Review of Existing Video Games for Computer Science Education

Daniel Ritchie

Writer's Comment: As a Computer Science major with a minor in Education, the intersection of technology and education has long been an interest of mine. I had done some research into this for other education classes, specifically in EDU 185: Learning in the Digital Age with Dr. Darnel Degand. This topic was so interesting to me that I wanted to continue it further. I am fascinated by how programming should be taught, because it is at a convergence of mathematics, language, and engineering. In my Honors Contract class, I was able to tailor my paper to my interests. I knew that I wanted to deal with technology that teaches students how to code. This was a challenging, unique, and wonderful opportunity to explore my interests, and it inspired me to try to continue this work. Eventually, I hope to build a game that follows the guidelines in this paper.

Instructor's Comment: When Daniel asked me to serve as the advisor for his Honor's program research project, I happily accepted. His topic (teaching computer science through video games) is a research interest of mine. I met Daniel in my course EDU 185: Learning in the Digital Age, wherein students have the choice to submit podcasts, video essays, written essays, or graphic narratives for their assignments. Daniel challenged himself by producing a podcast and a video essay for his midterm and final essay, respectively. Furthermore, Daniel and two classmates collaborated and won the Shark Tank*-inspired final presentation competition that I organize for that class. The effort that led to his success in that course is visibly present in the literature review you will now read. Not only did he read numerous articles and book chapters throughout the quarter, Daniel also evaluated multiple video games. He details his assessment of two of the games from this quarter and he also discusses the criteria used to evaluate educational games (i.e. ac-*

tive learning, flow, learning objectives, game design, and accessibility).

—Darnel Degand, School of Education

Introduction

Programming is an essential skill in today's workforce. As automation and computation become increasingly important, the technical skills to manage these systems become essential. However, the United States is falling behind other countries when it comes to problem-solving and technical skills (Cummins, Millar, & Sahoo, 2019). As a result, the teaching of programming is becoming increasingly important, despite the low retention rate in computer science courses (Päivi & Lauri, 2006). Reasons for this high dropout rate include a lack of motivation, engagement, and lack of confidence (Päivi & Lauri, 2006). It is even more difficult for people whose fields lie outside of the realm of computer science to learn programming when instructors assume a fixed mindset about programming: "You either get it or you don't." Because games have been shown to increase learning and engagement with the material (Gari, Walia, & Radermacher, 2018), they are a viable way to fix this problem. However, simply using a game to teach programming does not guarantee results; games need a set of relevant criteria that can be used to measure pedagogical effectiveness.

To discuss games that teach programming, the terms "video game" and "computer science education game" need to be defined. For the first term, Jesse Schell's *The Art of Game Design* defines a game as "a problem-solving activity approached with a playful attitude" (p. 37). According to Schell, "play [is] a manipulation that indulges curiosity" (p. 30). Based on these definitions, it would be reasonable to say that video games involve problem solving, manipulation, and curiosity from the player in a virtual setting. Meanwhile, Gibson and Bell (2013) define a computer science education game as being fun/free, in a separate space/time, uncertain in outcome, governed by rules, make-believe, and educational. To synthesize these definitions, this paper will be regarding video games for computer science education as fun, open-ended, rule-governed activities that exist in a virtual fantasy setting and involve problem-solving and educational value for the player.

The goals of this paper are (1) to review, analyze, and compile existing literature for the use of games in computer science education, (2), to use that literature to construct criteria to evaluate the effectiveness of games for use in computer science education, and (3) to evaluate some existing games for computer science education based on these criteria.

Method

Research Questions

1) What are the most important game elements and design strategies that research has shown to benefit learning?
2) How well do existing video games for computer science education implement these elements and strategies?

Source Search and Selection

The Google Scholar search engine and the University of California, Davis library database were searched for articles related to video game use in computer science education. In addition, the books used in this review were found by recommendation from a faculty mentor and through an internet search. The games to review were found through an internet search for games used in computer science education. All sources were filtered based on their adherence to the definition of computer science education video games previously given. In addition, many sources had to be removed because they analyzed how creating video games can teach programming, not how video games themselves can teach programming. Results were further narrowed by excluding games and articles oriented towards children. The desired target age for articles and games in this study is high school to early college. This search resulted in eight articles and eight games.

Data Collection and Game Evaluation

The articles were parsed for the authors' design or evaluation choices and which parts of the design or evaluation correlated with higher levels of learning. Certain chapters of the books were extracted and read for use in general strategies for game design or teaching computer science. Notes were then compiled and analyzed to develop the criteria detailed below, which were used to evaluate the chosen games. The beginning levels of each game were played for the evaluation. Two of those evaluations are given below as characteristic samples for using the criteria.

Results

The criteria identified within the literature are active learning, flow, learning objectives, game design, and accessibility. Critical media perspectives and debriefing also appeared in the literature and evaluations, but they are not aspects of game design and development.

Active Learning

Active learning is learning by engaging with the material and using it in applicable contexts. Problem solving or working on a project are examples of active learning. This is opposed to passive learning, which is intake of information through lecture, video, or another medium. Sentance, Barendsen, and Schulte (2018) advocate for similar styles of learning through student-led exploration called "enquiry-based learning" or through the five E's: engage, explore, explain, extend, and evaluate (p. 101). Both of these tactics exemplify active learning because the teacher presents a hook (a problem or mystery to solve) and the students explore a solution of their own making, therefore using the material actively. Active learning in educational video games improves learning in several of the studies examined for this report. Sitzmann (2011) found that the activity level of a learning group was a moderator of the effectiveness of an educational simulation game. Gibson and Bell (2013) used active learning as a criterion in their evaluation of games for computer science education, asserting it as a necessary component of educational games. Papastergiou (2009) found that a game designed with high student control and interactivity increased the retained knowledge of the students. The existing literature supports active learning as a vital part of games seeking to teach computer science.

Flow

Flow is a term coined to refer to player immersion in a game. A game with good flow will allow the player to progress without having it feel like a chore or obligation. This is especially important for educational games because flow involves adjusting to the increased difficulty of the game, which would also apply to the incrementally increasing complexity of the material the game teaches. Appropriate progression of the learner, from a consumer to a producer of technology, is an effective way of addressing difficulties in teaching computer science (Sentance, Barendsen, & Schulte, 2018) that mirrors the idea of flow. In fact, Schell's *Lens of Flow* (2020, p. 122) bears striking resemblance to Vygostky's Zone of Proximal Development (1978), which claims people learn best when they are being challenged but not so far past their ability that the challenge is impossible. Achieving flow by matching the player's skill to the skill required to complete a goal, with a high degree of player control, allows the player to build confidence in a risk-free environment and further develop their skills (Gibson & Bell, 2013). Games designed with challenging, clear goals of progressive difficulty have the ability to increase learning in their students (Papastergiou, 2009). Gari, Walia, and Radermacher (2018) found challenges and levels were common features of games that effectively taught their players. In addition to the actual goals of the game, a good analogy or story is important for the gameplay. Analogy and storytelling are important methods of teaching computer science, and it was found that a metaphor of spells and wizardry effectively taught Java programming concepts to players of an educational game (Sentance, Barendsen, & Schulte, 2018; Esper, Foster, & Griswold, 2013). Therefore, flow is a key aspect of games that seek to teach computer science.

Learning Objectives

The learning objectives of a game not only include what the game designers have decided to teach their players but also the way in which they order those learning objectives, how well they scaffold, and how broadly they cover programming. There are numerous games that can teach a very narrow learning objective, but this is less effective when the goal is to teach the player to program. In addition, there are pedagogical choices to make for learning goals which go beyond the teaching of syntax and structure for a specific language or languages. As previously mentioned, Schell (2020) describes a game as being a primarily problem-solving endeavor; correctly oriented goals are important, so it is important to ensure that these goals are appropriate for the audience. The goals can be thought of as problems for the players to solve. The IEEE Curriculum Guide is cited as an important source for topics to teach (Papastergiou, 2009; Sentance, Barendsen, & Schulte, 2018) and can be used as a comparison for the curriculum of educational video games. In addition, Bloom's Taxonomy of Learning Objectives is a good framework for evaluating these goals by making sure that multiple levels of the taxonomy are met (Sentance, Barendsen, & Schulte, 2018). In addition, these goals address common difficulties new programmers face, such as the properties of the computer a programmer must control or the methodology that goes into writing a program (rather than memorizing a language's syntax and structure) (Sentance, Barendsen, & Schulte, 2018).

Game Design

This criterion seeks to assess how a game has been designed and if that design is effective. While many of these criteria ultimately evaluate some aspect of the game design, an effective educational video game must also be an effective video game—which this criterion is meant to ensure. The main elements of game design are present in Schell's Lens of the Elemental Tetrad (2020). This lens describes the four elements that constitute a game—mechanics, story, aesthetics, and technology—and that this review uses to evaluate the design of a game. In particular, the mechanics and story elements were observed across the games. Because mechanics are the primary function by which the players act, they have great influence over the players' learning. Points are a primary motivating factor in video games (Gari, Walia, & Radermacher, 2018), and similar mechanisms are important to motivate players and learners. Story is vital because the fantasy of a game is both part of what makes it fun and what makes it educational (Papastergiou, 2009; Malone, 1980). All elements are vital because they affect the entertainment value of the game, which is a moderator of the effectiveness of a game (Sitzmann, 2011).

Accessibility

The accessibility of a game takes into account several factors including the platform, price, and accommodations made for players of differing backgrounds. A truly accessible game would not only be simply available (low-cost and on an easily-accessible platform) but would acknowledge the players of the game are a potentially diverse group of learners and seek to be effective in complement to a player's background. In Schell's (2020) terms, this would belong to the Lens of the Player, which asserts a game designer needs to know their audience in order to make an effective game. Multiple studies have shown enduring access to a game is important for its educational potential (Sitzmann, 2011; Lawrence, 2004; Gibson & Bell, 2013). In addition, it is obvious a game without accommodations for different players would lose value for some of its players, so this must be evaluated as well.

A note on critical media perspectives: Critical media is the idea of a consumers asking themselves questions about the purpose of the media they are consuming and the tools used to accomplish that purpose (Mirra, Morrell, & Filipiak, 2017). It includes consumption, production, distribution, and invention—all with a lens of critiquing the existing social, economic, and political bodies that made the media. This is important for this review because the purpose of an educational game may not be to actually educate its players—it may be to make money. Such a purpose may detract from the educational value of a game as the designers favor qualities that garner more purchases, subscriptions, or website hits. However, this is not included as one of the criteria because any lessened educational potential from this will be evaluated by the other criteria.

A note on debriefing: Debriefing is the process of explicitly linking the experiences of a game to the curriculum of a course. Multiple studies show debriefing is beneficial to learning through a game (Sitzmann, 2011; Gibson & Bell, 2013), but given this is a review of the games themselves, it is impossible to judge on this criterion. In short, the usage of these games should accompany debriefing for maximum effectiveness, but will not be evaluated here.

Game Evaluations

The following are two examples of how games may be evaluated according to the selected criteria.

CodeCombat

CodeCombat is a browser-based role playing game (RPG) that gives the player a choice from one of four champions. The player must code in a language of their choice to get their champion to move, fight, and interact with the challenges.

Active Learning: CodeCombat fosters active learning through making the players actually code. The interface has a coding environment on the right and the champion and level on the left. The player is forced to use the language they choose to code in, and even with the fantasy flavor, they are learning by *doing*.

Flow: CodeCombat has challenges that increase in difficulty—although slowly. This makes it easier for younger audiences to tackle the challenges, but it may be slower than late high school or college students would prefer to handle. This may reduce the immersion and motivation for that demographic.

Learning Objectives: CodeCombat has great learning objectives with broad areas of programming covered including logic, functions, objects, and even web development. The game seems to cover several levels of Bloom's taxonomy, including knowledge, understanding, and application; however, it contains less analysis, synthesis, and evaluation.

Game Design: The aesthetics of CodeCombat serve the game's purpose well; everything is oriented towards the fantasy theme, and the art is simple enough to keep the animation smooth. The in-game coding environment effectively features facilitating the writing of code. The mechanics seem a little rudimentary to start, as the character can only move at the beginning, but more features and interesting mechanics come with time. Lastly, there does not appear to be anything of a recurring story; each individual level has its own tasks as a kind of story but nothing recurring.

Accessibility: CodeCombat is free to play and online, so it is easily accessible. It does not have multi-language support or closed captioning, but since it is in-browser, Google will translate for the user and screen-readers are available.

Lightbot

Lightbot is a puzzle-based mobile app where the player uses a graphic interface to give a robot instructions to move and activate ("light") certain tiles. The game presents a series of puzzles of increasing difficulty.

Active Learning: In much the same way as CodeCombat, Lightbot has the player create their own solution to the puzzles, solving the puzzle with the graphical interface. It does not involve actually typing code, nor does it include any programming language.

Flow: Lightbot has an excellent progression from easy to difficult puzzles. Playing the game is a seamless progression to new levels and difficulties, and new concepts are introduced without breaking the flow of the game.

Learning Objectives: The learning objectives of Lightbot are somewhat limited. Basic programming concepts like functions and recursion are introduced, but a wide range of programming concepts are not addressed. In addition, many of the concepts are not tied back to code.

Game Design: The aesthetics of Lightbot are well crafted and simple, and the animation is very clean. The technology is used nicely with a clean and accessible mobile interface. The mechanics are simple enough to be easy to use but allow for complex solutions to challenging puzzles. There is no story, as the game seems to be devoted entirely to its role as a puzzle game.

Accessibility: Lightbot is a paid mobile app, so it is not easily accessible for a wide range of students. It is not a game that requires much language, as it is mostly visual.

Discussion

In general, the games either focused on the puzzle aspect of programming (like Lightbot) or the RPG aspect of game-playing (like CodeCombat). While the levels of CodeCombat can be thought of as a type of puzzle, they were more oriented towards getting the player to control a character. Active learning seems to be a common aspect of the games, which makes sense because a passive game would be less interesting and less likely to motivate play. However, the games were almost all lacking a recurring storyline—one that would keep the player invested in the events of the game and their actions within it. Also, each game seemed to be made with the intent of garnering a purchase or subscription rather than a purely educational purpose.

It is certainly possible to make a game that adheres to all the criteria outlined above, and this paper calls for the creation of such a game. It would have to be created with a purely educational purpose, and it would ideally be tested for its educational potential and improved according to that testing. The most noticeable gaps this game would fill are those of a compelling story and an orientation towards high school- and college-age students learning to code for the first time. As programming and computer literacy becomes more important to today's workforce and general needs, such a game would be an important step in improving the state of computer science education.

References

Cummins, P. A., Yamashita, T., Millar, R. J., & Sahoo, S. (2019). "Problem-solving Skills of the US Workforce and Preparedness for Job Automation." *Adult Learning*, 30(3), 111-120.

Esper, S., Foster, S. R., & Griswold, W. G. (2013). "CodeSpells: Embodying the Metaphor of Wizardry for Programming." Proceedings of the 18th ACM Conference on Innovation and Technology in Computer Science Education—ITiCSE 13. https://doi.org/10.1145/2462476.2465593.

Gari, M., Walia, G, and Radermacher, A. (2018). "Gamification in Computer Science Education: A Systematic Literature Review," presented at ASEE Annual Conference and Exposition, Salt Lake City, UT, 2018. American Society of Engineering Education.

Gibson, B., & Bell, T. (2013). "Evaluation of Games for Teaching Computer Science." Proceedings of the 8th Workshop in Primary and Secondary Computing Education—WiPSE 13. https://doi.org/10.1145/2532748.2532751.

Kinnunen, P., & Malmi, L. (2006). "Why Do Students Drop Out of CS1 Courses?" Proceedings of the 2006 International Workshop on Computing Education Research—ICER 06. https://doi.org/10.1145/1151588.1151604.

Lawrence, R. (2004). "Teaching Data Structures Using Competitive Games." *IEEE Transactions on Education*, 47(4), 459–466. https://doi.org/10.1109/te.2004.825053.

Malone, T. W. (1980). "What Makes Things Fun to Learn? Heuristics for Designing Instructional Computer Games." Proceedings of the 3rd ACM SIGSMALL Symposium and the First SIGPC Symposium on Small Systems—SIGSMALL 80. https://doi.org/10.1145/800088.802839.

Mirra, N., Morrell, E., & Filipiak, D. (2017). "From Digital Consumption to Digital Invention: Toward a New Critical Theory and Practice of Multiliteracies." *Theory Into Practice*, 57(1), 12–19. https://doi.org/10.1080/00405841.2017.1390336.

Papastergiou, M. (2009). "Digital Game-based Learning in High School Computer Science Education: Impact on Educational Effectiveness and Student Motivation." *Computers & Education*, 52(1), 1–12. https://doi.org/10.1016/j.compedu.2008.06.004.

Schell, J. (2020). *The Art of Game Design: A Book of Lenses.* Boca Raton, FL: CRC Press.

Sentance, S., Barendsen, E., & Schulte, C. (2018). *Computer Science Education: Perspectives on Teaching and Learning in School*. London: Bloomsbury Academic.

Sitzmann, T. (2011). "A Meta-Analytic Examination of the Instructional Effectiveness of Computer-based Simulation Games." *Personnel Psychology*, 64(2), 489–528. https://doi.org/10.1111/j.1744-6570.2011.01190.

Vygotsky, L. S. (1978). *Mind in Society: the Development of Higher Psychological Processes*. Cambridge, MA: Harvard University Press.