USING VIDEO GAMES IN COMPUTER SCIENCEEDUCATION

Fan Zhang, MA
David Kaufman, EdD
Simon Fraser
Corresponding author: David Kaufman, Faculty of Education,
Simon Fraser University, , Burnaby, BC, Canada

Abstract

Recent research on the use of video games for computer science (CS) education was reviewed to identify potential benefits, summarize useful experiences, synthesize empirical evidence on the effectiveness of video game use for CS education, and identify areas for further research. The benefit most frequently-identified was increased student motivation, particular to learn programming. Implementation strategies include using games to motivate students, making games to teach CS topics, and using games as environments or examples to teach CS topics. Methodologies vary widely and results about the effectiveness of these uses are inconclusive. Further empirical research is needed to better understand the potential impact and effective education implementation of video games in computer science.

Keywords: Video games; computer science education; review; learning effectiveness; implementation strategies

Introduction

In the last decade, computer and video games have gained increasing attention as ways for students to learn key skills for today's rapidly changing world. Various authors (e.g., Clark &Sheridan, 2010; Gee, 2003; Squire, 2003) have studied the use of games in education from the perspectives of cognitive and emotional development, problem-solving, motivation, and deep thinking. Gee (2005) argues that video games are good for learning and identifies sixteen learning principles built into good games. Prensky (2005) gives examples of what children can learn about real life from games, indicating that skills obtained from games, such as problem-solving, are useful in real life and that people who play games are more successful in cognitive tasks than their non-game playing peers. Chuang and Chen's (2009) experimental study of the impact of various instructional delivery

strategies on students' learning achievement found that playing computer-based video games improves participants' fact/recall processes and promotes their problem-solving skills.

Other research, however, concludes that games don't always lead to intrinsic motivation and instructional gains. Hoffman and Nadlson (2010) used a mixed methods study to identify the factors that influence motivational engagement in gaming. Their findings suggest that it is difficult for games to achieve instructional expectations unless there is a direct relationship between the game and the learning context, and student engagement does not transfer easily to a classroom if game playing is primarily associated with entertainment. After comprehensively reviewing 105 articles on the design, use, and evaluation of games, Hays (2005) concludes that games can be used for education if they are designed to meet specific instructional objectives and are logically integrated into instructional programs. The findings of these studies call for exploration of the application of games in specific disciplines in order to build a direct relationship between the advantages of game-based learning and the achievement of specific learning objectives.

Video games have been applied to a variety of disciplines such as language, business, mathematics, and history. A number of video games have been created for the discipline of computer science (CS), which includes four core areas: theory of computation, algorithms and data structures, programming methodology and languages, and computer elements and architecture (Wikipedia, 2012). Some of these games focus on programming skills; for example, *EleMental: The Recurrence*(Chaffin, Doran, Hicks, & Barnes, 2009), is a 3D game developed and implemented to teach recursion and the depth-first search algorithm using the C# language, while *Alice*(Cooper, Dann, & Pausch, 2003) is a 3D interactive programming environment that provides students with a first exposure to object-oriented programming concepts in the context of creating simple video games. *MindRover*("Mindrover," n.d.) is a 3D strategy/ programming game in which players must use a graphic interface to equip and program a virtual robot for accomplishing a series of specific tasks. *Light-Bot* (Armor Games, 2012) is a programming-style game in which players must program their robots to visit various "goal" squares and "light" them in order to achieve the ultimate goal.

Other game-based learning tools are aimed at attracting new CS students or increasing students' motivation to study the subject. *Scalable Game Design*(Repenning, Webb, & Ioannidou, 2010), hosted at the University of Colorado, is a project to motivate and educate middle-school students to learn about CS through game design. This project aims at getting CS into public schools. The *Game2Learn* project (Game2Learn, 2012), an initiative of the Centre for Learning Innovation at the Department of

Education and Training in New South Wales, Australia, is designed to increase students' motivation and engagement in CS class. To achieve this goal, the Game2Learn group builds video games that teach introductory CS concepts and are designed for use by classroom teachers as homework tools.

The *Gaming in Computer Science* project (Microsoft External Research, 2008), initiated by Microsoft Research and academics, uses game-inspired and game development curricula in introductory CS courses to help teach fundamental programming concepts and techniques. The goal of this project is to attract talented, committed students and transform them into the next generation of computer scientists.

Serious Games in Computer Science Education (Serious Games, 2011), launched by Aalto University in 2010, is a project to develop new educational games for CS courses with the goal of understanding how these games influence students' learning.

Thus it appears that video games can be used in CS education to enhance student motivation and knowledge acquisition about CS topics. However, we need both experience and empirical evidence to support this assertion. Although current research states that some games can generate positive educational outcomes for a variety of learners in specific domains (e.g., math, electronics, and economics), this conclusion cannot be generalized to all games in all learning areas for all learners (Hays, 2005). As Gee (n.d.) has pointed out, whether video games are good for learning depends not only on their incorporation of effective learning principles, but also on the ways in which the games are used and the specific learning systems that they involve.

Therefore, *the research question addressed by this study is whether and how video games can promote student motivation and the acquisition of curricular knowledge about computer science topics.* In order to answer this question, we reviewed a set of articles concerning the use of video games as educational tools in CS context.

The purposes of this review were to:

1) identify potential benefits of the use of video games as educational tools in CS education,
2) summarize useful experiences about how to integrate gaming technology into CS education,
3) present a synthesis of the available empirical studies regarding the educational effectiveness of implementing video game technology in the context of CS education, and
4) discuss some weaknesses in the existing literature and recommend areas for future research.

Method

For the purposes of this study, ERIC and Google Scholar were used to search for literature because of their large number of stored articles on video game research. The search keywords used were: ("computer games" OR "video games" OR "digital games" OR "electronic games") AND ("computer science education" OR "CS"). Since game-based learning has been most actively studied since 2003 (Johnson, Smith, Willis, Levine, & Haywood, 2011), searches were limited to articles published in peer-reviewed journals and conference proceedings from 2003 to 2012. The references in the selected articles were also reviewed to search for additional studies about the use of video games in computer science education.

Dempsey, Rasmussen and Lucassen (1996) present a categorization schema in their review on the general instructional gaming literature; the schema includes five categories: research, theory, review, discussion and development. Papastergious (2009b) cites this schema in a review exploring the potential of computer and video games for health and physical education, but the author adds some specific criteria to each category. Using the categorization schema as refined by Papastergious, articles were included in this review if they could be placed in one of the following five categories:

(a) Research: articles comprising empirical research related to gaming in CS;

(b) Theory: articles explaining basic concepts or aspects of derived outcomes of gaming related to CS;

(c) Review: syntheses of articles on general or specific areas of gaming in CS education;

(d) Discussion: articles describing experiences and opinions regarding gaming in CS with no empirical evidence;

(e) Development: articles discussing the design or development of games or projects about gaming in CS.

Initially, 53 articles were identified. However, careful inspection of these articles showed that some did not meet the inclusion criteria. For example, six articles were excluded because the focus of these studies was not the use of video games as educational tools to teach CS topics, but rather on game design as a CS curriculum topic or part of a gaming technology degree. In the end, thirty articles were selected for the review; six were assigned to the "Research" category, one to the "Theory" category, fourteen to "Discussion," and nine to "Development." There was no article in the "Review" category, which confirms the uniqueness of this study. The next section discusses the articles in each category.

Review results

Research category

The six articles in this category focus on the educational effectiveness of integrating video games into CS education. All of these articles are based on quantitative methods or a mixture of quantitative and qualitative analysis; they can be divided into two application approaches: playing games and making games.

Three articles concern students playing games and focus on the potential benefits and educational effectiveness of using games in CS education. The first (Papastergious, 2009a) assesses the learning effectiveness and motivational appeal of the use of a game in learning computer memory concepts. In this study, 88 students including 46 boys and 42 girls aged 16-17 years were randomly selected from two high schools located in Trikala, a typical town in central Greece, and were randomly assigned to two groups: Group A and Group B. Group A used a game application, while Group B used an educational website application. A Computer Memory Knowledge Test was employed as the pre-test and post-test, and students were observed during the interventions. After the interventions, students' views on various aspects of the application were collected through a feedback questionnaire. The results showed that the game approach was more effective and more motivationally appealing for students than the website approach.

In the second article, Wang and Chen (2010) describe using an experiential gaming activity to investigate the relationships between game strategies and learners' preference on novice learners' flow experience, learning motivation and programming performance. One hundred fifteen participants were categorized into a "matching-challenging" group and a "challenging" group. During the gameplay phase, the matching-challenging group first received a matching game which was employed for concept clarification and then played a challenging game to help with concept consolidation and elaboration. The challenging group only played the challenging game. Participants who preferred the challenging game were identified as the preference-matched group and those who did not like the challenging game were identified as the preference-mismatched group. Findings were inconsistent with previous assertions in the literature: (1) a matching game was better able to clarify and consolidate programming concepts than purely challenging gameplay; and (2) matching learners' challenging game preference with challenging gameplay enhanced neither learners' flow experience nor project performance, while mismatching-challenging gameplay enhanced their project performance for those who did not prefer challenging games.

The third article is an account by Long (2007) of a survey to test the learning effectiveness of an IBM *Robocode* game designed to teach introductory Java programming. Using data from a randomly-selected sample of the *Robocode* community, Long concludes that *Robocode* can improve students' motivation to participate in the learning process and that students using *Robocode* can learn more effectively.

The three articles that focus on students building games generate conflicting results about the educational effectiveness of incorporating games in CS education. Wang (2011) evaluates the introduction of a game project to a software architecture course in which a robot project had been successfully used for five previous years to teach students software architecture. This study takes a mixed method approach using the robot project as a benchmark; if the game project performs at the same level as the robot project, the game project is well suited for teaching software architecture. The results showed that a game development project could successfully be integrated into a software architecture course and students were motivated by the game project, but there were no statistically significant differences in the final grades awarded to the game project students and robot project students

The research reported by Thomas, Greene and Ge (2011) is less quantitative in method and can be viewed as an observational study. It employed a technology-rich ethnography (TRE) approach to examine the use of game development in a high school computer programming class; students from an elementary school were invited to provide feedback to the high school students about their game designs. The findings showed that using games and gaming culture as the context for developing an instructional program for younger students could engage students in learning because it enabled students to view a challenging task as something that was largely fun.

The third article (Rankin, Gooch, & Gooch, 2008) examined the impact of a game design assignment on students' interests and attitudes about (1) attaining a CS degree, (2) continued development of programming skills, and (3) experience in game design. A game design assignment with a 2 ½ week completion limit was given to 56 undergraduate CS and non-CS majors. Pre-assignment and post-assignment surveys assessed participants' interest and attitudes about CS, programming and game design; the authors reported that the game design assignment negatively influenced the majority of participants from non-CS majors. Conversely, participants from CS majors were still committed to pursue a CS degree even though they showed decreased interest in programming and game design.

To summarise, the six articles in the "Research" category report on studies with widely varying methodologies. Four found that using games

enhanced student engagement in the learning process, and three demonstrated effective student learning. However, one led to decreased student interest in programming, and one showed an apparent interaction between student preference and game type.

Discussion category

Fourteen articles in the "Discussion" category examined students' experiences and opinions regarding gaming in CS education. The majority of papers report on games and related projects for courses such as Programming, Computer Science Foundations, Software Engineering, Data Structures, and Artificial Intelligence (AI); nearly 50% of the games were used in programming courses. All of these articles focus solely on the implementation of games in introductory courses and have no empirical evidence supporting the reported benefits.

The authors of these studies suggest that video games can motivate students to participate in the learning process, promote acquisition of knowledge (e.g. programming, design patterns), increase learning experience, and develop 21st–century skills such as leadership, cooperation and problem-solving. Hu (2007) used game scenarios and the revised Bloom's taxonomy to motivate students to learn programming principles, after which students had a more positive attitude to programming in general. The number of students in that study who failed their assignments dropped to zero after the introduction of games. Leutenegger and Edgington (2004) used a "Game First" approach in which students learned introductory programming concepts through practicing game programming assignments before they learned more advanced programming languages. Feldgen and Clúa (2004) conclude that game programming assignments are an effective way to motivate freshman students who have no interest in programming. Barnes and Gini (2008) present a project in which students engaged in a full day of programming to build a MMORPG (massively multiplayer online role-playing game). The feedback from students indicated that they achieved a better understanding on the importance of teamwork, the difficulty of establishing communication among teams and how to work on large-scale projects.

Some of these articles also report on implementation strategies for integrating games into CS education. Almost every article mentions using games to motivate students and increase their interest in CS topics. Some report on projects in which students learned CS topics (e.g. database structures, software architecture) through making games. Claypool and Claypool (2005) describe a model integrating game design into a software engineering course. Ryoo, Fonseca and Janzen (2008) describe a problem-

based learning curriculum to teach basic object-oriented programming concepts through game development.

Using games as an environment is another strategy to teach CS topics. For example, Jiau, Chen and Ssu (2009) report on a programming learning environment based on a framework known as SIMPLE and the strategy game *Resource Craft*, which were shown to enhance students' motivation to develop basic programming skills and practice advanced programming skills on their own initiative. Other articles describe using game examples to teach CS topics. For instance, Wadley and Sobell (2007) present an assignment in which students were asked to build an MMORPG that stored game-world and player states in a relational database. Using this game as a concrete database project, students acquired a better understanding of the benefits and pitfalls of developing systems based on multi-user access to shared data.

As in the Research category, the Discussion research produced conflicting results about the educational effectiveness of incorporating video games in CS education. Eight articles suggest that video games were found to be successful at improving students' learning motivation and programming skills (e.g., Feldgen & Clúa, 2004; Hu, 2007; Jiau et al., 2009), helping students learn effectively and increasing the enrolment in CS courses (Leutenegger & Edgington, 2004), and enhancing students' learning experience (Wallace, Russell & Markov, 2008). However, Cliburn's (2006) formal assessment of the effectiveness of computer games as introductory programming assignments reports that while student motivation increased when they had game assignments, student scores did not improve for those choosing game assignments compared to the non-game group. Sung, Shirley and Rosenberg (2007) describe a top-down approach integrating game programming into a computer graphics programming class. The results were assessed by pre/post-tests and students' self-reflection; students' opinions on the class did not show any changes even though the quality of their projects improved.

Theory category

The one article in this category provides theoretical support for the notion of scalable game design as an approach to increasing students' natural interest in CS. Based on a computational thinking pattern inventor, which is an inside-out gradual and iterative exploration of transferable computational thinking patterns, Repenning et al. (2010) argue that students can start with designing a simple game in middle school and then gradually develop complicated games exhibiting artificial intelligence by graduate school. Through this scalable game design, educators can bring CS to middle schools in a systematic way, balancing educational and motivational aspects of CS.

However, this approach has not yet been fully implemented and may face challenges, as suggested by some of the articles discussed above.

Development category

The nine articles in this category focus on the design and development of gaming projects in CS education. No article discusses the game creation process, methodology to guide the process, or the essential components that should be included in game development.

Four of the nine articles focus on web-based games, with one considering multi-player games. MacGlashan, Miner and desJardins (2010) describe the MAPLE Game Playing System, a web application in which students can design and program game playing agents using the Python programming language. This system is well suited for introductory CS classes, IT classes or game theory classes. Barnes and Gini (2008) outline a class project in which students are engaged for a full day of developing a MMORPG scheme. Connolly, Stansfield and McLellan (2006) present a range of guiding principles and a framework for an online games-based learning environment in which students can develop skills needed to understand and perform database analysis and design. Finally, *Age of Computers* uses a creative approach to a web-based multiplayer game in which students can walk around and play themselves through avatars during exercises in an introductory course on computer fundamentals (Djupdal & Natvig, 2004).

The majority of games in this category target undergraduate students. However, Schmiz, Czauderna, Klemke, & Specht (2011) report on a game-based learning scenario for people with low educational backgrounds. The initial design began by assessing the target group's game preference and game literacy; following this, a conceptual framework and basic technical architecture were developed. Connolly, Stansfield and Hainey (2007) describe a computer game for teaching software engineering concepts to both academic and training communities. The game is based on the traditional multi-client/single-server architecture, and development is guided by Participatory Design principles. Howard (2008) outlines the development and assessment of a robotic adventure game with which middle school students can learn fundamental programming concepts.

Almost every study in this category discusses gaming project benefits, such as encouraging the participation of young students (Howard, 2008), motivating students to perform better in the class and put more effort on their work (MacGlashan et al., 2009), and improving student performance (Barnes & Gini, 2008). However, there is only limited evaluation of their effectiveness, and there are some weaknesses in the studies. Djupdal and Natvig (2004) assessed the learning effectiveness of the *Age of Computers*

using feedback from the first version rather than direct observation. Distasio and Way (2007) built a game design framework for use in any CS course; the preliminary evaluation of this framework was conducted with just five participants and was based on self-report instruments such as source code, executable game and programmers' manual. The authors felt that their results were promising but that larger scale studies were needed. Barnes and Gini (2008) concluded that their text-based MMORPG project had merit and could serve as a useful tool in the course, but this project lasted just one day.

Discussion category: Playing games vs. making games

The 30 articles in the Discussion category can be divided into two areas of focus: playing games and making games. Specifically, nine articles consider learning through playing games compared to 21 on learning by making games. The projects concerning playing games were mainly used in courses like AI, Computer Science Foundations, Database Structures and Software Engineering, while 15 of the 21 projects on making games were used in introductory programming courses. The prominence of using video games to introduce programming might reflect the common concern about learning programming mentioned by Guimaraes and Murray (2008) and shared by other researchers that "a disconnect often exists between student perceptions of computer programming and the reality behind what it takes to build programming skill" (p. 147). Game-based courses help students gain a new perspective about programming through active engagement.

By making games, students can connect abstract programming languages with concrete game elements that they are familiar with and passionate about. More important, the skills and techniques used in game development can be transferred to a wide-range of simulations of other systems (Claypool & Claypool, 2005). However, Sung et al. (2007), investigating the integration of game programming into a Computer Graphics (CG) class, report on a set of studies that compromised the academic integrity of an introductory CG class by focusing on game programming to the extent that students misunderstood the CG class as a game programming class.

Therefore, when integrating game development into CS classes, it is worthwhile to ensure that game projects reinforce and support class objectives. Overall, both game-playing and game-making projects can show positive effects on students' motivation and performance, and both directions should be pursued. (Kafai, 2006).

Potential benefits of integrating video games into CS education

The potential benefits examined in the thirty articles can be grouped into the following categories (Figure 1): (a) motivating students, (b)

promoting knowledge acquisition (e.g. of programming, design patterns), (c) developing 21st- century skills (e.g. leadership, collaboration, problem-solving), (d) improving performance, (e) enhancing the learning experience, and (f) developing computational thinking.

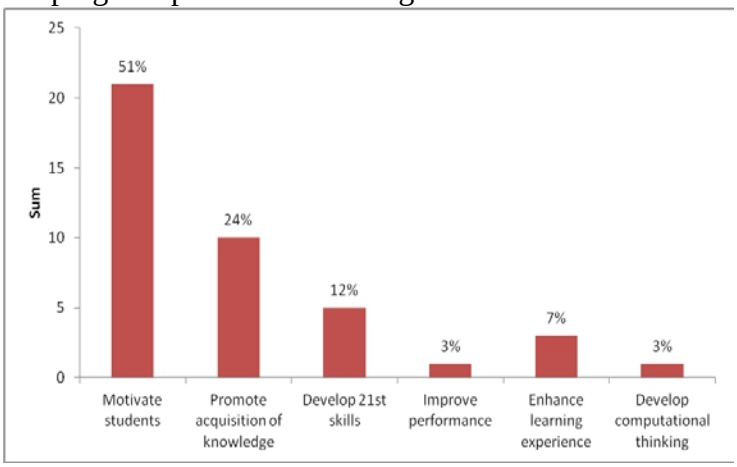

Figure 1. Distribution of potential benefits of integrating games into CS education

More than half of the thirty articles focus on using games to motivate students, followed by promoting knowledge acquisition. This is mainly an attempt to increase the enrolment in CS courses (Barnes, Richer, Powell, Chaffin, & Godwin, 2007; Leutenegger & Edgington, 2004; Rao & Mitra, 2008). Since 2000, enrolment of CS majors has been declining (Vegso, 2005). One of the main reasons is that students imagine that what CS majors learn is programming (Carter, 2006); many believe that programming is a difficult skill to acquire and have no interest in it (Feldgen & Clúa, 2004). To address this issue, many CS departments have started to use video games to recruit and retain more students by sparking their interest in CS topics and motivating their engagement in the learning process (Hu, 2007; Leutenegger & Edgington, 2004). However, the study conducted by Rankin et al. (2008) found this to be counterproductive; students from non-CS majors expressed less interest in pursuing a CS degree, taking additional game design classes and improving their programming skills, while students from CS majors showed continued commitment to pursue a CS degree, but their interest in game design and programming decreased. Since this was just one case, the study was unable to confirm or reject the relationship between playing/making games and interest in CS.

Another issue with the reported benefits for of video games in CS is that they are generally based on self-reported student perceptions, intuitions and feelings. Several studies used appropriate research designs to identify the added benefits of using video games, but they failed to provide empirical evidence to demonstrate that students actually achieve these benefits. For

example, if students state that their critical thinking has been improved by playing or making video games, appropriate empirical support such as randomized trials or pre-post testing is needed to support this statement.

Implementation strategies for integrating games into CS education

Figure 2 shows the distribution of implementation strategies reported in the articles. Most researchers used the strategy of creating games to teach CS topics. There were also some studies that used games as an environment to teach CS topics. Through interactivity and engagement with a game-based learning environment, students can start with an authentic problem and develop their own process of knowledge construction to reach a solution. For example, Connolly et al. (2006) report on a simulation game which provides students with opportunities to "learn and apply a range of relevant skills and techniques relating to database analysis and design with a more interactive, engaging and stimulating environment more akin to the real-world setting where students may find themselves in industry"(p.107).

A number of studies used games as examples to teach CS topics. The majority of these taught programming through game-oriented programming assignments. A large number of students are regular game players (McFarlane, Sparrowhawk & Heald, 2002); in order to take advantage of their fascination with games, researchers proposed programming projects which used games for examples and assignments. One evaluation of the effectiveness of computer games as programming assignments suggests that "games do indeed provide psychological motivation and increase course enjoyment; even though they may not improve students' scores" (Cliburn, 2006, p.6). However, no article provides concrete guidelines or principles for how to implement these strategies in a CS context to foster student engagement or deep learning, or in which context each strategy can work best.

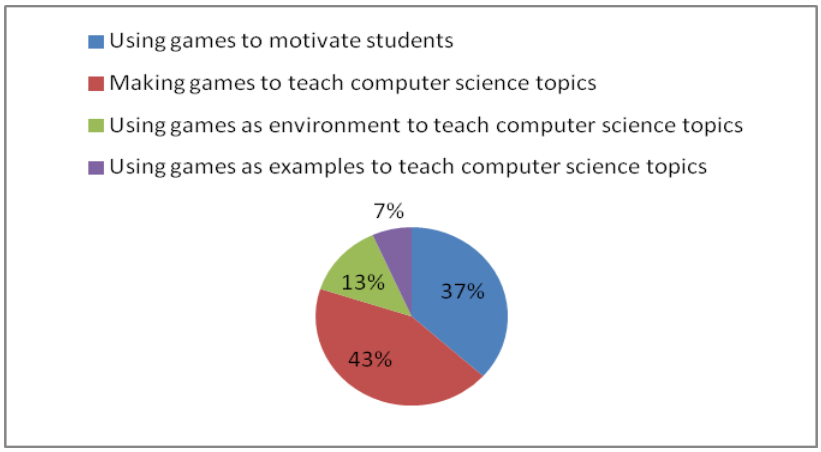

Figure 2. Distribution of implementation strategies for integrating games into CS education

Empirical studies

Table 1 provides an overview of the six empirical studies examined in this review. Although video games are said to be effective in improving students' engagement and knowledge acquisition, the empirical evidence to support this conclusion is somewhat limited, fragmentary and contradictory. Only Papastergious (2009a) and Wang (2011) compare the effectiveness of video games to other instructional media. The research findings of Rankin et al. (2008) are based on analysis of data from pre-game assignment and post-game assignment surveys. Only Wang and Chen (2010) assess the effects of the relationship between different game strategies and students' personal preference on their flow experience, learning motivation and programming performance. Two studies did no comparison of game-based learning to other forms of instruction.

Table 1 Empirical studies of the effectiveness of video games in CS education

Author(s)	Comparison Research	Evaluation
Long (2007)	No comparison	Effective
Papastergious (2009a)	Video game vs. website	Effective
Rankin, Gooch and Gooch (2008)	Pre-game assignment vs. post-game assignment	Negative effects
Thomas, Greene and Ge (2011)	No comparison	Effective
Wang (2011)	Game project vs. robot project	Improving motivation without learning gains
Wang and Chen (2010)	Matching students' reference vs. mismatching students' preference	Effective

In addition, the empirical studies had methodological limitations including use of self-report instruments, small samples, short intervention duration, and lack of random assignment to experimental or control groups. For example, Papastergious (2009a) shows that Digital Game-Based Learning (DGBL) can promote curricular knowledge and student motivation in core high school CS subjects, but the study's experiment group used the gaming approach while the control group used a website approach. This raises the question of how a control group would have performed using traditional classroom instruction rather than a website. Furthermore, the experiment lasted only two hours and compared two educational applications on computer memory concepts. Even though the game approach was effective in promoting students' knowledge of computer memory concepts, one can't conclude that games can promote curricular knowledge in other core high school CS subjects. Another issue is that the study's conclusion that DGBL can promote students' motivation was based on the observation that students of experiment group were more willing to suggest improvements and more demanding than those in control group. It is possible that the game used in the experiment group was relatively simple in

comparison to the games students play outside school, leading them to suggest more improvements than the control group students. Papastergious' suggestion that "students expect to find in the educational games that they use within school the elements that they encounter in the games that they play outside school" (p.11) strengthens this possibility.

Conclusion and future research

This review has identified potential benefits of implementing video games in the context of CS education by reviewing 30 articles published from 2003 to 2012. The most distinctive benefit identified in the articles is promoting students' motivation to participate in the learning process, especially in learning programming. Quite a lot of work has been done in the area of teaching programming by completing game-oriented programming assignments. Four implementation strategies for using video games as educational tools in CS education are: (a) using games to motivate students, (b) making games to teach CS topics, (c) using games as environments to teach CS topics, and (d) using games as examples to teach CS topics. Finally, a set of practices are presented for using games as educational tools. Even though current empirical studies present a positive picture, they show some conflicting results about the educational effectiveness of video games.

The largest group of studies (42%) focused on experiences regarding implementing video games into CS education but did not produce empirical evidence. Also, results were conflicting about the effectiveness of such an implementation. Further empirical studies are clearly needed to validate the effectiveness of these experiences. Meanwhile, more research, focusing on how to foster students' motivation, high levels of engagement and deep learning in CS topics, rather than surface similarities between video games and CS topics, is necessary in order to leverage students' interest and familiarity with video games. The number of studies (51%) that used games to increase students' interest in CS was relatively high. While this work is important, fostering deep learning rather than casual interest merits further research. Furthermore, more comparative studies that address methodological weaknesses and compare the impact of game-based instruction with other forms of instruction are needed to provide hard evidence that better support the findings of these studies.

References:

Microsoft External Research (2008). *Transforming computer science in the gaming age*. Retrieved from http://research.microsoft.com/en-us/collaboration/papers/usc_uwb_rit.pdf

Mindrover (n.d.). *Mindrover*. Retrieved from http://www.mindrover.com/mindrover.html.

Papastergious, M. (2009a). Digital game-based learning in high school computer science education: Impact on educational effectiveness and student motivation. *Computers & Education, 52*(1*),* 1-12.
Papastergious, M. (2009b). Exploring the potential of computer and video games for health and physical education: A literature review. *Computers & Education, 53*(3), 603-622.
Prensky, M. (2005). *Don't bother me Mom-I'm learning.* St. Paul, MN: Paragon House.
Rankin, Y., Gooch, A., & Gooch, B. (2008). The impact of game design on students' interest in CS. In *Proceedings of the 3rd International Conference on Game Development in Computer Science Education* (pp. 31-35). New York: ACM. doi:10.1145/1463673.1463680
Rao, T. M., & Mitra, S. (2008). *Synergizing AI and OOSE: Enhancing interest in computer science through game playing and puzzle-solving*. *In Using AI to motivate greater participation in computer science: Papers from the 2008 AAAI Spring Symposium (Technical Report SS-08-08)* (pp. 74-79). Palo Alto, CA: AAAI Press. Available at http://www.aaai.org/Press/Reports/Symposia/Spring/ss-08-08.php
Repenning, A., Webb, D., & Ioannidou, A. (2010). Scalable game design and the development of a checklist for getting computational thinking into public schools. In *Proceedings of the 41st ACM Technical Symposium on Computer Science Education* (pp. 265-269). New York: ACM. doi:10.1145/1734263.1734357
Ryoo, J., Fonseca, F., & Janzen, D. S. (2008). Teaching object-oriented software engineering through problem-based learning in the context of game design. In H. Saiedian & L. Williams (Eds.), *Proceedings of the 21st conference on Software Engineering Education and Training CSEE&T 2008* (pp. 137-144). Washington, DC: IEEE Computer Society. doi:10.1109/CSEET.2008.26
Schmitz, B., Czauderna, A., Klemke, R., & Specht, M. (2011). Game based learning for computer science education. In G. van der Veer, P. Sloep, & M. van Eekelen (Eds.), *Proceedings of CSERS '11, the Computer Science Education Research Conference* (pp. 81-86). Heerlen, The Netherlands: Open Universiteit.
Serious Games (2011). *Serious games in computer science education.* Retrieved from http://www.cse.hut.fi/en/research/LeTech/lasse-hakulinen.shtml
Squire, K. (2003). Video games in education. *International Journal of Intelligent Games and Simulation,2*(1), 49-62.
Sung, K., Shirley, P., & Rosenberg, B. R. (2007). Experiencing aspects of games programming in an introductory computer graphics class. In

Proceedings of the 38th SIGCSE Technical Symposium on Computer Science Education (pp. 249-253). New York: ACM. doi:10.1145/1227310.1227400

Thomas, M. K., Greene, B. A., & Ge, X. (2011). Fostering 21st century skill development by engaging students in authentic game design projects in a high school computer programming class. *Journal of Educational Computing Research, 44*(4), 391-408.

Vegso, Jay. (2005, May). Interest in CS as a major drops among incoming freshmen. *Computing Research News, 17*(3).

Wadley, G., & Sobell, J. (2007).Using a simple MMORPG to teach multi-user, client-server database development. In *Proceedings of the 2nd Annual Microsoft Academic Days on Game Development in Computer Science Education* (pp. 75-79).

Wallace, S. A., Russell, I., & Markov, Z. (2008). Integrating games and machine learning in the undergraduate computer science classroom. In *Proceedings of the 3rd International Conference on Game Development in Computer Science Education* (pp. 56-60) New York: ACM. doi:10.1145/1463673.1463685

Wang, A. I. (2011). Extensive evaluation of using a game project in a software architecture course. *ACM Transactions on Computing Education, 11*(1), 1-28.

Wang, L. C., & Chen, M. P. (2010). The effects of game strategy and preference-matching on flow experience and programming performance in game-based learning. *Innovations in Education and Teaching International, 47*(1), 39-52.

Wikipedia (2012). *Computer science.* Retrieved October 1, 2012 from http://en.wikipedia.org/wiki/Computer_science.