

The Role of Feedback in Game2Learn

**Tiffany Barnes, Amanda Chaffin, Alex
Godwin, Eve Powell**

Department of Computer Science
University of North Carolina at Charlotte

tbarnes2@uncc.edu

Heather Richter

Department of Software and Information Systems
University of North Carolina at Charlotte

richter@uncc.edu

ABSTRACT

Games are increasingly being used for education and training in a variety of areas. We are developing a game to teach introductory computer science concepts, called Game2Learn, to increase student motivation and engagement in learning to program. As part of our early prototyping and evaluation, we are examining the role of in-game feedback on user behavior and performance. In this paper, we present the results of a study which demonstrate the importance of providing rewards and punishments that both strongly relate to the game and to the learning objectives.

Author Keywords

Education, games, introductory programming.

ACM Classification Keywords

H5.m. Information interfaces and presentation (e.g., HCI): Miscellaneous.

INTRODUCTION

Games are increasingly being recognized to have built-in motivation and familiarity for most students [2, 5, 6, 8], as well as incorporating expertise from a variety of computer science-related fields [7]. Although women are not well-represented in IT undergraduate programs, women make up forty percent of all gamers, and spend more of their game time than men playing with friends and family [1].

The goal of the Game2Learn Research Lab is to improve recruiting and retention in computer science, particularly for women and other underrepresented minorities, through immersing computing instruction in game-based learning environments. We hypothesize that teaching introductory programming using a multiplayer online role-playing game can improve student engagement, satisfaction, and skill transfer. We are employing a spiral software development cycle, alternating rapid prototypes with evaluation to investigate the effectiveness of a variety of game and interface possibilities. We have recently completed two rapid prototypes, along with initial evaluations of these early ideas.

In this paper, we explore the role of feedback of performance in such an educational game. The issue of feedback was raised in our first exploratory study, where

students questioned the seriousness of using a game to do homework assignments. These comments led us to revise our prototypes to provide more explicit rewards and punishments for right and wrong answers and perform a follow up study to further explore the issue. We report here on the effects of this added feedback on user behavior and opinions of the notion of learning to program in a game. These results demonstrate the importance of feedback in motivating the user to learn in a game, and the sometimes subtle effects on performance such rewards and punishments may have.

We first briefly discuss related work on educational games and feedback. We then introduce our prototypes and evaluations. Finally, we present the results of these evaluations and their implications on providing in-game feedback in Game2Learn and other educational games.

BACKGROUND

Games are becoming increasingly recognized for their inherent motivation, as inspiration for improving educational applications [3, 4]. Instead of merely placing learning in a game context, a careful consideration of the cycle of “user judgements, behavior, and feedback” [3] states that is important for game engagement must be woven in with the types of feedback most useful in learning. Even in educational software and intelligent tutoring systems, the choice of when, how, and how much feedback to give is a topic of heated debate, but researchers have found that feedback on a level to support students’ problem-solving steps to be effective [9]. We have therefore striven to design an educational game which balances play and learning time, makes strong ties between in-game motivation and learning outcomes, that can be used as a homework tool in a standard introductory computing course.

GAME2LEARN PROTOTYPES

The development of a full-scale massively multiplayer online role playing game (MMORPG), such as one we envision for Game2Learn, requires extensive effort and manpower. To balance the need for formative feedback to ensure success, and to accommodate smaller development teams, we have chosen to employ a highly iterative rapid prototyping development model, and to use existing game

engine technologies to provide content including art, models, and sounds. To begin the process of game development, we chose concepts from the IEEE and ACM joint curriculum for CS1, including conditionals (if-then), iteration (for and while loops), and recursion. We created two very different prototype games around these concepts to explore various game and interface possibilities.

“Saving Princess Sera” is a two-dimensional exploratory game, implemented using RPGMaker, where the player learns of the kidnapping of the princess and determines to rescue her. The user must perform various tasks involving programming concepts: correctly reordering a while loop statement of a confused old fisherman’s mind; correcting a nested for loop placing eggs in crates; and visually piecing together a quicksort algorithm. When the player makes a mistake, the character must fight a script bug, which asks the users various computer science questions in order to fight the bug.

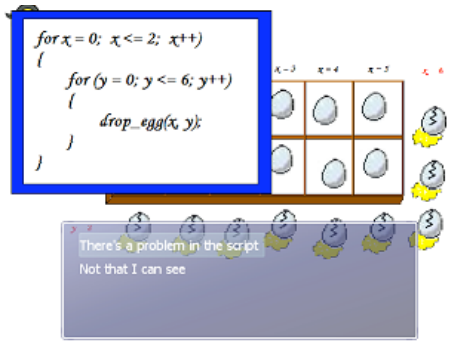


Figure 1. The egg drop quest in Saving Sera.

“The Catacombs” is a three dimensional game developed using the BioWare Aurora toolset that was used to build NeverWinter Nights, a popular fantasy role-playing game based on Dungeons and Dragons. In this game, the user is an apprentice wizard who must perform three progressively more complicated tasks to save children stuck in catacombs. The first involved unlocking the door to the catacombs involving two if statements; the second, building a bridge brick by brick with nested for loops; and the third, to solve a cryptogram using more nested for loops. In the 2nd and 3rd quests, incorrect answers result in decreasing player health.

We created two versions of this game, to explore the same tasks with two different interfaces for creating the code. One, called Grimore, uses dialogue with a sarcastic spellbook named Grimore and multiple choice questions (a dialogue tree) to create the code and complete the task. In the second, called Konijn, the player chooses the correct scroll among many incorrect ones, or uses gemstones received in the game, which represent code snippets, to fill in the blanks of chunks of code.



Figure 2. Casting a spell in The Catacombs.

INITIAL EVALUATION

To gather impressions and behavior on these prototypes, we performed an exploratory evaluation. We used students who had taken at least one introductory Computer Science course and were thus somewhat familiar with the programming concepts in the game. We gave each participant a demographic questionnaire and a pre-test of computer programming concepts. Each participant then spent approximately 20 minutes playing Saving Princess Sera and 20 minutes playing one version of The Catacombs. The participant was then given a post-test, and interviewed to gather his/her feedback about the game ideas. We asked students about their impressions of each game, and each quest within the game. We queried them about what they thought of using such a game as homework in a course, and if they would have liked such a method. We videotaped all gameplay and interviews, and created software logs of time-stamped interactions in The Catacombs.

Our initial evaluation included 13 students from a variety of backgrounds and computer experience, including 2 Asian females, 2 white females, 1 black male, 1 Hispanic male, and 7 white males. Most were ages 18-24, with 2 participants ages 25-30 and 1 aged 31-40. Nine participants are in computing related majors while the other four are studying electro-mechanical systems, psychology, communications, and art. Most participants had taken a class in either Java or C/C++. Participants included 5 sophomores, 4 juniors, 2 seniors, and 2 college grads. Five participants used the Grimore version of “The Catacombs” while eight the Konijn version.

Overall, we received relatively positive comments from participants about our prototypes, providing at least initial evidence that we can achieve our goals of providing an engaging learning environment [1]. However, we noticed an interesting pattern in how students approach the game, whether as a game or as homework, affected their comments in the interview. One student stated, “It’s something other than mindless clicking. You actually have to think, something rarely seen in games today.” This and a few other students seemed to think of our prototypes as mainly a game, and enjoyed playing such a game involving programming, but some were unsure if the game tasks

would teach them enough as part of a course. Other students thought of the prototypes as potential homework, and thought a game would make homework more fun but questioned whether the game tasks would be serious enough assignments, especially since students could guess and eventually get correct answers on the game tasks. Additionally, we noticed that a number of students sometimes did not read task instructions or other game screens very carefully, leading to errors in performance. Thus, not only did students question the potential seriousness of the game, they also did not always take the game seriously themselves, despite performing incorrectly.

These results reinforce our goal to provide in-game feedback that is closely related to the learning tasks. If this feedback is appropriate, whether students approach the tasks as a game or as homework, the feedback should be relevant and motivating to them. Yet, we realized that in these prototypes, we did not provide very clear feedback that a player's health or battles were tied to correctly performing tasks. This led us to modify our prototypes to add more explicit rewards and punishments, and perform another evaluation to see if there were any changes in the behavior or comments of students.

To "Saving Sera" we added a ranking system, where you start as a rank 7 player, and if you perform well you move toward becoming rank 1. Your rank is always visible on the screen. While playing, you can gather extra bonuses if you are of a higher rank. Non-player characters also give you more pointers on where to go next. For "The Catacombs" we only made modifications to the Grimore version of the game because of usability problems with the Konijn interface that were difficult to address. We added a "cut scene" at the beginning of the quest where the player is about to "graduate" from the wizardry academy but receives one final quest to complete: rescuing a family. During gameplay, players were penalized for incorrect answers by decreasing the player's experience points by 20% each time. If the player reached 0%, the spellbook turns the player into a chicken for a short while. At the end of the game, a cut scene was added to let the student know how they performed in-game.

FEEDBACK EVALUATION

In modifying our prototypes, we aim to investigate whether explicit feedback of game performance will lead to a difference in game behavior and opinions. Thus, we performed another round of evaluation, administered with the same format and materials as the first evaluation. We did not inform participants that we were looking more specifically at the feedback of the game. We hypothesized that the added feedback would lead to more motivation to correctly perform tasks the first time, making fewer errors in each task. We also hypothesized that users would have fewer concerns stated during the interview regarding the seriousness of the game to be used as homework in a course.

For this set of evaluations, we had 8 participants, 7 males and 1 female. Five were computer science majors, while three were in other disciplines. Again, most were aged 18-24, while one student was 25-30 and two were 31-40. Three participants were sophomores, three juniors, one senior and one graduate student.

Tables 1 and 2 summarize the averaging time spent and number of incorrect answers for each quest in The Catacombs and Saving Sera, for each set of study participants. Study one participants were our original subjects, while study two participants had the additional in-game feedback. Note that we can only compare Grimore versions of the Catacombs performance, and only five participants from the first study used this version of the game. Additionally, we did not include the final quest in the Saving Sera game in this table, as this quest had multiple parts, and most participants did not fully complete the quest, either because they died or for time reasons.

Quest:	1		2		3	
	Time	#	Time	#	Time	#
Study 1	2:33	2	2:39	2.6	2:17	1.5
Study 2	2:55	2.9	2:08	1.2	1:51	2

Table 1. The Catacombs performance results, showing the average time to complete and average number of errors for each quest. The number of errors for quest 2 are significantly different.

Quest:	1		2	
	Time	#	Time	#
Study 1	3:17	1	3:19	0.9
Study 2	3:40	1	1:59	0.9

Table 2. Saving Princess Sera performance results, showing the average time to complete and the average number of errors for each quest.

As the tables show, our results are not as clear cut as we had hoped. In only one quest, the second Catacombs quest, did we see a significant change in the number of incorrect answers. However, we also see an interesting trend, though not statistically significant, where users are completing the quests more quickly. The second and third quests in the Catacombs, and second quest in Saving Sera are each showing this trend. Thus, while for the most part, students are not answering questions more correctly, they may be working more efficiently due to their increased attention.

We saw another unexpected result in post-test versus pre-test scores amongst the two groups. In the first study, 3 participants improved their post-test score, 5 stayed the same, and 5 actually performed worse. Yet in the second study, 5 improved, 1 stayed the same, and only two

performed worse. We do not necessarily believe that actually learning occurred in such a short time period, but instead conjecture that perhaps the added motivation carried over into the post-test and led participants to try harder outside of the game as well.

Both sets of performance results imply that we may be seeing some more subtle effects of the feedback. While in-game correctness was not affected except in one task, perhaps the added motivation did lead to more engagement or attention to the tasks, leading to more efficient behavior and better performance on the post-test.

An even bigger indicator that the students were affected by the additional feedback can be seen in their interview comments. In the first study, several students commented on the seriousness of using a game as homework, and only 54% said that a game could be used to learn how to code. Yet in the second study, 88% said a game could be used for learning to program, and no one questioned the seriousness of the game. Instead, these students commented on what kind of learning was more appropriate for the game. Specifically, students expressed that the game was more appropriate for learning to modify code, but questioned whether a game could contain the complexity of tasks where one learns to create new code. Another student commented that "Its not as good as a learning tool, but good as a testing tool." And indeed, the punishment of removing experience points in The Catacombs for incorrect answers is more similar to taking a test. This feedback is reasonable given that our prototypes only had multiple choice questions and simple code rearrangement and did not involve creating code from scratch (although we feel we can scale the game appropriately in the future). Thus, these participants did seem to take the punishments and rewards seriously enough to consider the best use for this tool in the classroom, as opposed to whether the game belonged there at all.

Another interesting difference in comments was in the balance of quest (task) and other play times. In the first study, 37% of the students thought this was balanced and the other participants requested less quest time. Yet in the second study, 63% thought quest and play time was balanced, and 2 of the remaining 3 actually requested more quest time. This again implies that the students found the programming tasks to be both engaging and useful enough.

Finally, we asked the second set of participants whether or not they were motivated by the feedback we provided. Five out of eight said they were motivated by the gold and ranking in Saving Princess Sera. Fewer, only three out of 8, said they were motivated by the wizard experience points in The Catacombs, perhaps because these points were reduced for incorrect answers and this was actually demotivating. Our other results do seem to imply there was some level of attention paid to the performance feedback in both games, thus students can be motivated without their explicit awareness of the mechanisms.

CONCLUSION

The results of our two evaluations emphasize the importance of appropriate feedback, particularly in our case in how seriously the users considered the learning objectives of the game. The feedback does appear to have affected our participants comments on whether the game would be appropriate to use to learn how to program. Our performance results are more subtle in this small study, and would benefit from additional investigation. In particular, we need to examine which forms of feedback would be most appropriate and engaging to a variety of students over the long time period of an entire course. We continue to develop Game2Learn prototypes and explore various ideas for learning to program in a game environment.

ACKNOWLEDGMENTS

This work was partially supported by the CRA Distributed Mentor Project and by NSF-0552631 Computing REU Site at UNCC.

REFERENCES

1. Barnes, T., Richter, H., et al. (2007). Game2Learn: A study of games as tools for learning introductory programming. Submitted to SIGCSE2007, Kentucky, USA, Mar. 2007.
2. Becker, K. Teaching with games: The Minesweeper and Asteroids experience. *The Journal of Computing in Small Colleges* Vol. 17, No. 2, 2001, 22-32.
3. Garris, Ahlers, & Driskell. Games, motivation, and learning: a research and practice model. *Simulation & Gaming*, Vol. 33, No. 4, 2002, 441-467.
4. Gee, J. P. 2003. What video games have to teach us about learning and literacy. *Comput. Entertain.* 1, 1 (Oct. 2003), 20-20.
5. Gumhold, M. & Weber, M. Motivating CS students with game programming. In *Proceedings of the 6th International Conference on New Educational Environments (ICNEE)*, (Neuchatel, Switzerland, Sep. 27-30, 2004).
6. Jenkins III, H. Expanding and empowering the audience. *The Education Arcade: Games Literacy Workshop*, 2005. Online: <http://www.educationarcade.org/>
7. Jones, R. Design and implementation of computer games: A capstone course for undergraduate computer science education, SIGSCE 2000, (Austin, TX), New York, ACM Press, 2000, 260-264.
8. Prensky, M. *Digital Game-Based Learning*, New York, McGraw-Hill, 2001.
9. VanLehn, K., Lynch, C., Schulze, K., Shapiro, J.A., Shelby, R., Taylor, L., Treacy, D., Weinstein, A., and Wintersgill, M. (2005). The Andes Physics Tutoring System: Lessons Learned. *International Journal of Artificial Intelligence and Education*, 15 (3).

