically larger than the other", *The Annals of Mathematical Statistics*, Vol. 18 No. 1, pp. 50-60, available at: www.jstor.org/stable/2236101

Meerbaum-Salant, O., Armoni, M. and Ben-Ari, M. (2011), "Habits of programming in scratch", *Proceedings of the 16th Annual joint Conference on Innovation and Technology in Computer Science Education, ACM, Darmstadt*, 168-172.

Meerbaum-Salant, O., Armoni, M. and Ben-Ari, M. (2013), "Learning computer science concepts with scratch", *In Computer Science Education*, Vol. 23 No. 3, pp. 239-264, available at: http://dx.doi.org/10.1080/08993408.2013.832022

Moreno-León, J. and Robles, G. (2015), "Analyze your scratch projects with Dr Scratch and assess your computational thinking skills", *Scratch Conference*, *Amsterdam*, pp. 12-15, available at: http://jemole.me/replication/2015scratch/InferCT.pdf (accessed August 2017).

Navarrete, C.C. (2013), "Creative thinking in digital game design and development: a case study", *Computers and Education*, Vol. 69, pp. 320-331.

Papert, S. (1972), "Teaching children thinking", *Programmed Learning and Educational Technology*, Vol. 9 No. 5, pp. 245-255.

Papert, S. (2000), "What's the big idea? Toward a pedagogy of idea power", *IBM Systems Journal*, Vol. 39 Nos 3/4, p. 720.

Papert, S. and Harel, I. (1991), "Situating constructionism", *Constructionism*, Vol. 36 No. 2, pp. 1-11.

Resnick, M., Maloney, J., Monroy-Hernández, A., Rusk, N., Eastmond, E., Brennan, K. and Kafai, Y.B. (2009), "Scratch: programming for all", *Communications of the ACM*, Vol. 52 No. 11, pp. 60-67.

Richards, D. (2009), "Designing project-based courses with a focus on group formation and assessment", *ACM Transactions on Computing Education (TOCE)*, Vol. 9 No. 1, p. 2.

Rosen, Y. and Salomon, G. (2007), "The differential learning achievements of constructivist technology-intensive learning environments as compared with traditional ones: a Meta-analysis", *Journal of Educational Computing Research*, Vol. 36 No. 1, pp. 1-14.

Saeli, M., Perrenet, J., Jochems, W.M. and Zwaneveld, B. (2011), "Teaching programming in secondary school: a pedagogical content knowledge perspective", *Informatics in Education-An International Journal*, Vol. 10 No. 1, pp. 73-88.

Schanzer, E., Fisler, K., Krishnamurthi, S. and Felleisen, M. (2015), "Transferring skills at solving word problems from computing to algebra through bootstrap", *Proceedings of the 46th ACM Technical Symposium on Computer Science Education, ACM, Indianapolis, NI*, pp. 616-621.

Stahl, G., Koschmann, T. and Suthers, D. (2006), "Computer-supported collaborative learning: an historical perspective", in Sawyer, R.K. (Ed.), *Cambridge Handbook of the Learning Sciences*, Cambridge University Press, Cambridge, pp. 409-426.

Tomczak, M. and Tomczak, E. (2014), "The need to report effect size estimates revisited: an overview of some recommended measures of effect size", *Trends in Sport Sciences*, Vol. 21 No. 1, pp. 19-25.

Van Der Stuyf, R.R. (2002), "Scaffolding as a teaching strategy", *Adolescent Learning and Development*, Vol. 52 No. 3, pp. 5-18.

Voogt, J., Fisser, P. and Good, J. (2015), "Computational thinking in compulsory education: towards an agenda for research and practice", *Education and Information Technologies*, Vol. 20 No. 4, p. 715, available at: https://doi.org/10.1007/s10639-015-9412-6

Weintrop, D., Beheshti, E. and Horn, M. (2016), "Defining computational thinking for mathematics and science classrooms", *Journal of Science Education and Technology*, Vol. 25 No. 1, pp. 127-147. available at: https://doi.org/10.1007/s10956-015-9581-5

Werner, L., Denner, J., Campe, S. and Kawamoto, D.C. (2012), "The fairy performance assessment: Measuring computational thinking in Middle school", *Proceedings of the 43rd ACM Technical Symposium on Computer Science Education, ACM*, pp. 215-220.

Wing, J.M. (2008), "Computational thinking and thinking about computing", *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences*, Vol. 366 No. 1881, pp. 3717 -3725.

Wu, B. and Wang, A.I. (2012), "A guideline for game development-based learning: a literature review", *International Journal of Computer Games Technology*, Vol. 8.

Yılmaz, F.G.K., Yılmaz, R. and Durak, H.Y. (2018), "A review on the opinions of teachers about the development of computational thinking skills in K-12", in Huseyin, O. (Ed.), *Teaching Computational Thinking in Primary Education*, IGI Global, Los Angeles, pp. 157-181.

Further reading

Guzdial, M. (1994), "Software-realized scaffolding to facilitate programming for science learning", *Interactive Learning Environments*, Vol. 4 No. 1, pp. 001-044.

Román-González, M. (2015), "Computational thinking test: design guidelines and content validation", *Proceedings of the 7th Annual International Conference on Education and New Learning Technologies (EDULEARN 2015)*, *Barcelona*, 6-8 July, pp. 2436-2444, available at: https://library.iated.org/view/ROMANGONZALEZ2015COM

Appendix 1. Semi-structured interview guide

Capturing student answers. Recording of answers will be done through taking notes. This procedure allows the interviewer to highlight key points, guide the discussion depending on different students' reactions and may make the production of the final notes and their evaluation quicker because there is no need to wade through large files of transcripts.

Develop a rapport with the respondent. Obtaining meaningful information from respondents could be easier if the interview's atmosphere is not formal. This can be done by using questions related to students' hobbies, their spare time and so on. Another significant parameter could be that the questions should lead to detailed answers and not a simple "Yes" or "No".

Examples of questions:

Q1. How do you feel about programing-based activities?

Q2. Which difficulties of such activities can you mention?

Q3. Was the instructional approach according to your needs?

Q4. Do you like the project you created?

Q5. Do you want to share your work with the others?

It is good to have a set of questions on hand, but the interviewer needs to also be prepared to expand on or probe the predetermined questions as the need arises. This is the essence of qualitative interviews.

End the interview. Deciding when to end an interview may depend on several factors. For example, interviewers may feel that they have exhausted their questions, and that they are no longer getting new information or if the respondent seems tired or has other commitments to attend to.

Finally, it is important to thank the respondent for their time and provide them with the interviewer's contact details. It is good practice for interviewers to summarize the key points that they feel the respondent has provided, because this gives the respondent a final chance to expand or clarify any points.

Appendix 2

The methodology which was used to assess the CT skills score within student projects is described below (Table AI).

CT skills grading	1	2	3
Abstraction and Problem Decomposition	More than one script and more than one sprite	Definition of blocks	Use of clones
Parallelism	Two scripts on green flag	2 scripts on key pressed, 2 scripts on sprite clicked on the same sprite	Two scripts on when I receive message, Create clone, Two scripts when %s is >%s, Two scripts on when backdrop change to
Logical Thinking	If	If else	Logic operations
Flow Control	Sequence of blocks	Repeat, forever Key pressed, Sprite Clicked	Repeat until
User Interactivity	Green flag	Ask and wait, Mouse blocks	When %s is >%s, Video, Audio
Data Representation	Modifiers of sprites properties	Operations on variables	Operations on lists
Synchronization	Wait	Broadcast, When I receive message, Stop All, Stop program, Stop programs sprite	Wait until, When backdrop change to, Broadcast and wait

Source: Moreno-León and Robles (2015)

Table AI. Computational thinking skills grading

Appendix 3. Study's quantitative factors calculation example

An example of student code could be found in Figure A1. It is a storytelling between two sprites, Boy 1 and the Boy 2.

We graded the CT categories and calculated the CT score according to Dr Scratch methodology (Moreno-León and Robles, 2015) (Table AII).

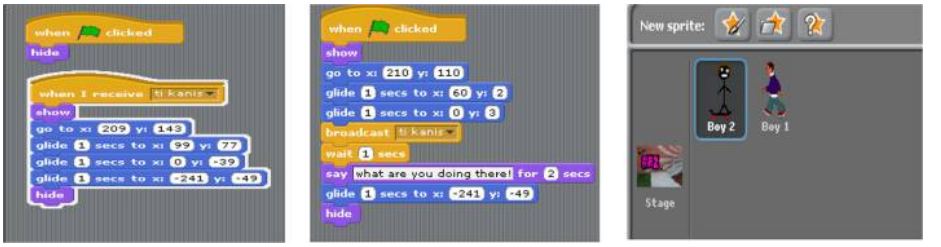

Figure A1.
Code example

CT categories	Grade 1	Grade 2	Grade 3	Category Score
Abstraction - problem decomposition	More than one script and more than one sprite	–	–	1
Parallelism	Two scripts on green flag	–	–	1
Logical thinking	–	–	–	0
Synchronization	Wait	Broadcast, When I receive message	–	2
Flow control	Sequence of blocks	–	–	1
User interactivity	Green flag	–	–	1
Data representation	Modifiers of sprites properties	–	–	1
CT skills				7

Table AII.
CT categories score calculation

Corresponding author
Varvara Garneli can be contacted at: c13garn@ionio.gr