Part of the Game: Changing Level Creation to Identify and Filter Low Quality User-Generated Levels

Andrew Hicks, Veronica Cateté, Tiffany Barnes
NC State University
890 Oval Dr.
Raleigh, NC
{aghicks3,vmcatete,tmbarnes}@ncsu.edu

ABSTRACT

While there are many potential benefits of user-generated content for serious games, the variability of that content's quality poses a serious problem. In our game, BOTS, players can create puzzles which are shared with other users. However, other players often find these puzzles irrelevant, unplayable, too difficult, or simply boring. To avoid frustrating players with low-quality puzzles, we have implemented a "Solve and Submit" process, where a player must "set the bar" for their level by providing a solution. We compare the levels that make it through this submission process to levels created when there is no such mechanism in place. We show that employing a self-evaluation mechanic for user generated content will reduce the number of low-quality puzzles submitted.

Categories and Subject Descriptors

H.5.3 [**User Interfaces**]: Evaluation/Methodology; L.5.1 [**Game-based Learning/Gaming**]

General Terms

Experimentation, Human Factors

Keywords

Game Based Tutors, Moderation, Player Engagement, Self-Evaluation, Serious Games, Social Games, User Generated Content.

1. INTRODUCTION

In this paper, we explore the potential for user-generated content (UGC) within the context of a programming puzzle game, BOTS. Content creation is an expensive part of game development, in terms of both time and expertise required. This expense is compounded when the game in question is a serious game. For intelligent tutoring systems, it is estimated that 300 hours are needed to develop 1 hour of educational content[16]. Increasingly, game developers are turning to alternate methods of content creation such as procedural generation or community authoring [14, 15]. in order to expand the body of playable content in their games without drastically increasing the expert commitment. However, since users are generally unlikely to be domain experts, the quality of the created content can vary widely. This is of particular concern for educational games, where poorly designed content could result in disengagement, or even false learning. Therefore, some method of filtering and evaluating user-authored content is required. In order for this to be scalable, this should require as little expert intervention as possible.

With this work, we examine our corpus of existing user–generated problems, identifying patterns of unwanted or low-quality UGC, and evaluate a first step towards filtration and evaluation of user-generated problems. We compared problems created under an open submission policy to those created under a "solve-and-submit" policy, where players must provide an example solution to the problem they create before they can submit it to the system. We found that the solve-and-submit policy was effective at increasing the overall quality of submitted levels, and specifically reduced the submitted quantity of certain types of unwanted UGC. We believe these results show the value of "solve-and-submit" as a first step in a content filtration process, and that based on these results, we have gained important insights on how to proceed with user- or data-driven evaluation of community-authored problems.

1.1 Filtering and Quality Assessment of User Generated Content

Intelligent tutoring systems (ITS) use adaptive strategies to teach and tailor feedback to individuals, and have been shown to be nearly as effective as one-on-one human tutoring, but building these systems takes a great deal of time and expert knowledge needed from content experts, instructional designers, and software engineers [16, 5]. Murray estimated a 300-to-1 ratio of expert hours to hours of content for Intelligent Tutoring Systems. To create effective game-based learning environments requires additional expertise and time from experts in game design, user immersion, and content creation. Additionally, problems created by educators or developers are usually presented in a meaningful sequence. Once those problems are exhausted, the experience is generally over. Prensky stresses that replayability is a major component of successful games [20], and games constructed as linear progressive learning experiences are not particu-

larly replayable. By creating games that are solitary and non-replayable, serious games developers are failing to harness several advantages of learning in games, and may not be giving the players enough time to refine learned skills outside of rigidly structured areas. In his work with meaningful gamification, Scott Nicholson states that principles from the use of Player-Generated Content can be used to improve games by allowing players to set their own content mastery goals. Through allowing users to set their own challenges, we can guide the user to create their own short- and long-term objectives [17].

While user-generated content certainly has a lot to offer for educational games, there are also several downsides to it, which can hinder or disrupt game-play. User-generated content systems must provide some form of quality control, or be overrun by hastily created, poor-quality content. Heavily UGC-based games like LittleBigPlanet and Spore [15, 14] have come up with solutions, such as limiting the number of "upload slots" an individual user has, or allowing the community co collectively vote on whether or not a creation is useless or offensive. Collaborative filtering and community based editing has been used to identify flawed or unwanted content in Wikipedia [2], product reviews (amazon.com), and other recommender systems. However, these approaches require that 'bad' content be shown to at least some users, which we would like to avoid. To accomplish that, we will need to filter user-generated content before it has been provided to users.

In the Intelligent Tutoring Systems (ITS) domain, Aleahmad showed that non-expert users can be competent content creators in domains such as high-school geometry [1], and are often able to provide more understandable problem descriptions than experts. Previous work done with user-created problems in serious games by Boyce, et al showed that including puzzle creation as another path to the game's core concept enhanced motivation for players less interested in competition [6], allowing the game to more effectively engage those players. Allowing users to create their own challenges is a very powerful motivator, according to research on design patterns for educational games identified by a team at Microsoft Research [19]. Effective educational games take advantage of the fact that "constructing things is fun and helps learning," and that "a social component (collaboration, competition) makes games fun/engaging [19]." User-generated content can be used to satisfy both of these creative and social design patterns at once.

1.2 Game Description

BOTS is a programming puzzle game designed to teach fundamental ideas of programming and problem-solving to novice computer users. The goal of the BOTS project is to investigate how to best use community-authored content within serious games and educational games. BOTS was inspired by games like LightBot [23] and RoboRally [9], as well as the success of Scratch and it's online community [8] [13]. In BOTS, players take on the role of programmers writing code to navigate a simple robot around a grid-based 3D environment, as seen in Figure 1. The goal of each puzzle is to press several switches within the environment, which can be done by placing an object or the robot on them. To program the robots, players will use simple graphical pseudo-code, allowing them to move the robot, repeat sets of commands using "for" or "while" loops, and re-use chunks

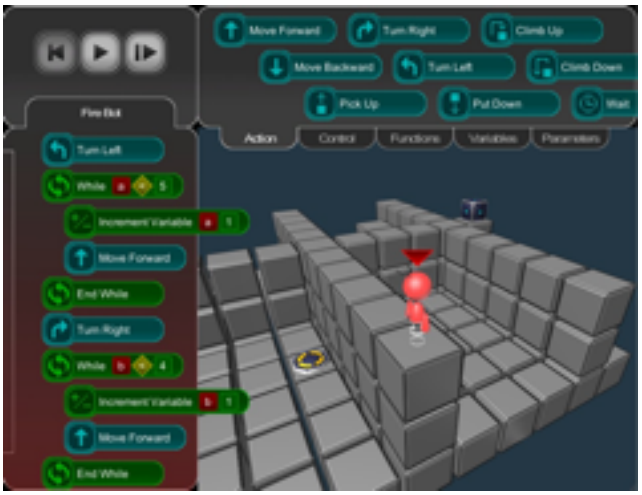

Figure 1: The BOTS interface. The robot's program is along the left side of the screen. The "toolbox" of available commands is along the top of the screen.

of code using functions. Within each puzzle, players' scores depend on the number of commands used, with lower scores being preferable. In addition, each puzzle limits the maximum number of commands, as well as the number of times each command can be used. For example, in the tutorial levels, a user may only use the "Move Forward" instruction 10 times. Therefore, if a player wants to make the robot walk down a long hallway, it will be more efficient to use a loop to repeat a single "Move Forward" instruction, rather than to simply use several "Move Forward" instructions one after the other. These constraints are meant to encourage players to re-use code and optimize their solutions.

In addition to the tutorial / "story" mode, BOTS also contains an extensive "Free Play" mode, with a wide selection of puzzles created by other players. The game, in line with the "Flow of Inspiration" principles outlined by Alexander Repenning [21], provides multiple ways for players to share knowledge through authoring and modifying content. Players are able to create their own puzzles to share with their peers, and can play and evaluate friends' puzzles, improving on past solutions. Features such as peer-authored hints for difficult puzzles, and a collaborative filtering approach to rating are planned next steps for the game's online element. We hope to create an environment where players can continually challenge their peers to find consistently better solutions for increasingly difficult problems.

User-generated content supports replayability and a sense of a community for a serious game. We believe that user-created puzzles could improve interest, encouraging students to return to the game to solidify their mastery of old skills and potentially helping them pick up new ones.

2. METHODS

There are two parts to our methodology for this work. First, we examined existing user-created puzzles, in an effort to find patterns of unwanted UGC and hopefully identify common features that would make those patterns easier to identify and remove. Second, we implemented a "solve-and-submit" mechanism into the game's level creator, and investigated what effect that had on the quality and quan-

tity of levels created by users under both the new condition and the original, open submission condition.

2.1 Elements and Patterns of User Generated Content

While the quality of a game puzzle is subjective, we developed a set of criteria necessary for a BOTS puzzle to be solvable and relevant to the game's core concepts. To develop these criteria, we were inspired by the use of design patterns in level design analysis in Hullet and Whitehead's work [10]. We examined the existing puzzles for BOTS, looking for common structures and patterns that exist across multiple puzzles. Once these structures were identified, we discussed how they could positively or negatively impact gameplay, why a puzzle creator could be motivated to create them, and how we could incentivize or discourage puzzle creators from using them. These structures were identified in puzzles created in an older version of BOTS, so the pictures that follow contain slightly different interface than is shown in Figure 1.

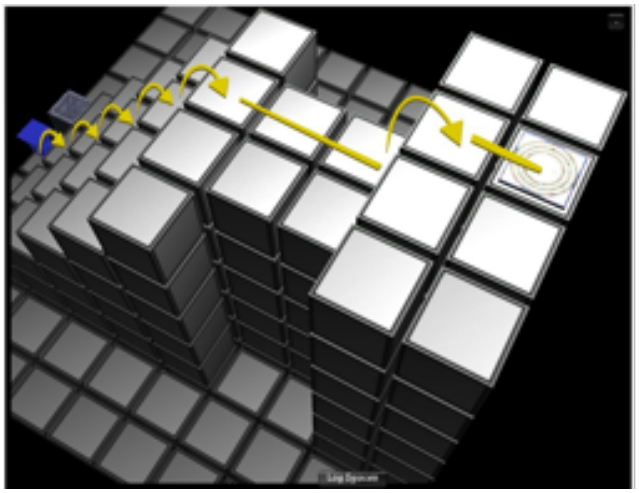

Figure 2: This puzzle has a clear trivial solution. The blue robot can climb up the steps and walk towards the target (as shown with the yellow arrows). There are also opportunities for optimization in the repeated "Jump" and "Move Forward" actions needed to move up the stairs.

The most important characteristic for user-created BOTS puzzles is the opportunity for puzzle solvers to use the target learning objectives for BOTS of efficient and reused code. To satisfy this condition, the user-created puzzle must first have a *trivial solution* that requires only simple direct commands such as Move Forward or Turn. This allows users to solve the puzzle in a straightforward way. The puzzle must also have an *advanced solution*, a higher-performance solution that uses the game's advanced concepts of iteration and functions. This ability for players to approach puzzles simply at first, and then gradually increase proficiency with more advanced concepts is a hallmark of good game play. Good games balance ease with challenge to keep players in a flow state [7] as players start as novices and increasingly become more skilled. If all puzzles have both simple and advanced solutions, then this increases replayability since players can master new skills but later revisit older puzzles to improve their performance.

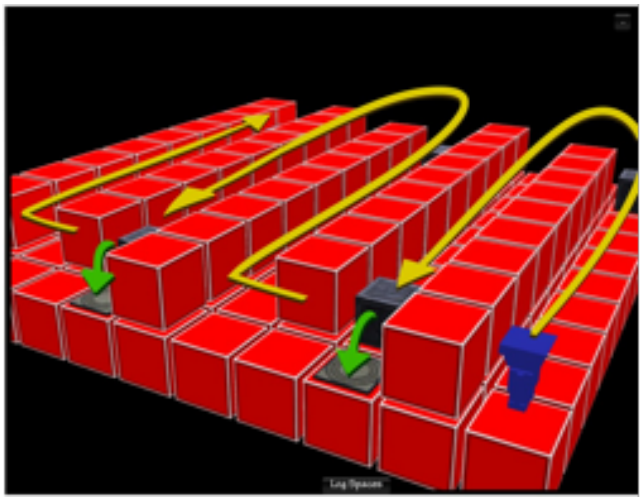

Figure 3: Puzzle with obvious structural cues for optimization of the solution. Hallways, stairs, or patterns of objects can be used to show where looping may be valuable, or where code can be reused. In this level, the first "leg" of the puzzle uses the same layout as the second "leg".

In addition to possessing both trivial and advanced solution paths, a good user-created puzzle should contain *structural cues* that help guide the player between those solutions. In his 2009 GDC talk, "Puzzles as User Interface," Randy Smith [22] stressed that in order for puzzles to challenge players without making them feel dumb, they should try to avoid "red herring" elements which are not actually relevant. He also stresses the importance of re-using familiar structures, accompanied by visual patterns which indicate where to use those known structures.

In the case of BOTS, these would be obvious *patterns in the placement of obstacles and objectives* that show the user where loops and functions would be best implemented. Elements that are wholly unnecessary can serve the opposite purpose, distracting users from the goal by presenting useless paths or unreachable objectives. A good user-created puzzle should make it apparent what the user is supposed to do using structural cues, and provide as *few unnecessary structural elements* as possible to avoid distracting the player from their goal.

Finally, a good user-generated puzzle should derive difficulty from changing the trivial solution into the advanced one, rather than from forcing the player to struggle through many repetitive actions. The advanced solutions should *not be tedious solutions*, which try the user's patience rather than test their knowledge. [18] That is, the core mechanic should be related to the core concepts of the game, rather than unrelated UI interactions. Compare this to a very long, completely featureless level in a 2D platformer. The task itself is not providing difficulty; rather the interface itself has become an obstacle. [12, 11] Though this seems obvious, it is clear when looking at the existing puzzles that users often achieve difficulty in their puzzles by forcing these kinds of repetitive actions. Using these criteria, we have identified several categories of unwanted user-created puzzles that we hope to eliminate or discourage. We have also categorized these puzzles by the type of player that might create them,

using Bartle's player types: *killers, achievers, explorers,* and *socializers* [3, 4].

We would expect *killers* to create very difficult puzzles, hoping to frustrate the progress of others in a competitive way. *Achievers* might create simple puzzles, completing them in order to get their name on the high-score list. Explorers might play with the interface as a toy, and *Socializers* might use the interface to communicate with others, riffing off of the designs of other players, or simply spelling messages with objects. Bartle also defined a subtype of the killer group known as the *griefer*. Rather than creating difficult puzzles, a griefer might create intentionally impossible puzzles, or flood the game with "spam" puzzles. These behaviors correspond with types of content that we have observed.

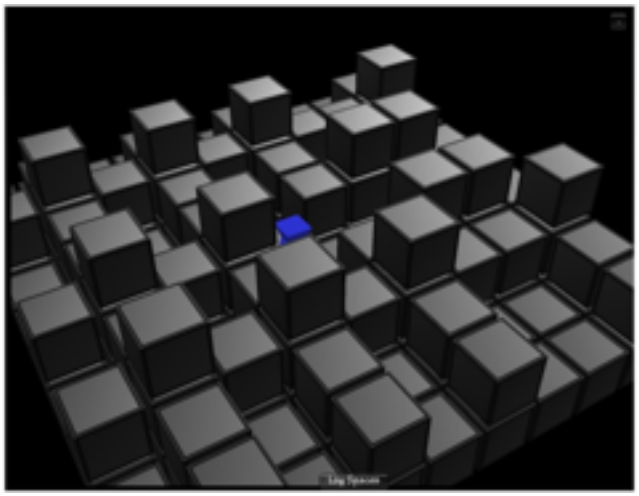

Figure 4: Puzzle with misleading or distracting structural cues. This puzzle contains irregularly placed blocks which don't lend themselves to any particular pattern. It is not apparent how loops or functions could intuitively help players with this puzzle.

2.2 Puzzle Categorization

We describe undesirable puzzles using four main categories.

2.2.1 *Sandbox Puzzles*

The first category of unwanted user-generated content is the *sandbox puzzle*. An example Sandbox puzzle is shown in Figure 6. Sandbox puzzles are characterized by the following traits:

Trivial or Non-existent solution The solution is straightforward once it is found, using few loops or functions. Difficulty comes more from visually finding the correct solution than from programming the robot to complete it.

Distracting structural elements Elements such as terrain blocks or unnecessary objectives are placed off-path, where the robot will never interact with them during the course of solving the problem.

Sandbox puzzles may be built by explorer players who are either new to the puzzle-creation interface, or socializer players who simply want to share something as quickly as possible. The solution (if there is one) is typically very straightforward. Structural elements are added for purely aesthetic value, spelling out words or forming shapes in the "unused' part of the puzzle. These puzzles are unwanted because such trivial puzzles do not address the core mechanics of the game. Additionally, these puzzles can be boring for players to solve.

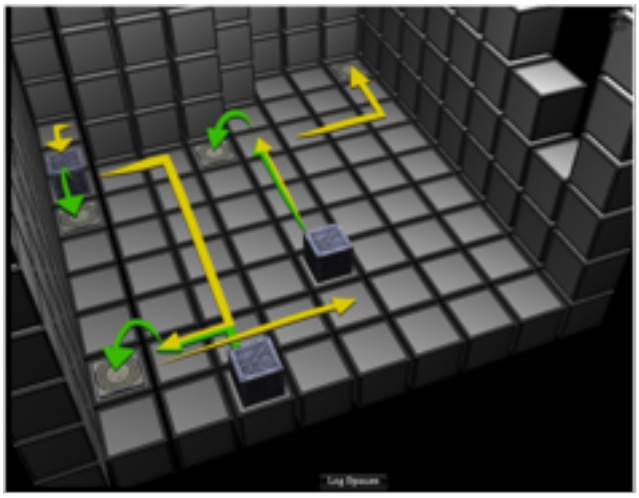

Figure 5: Puzzle with an obvious but tedious best solution. The robot (off screen) should move each block to the closest target (as shown with the green arrows), then stand on the target in the top right. However, there are very few iterative or repeated patterns which would allow a user to make their program shorter.

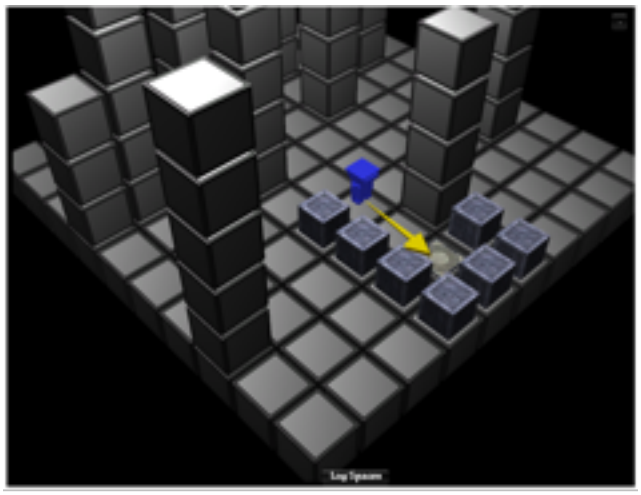

Figure 6: An example "sandbox" puzzle. The puzzle is simple to solve by simply moving the robot onto the visible target. There are a lot of crates and towers scattered around the puzzle which aren't relevant to this solution, but which might distract users.

2.2.2 *Power-Gamer Puzzles*

The next type of unwanted user puzzle is the *power-gamer puzzle*. such as the puzzle shown in Figure **??**. Power-gamer puzzles are characterized by the following traits:

Tedious solution The straightforward solution requires a very large number of "lines of code' even when written by an expert. The solution may be entirely impossible due to limited space.

Trivial or Non-existent solution as described above.

Few repeated patterns Repeated patterns give players opportunities to optimize their solutions by re-using code. These puzzles have no repeated patterns, so each objective must be handled separately.

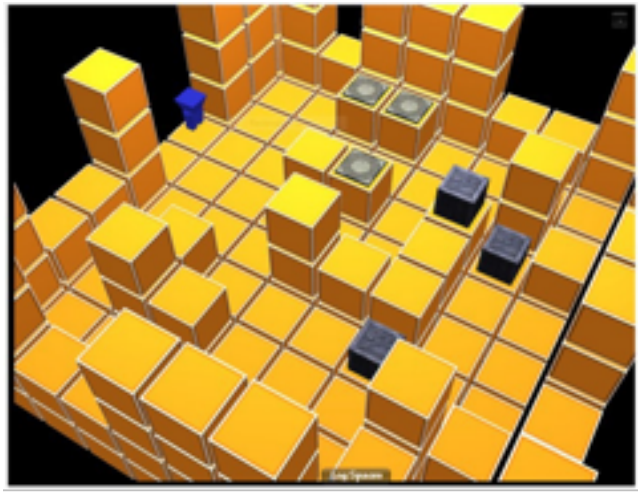

Figure 7: An example "power-gamer" puzzle. This puzzle, while possible to solve, requires over thirty instructions to build the most straightforward solution.

Figure 8: power-gamer

To create a difficult puzzle, griefer or killer players might seek to increase the time it takes to build a solution. This results in puzzles with conceptually simple solutions which take a long time to construct. These puzzles are undesirable because in order to be useful, the puzzle must have a solution that players can reach easily without requiring an unusual amount of effort or time from the player in terms of interface interactions.

2.2.3 *Griefer Puzzles*

The third type of unwanted puzzle we identified was the *griefer puzzle*, as shown in Figure 9. These puzzles exhibit the following characteristics:

Distracting structural elements as defined above

Trivial or Non-existent solution as defined above, but usually, the puzzle is obviously impossible because the robot is trapped, or the goal is inaccessible

Subversion of game mechanisms players use the puzzle title to communicate, spell words using objects, or block the player's view of the puzzle with objects or terrain

Griefer players create these in an effort to get other users to waste time on them before noticing that they simply cannot be completed. Griefer puzzles are distinguished by the perceived (and often communicated) intent of the designer to make them impossible.

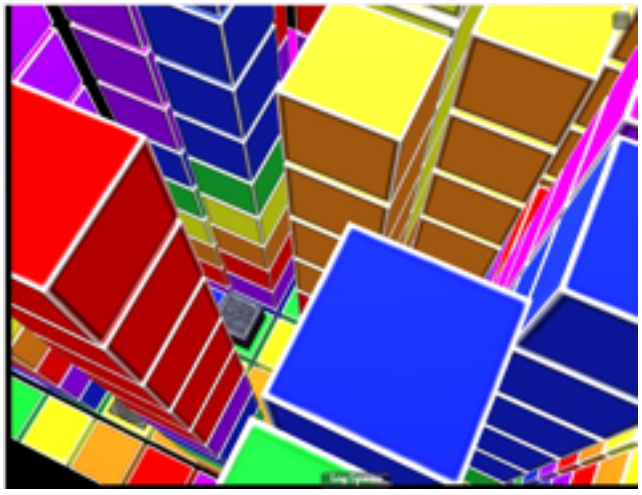

Figure 9: An example "griefer" puzzle, with high walls around the robot, the crates, and the targets, making it difficult for another player to maneuver the camera to see what they're doing.

2.2.4 *Trivial Puzzles*

Finally, a fourth type of unwanted puzzle is the *trivial puzzle*. An example Trivial puzzle can be seen in Figure 10. This type of puzzle has the following traits:

Trivial Solution as defined above

Simplistic Solution as defined above

These puzzles can usually be solved in one or two moves. Oftentimes the puzzles show a lack of understanding of the game's mechanics. For example, in a puzzle with one switch near the start and a crate placed at the end of a challenging maze. The author might intend for the player to go get the crate, but the player can solve the puzzle by simply stepping forward onto the switch.

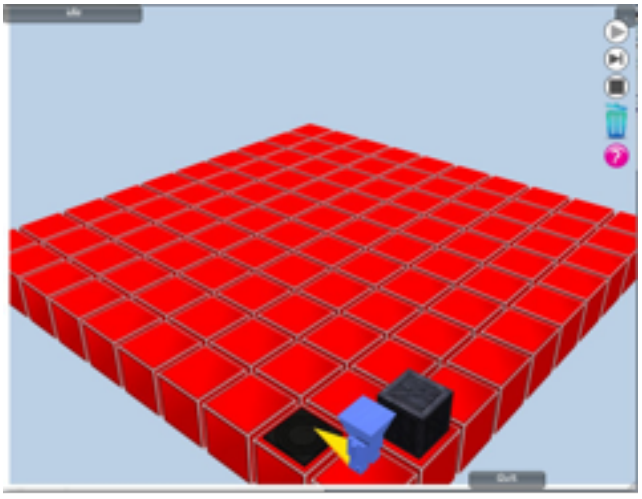

Figure 10: An example "trivial" puzzle. Even though there is a crate on this puzzle, it is not necessary to pick it up. The player can simply step forward onto the goal.

While the individual users' motive for creating unwanted pieces of content is useful for categorizing puzzles, our primary focus is on the qualities of the *content* not qualities of the *creator*. Our goal is to prevent puzzles with those qualities from causing frustration and disengagement for other players.

2.3 Hypothesis and Evaluation Design

We hope to discourage the creation of puzzles that are too easy, difficult, obscure, or boring, while also preventing their submission, keeping other users from accessing them. In order to do this without developing a domain-specific classifier to learn which types of content to filter, we decided to add an additional constraint to the puzzle submission process.

After examining the previously created puzzles and grouping them into the categories above, we identified the following *desirable qualities* for a puzzle to have:

The puzzle SHOULD be solvable.

The puzzle SHOULD have a straightforward solution. Players should be able to make progress towards completing the puzzle without having to use functions or loops.

The puzzle SHOULD contain opportunities for optimization. The straightforward solution should contain repeated patterns that can be encapsulated by functions or simplified by loops.

The puzzle SHOULD contain structural cues that show that optimization can be used. Hallways with raised walls highlighting the section to be repeated, or patterns of crates and switches that are used multiple times are good examples.

The puzzle SHOULD NOT contain unnecessary structural elements. Crates or terrain that are not part of the puzzle should be minimal.

The puzzle SHOULD be possible to complete in less than 5 minutes for an expert player. That is, the actual interactions (dragging buttons, creating functions) should take less than 5 minutes, without accounting for time spent analyzing the problem. (5 minutes is an arbitrary time limit, but we seek to prevent tedium by limiting this time).

Based on these quality criteria, we devised a change to the puzzle submission process which we hypothesized would reduce the number of low-quality puzzles submitted.

Condition 1: Unrestricted (Open) Puzzle Submission (Control)

Under this condition, there is no filtering process in place and the puzzle will be made public and immediately available for play as soon as the participant submits it. As there is no moderation on this group, we expect to see a pool of user generated puzzles that do not meet our desired design criteria.

Condition 2: "Solve and Submit" (S&S)

With all four types of unwanted content (sandbox, power-gamer, griefer, and trivial puzzles) the primary problem is that a player may be unable or unwilling to complete the puzzle. This could be due to the puzzle being impossible, seeming impossible, being too long, or simply being boring. If this is the case, we hypothesize that *requiring authors to solve their own puzzles as part of the submission process* will cut down on the number of unwanted puzzles submitted. If an author creates something that is unpleasant or impossible to solve, we believe that it is unlikely that they will go to the trouble to solve it themselves.

We expected that the total number of submitted puzzles would be somewhat fewer, but that the overall quality of the user generated puzzles would be higher for both submitted puzzles and created puzzles. We hypothesized that the authors of *griefer* and *power-gamer* puzzles would have less incentive to create multiple unwanted puzzles if other players will never have access to them).

To evaluate these hypotheses, we tested the game on two groups of middle-school students. Both groups were self-selected into STEM-related weekend day camps. However, students in one group were required to maintain a B average or higher in order to participate, while the other group had no such requirement. The age ranges for both programs were the same, with students in 6th through 8th graders (ages 11 - 13). Both groups have a skewed gender distribution, with less than one-third of the students in either group being female.

Each game session lasted 90 minutes. Players were required to first play through the tutorial up to a cut-off point. Once players had reached that point, they were free to enter the game's "Free Play" mode, where they were able to create and play custom puzzles.

Participants were divided randomly between the two conditions, "Open Submission"(Open) and "Solve and Submit" (S&S). All players played through the initial guided tutorial portion of the game the same way, experiencing the exact same introduction to the game and its mechanics. At this point, players were presented with a short description explaining the content creation process, including how their puzzles will be approved for inclusion in the game. Each player could only see puzzles created by other players in their same test condition. Players were asked by the group moderator to create at least one puzzle, and then solve at least one puzzle created by another user. Players could then choose to spend the rest of their time in the game however they wished.

After the session, a researcher blind to the study conditions scored each puzzle on its desirable qualities, and then classified the puzzles according to the categories defined above.

3. RESULTS AND DISCUSSION

Table 1: Mean quality scores by study condition

	n	M(Quality)
Open	23	2.78
S&S	24	3.92
S&S (sub)	11	4.55

Our hypothesis is that levels created under the S&S condition will be higher quality than levels created under Open Submission. We expected this to be true both for submitted levels, as well as for levels which were created but not submitted. We used two different methods of evaluation.

First, we checked to see if there was a difference in quality score of created levels between the students in the group with no grade requirement ($M = 3.33, SD = 2.33$), and students in the open group ($M = 3.38, SD = 2.03$). These

Table 2: Number of each type of level created, by study condition

	n	Sand-box	Power Gamer	Griefer	Trivial	Ok
Open	23	2	6	7	1	7
S&S	24	4	4	3	3	10
S&S (sub)	11	2	1	0	1	7

results are presented in Table 1. The groups were not significantly different ($t(45) = 0.16 < 2.01, p = .93$). Having established that the groups were not inherently different from each other, we next analyzed quality scores between the Open and S&S conditions. Quality scores of *created* puzzles in the S&S condition ($M = 3.91, SD = 1.91$) when compared with those in the Open Submission condition ($M = 2.78, SD = 2.28$) were largely higher, though not statistically significantly so. ($t(45) = 1.85 < 2.01, p = .07, d = .54$). The value of Cohen's d here shows that scores in the S&S condition were more than half of a standard deviation higher. We then compared the *submitted* puzzles only, between submitted puzzles in the S&S condition ($M = 4.55, SD = 1.36$) and submitted puzzles in the Open Submission condition ($M = 2.78, SD = 2.28$).We found that the puzzles in the S&S condition scored higher ($t(32) = 2.36, p = .024$) by nearly a full standard deviation ($d = .94$).

When we looked at the categories of the levels created under each condition (as shown in Table 2) we found that, for the most part, unwanted levels created under S&S were not submitted. Of 14 unacceptable levels created under the S&S condition, only 4 were submitted. However, there were a few exceptions, both unacceptable levels which were submitted, and acceptable levels which were not submitted.

First, we examined the unacceptable levels which were submitted under the S&S condition. No griefer levels were submitted, but there were a small number of sandbox, power-gamer, and trivial levels. These levels usually had borderline quality scores (3, 3, 3, and 6). Since our hypothesis regarding the S&S method was that players would be unlikely to play through "bad" puzzles, it makes sense that these borderline levels might be submitted. Our definitions could be improved take into account these borderline cases.

We want to ensure that our approach is not inappropriately filtering out quality levels. To do this, we looked at the set of acceptable levels which were not submitted under the S&S approach. One such level is shown in Figure 11. Out of 10 acceptable levels created under this condition, 3 were not submitted, and in each of those cases, the level had a quality score of 6, meaning it had all of the criteria outlined in Section 2.3. However, all of these levels had *tedious solutions*, taking a large amount of time to solve. Though these levels did not exhibit the other aspects of *power-gamer* levels (as outlined in section 2.2) we hypothesize that there was a large disparity between the author's skill and the skill needed to find the solution. In cases like this, presenting the level to a more skilled player may allow us to identify and distinguish these levels from *power-gamer* levels, accepting them as appropriate.

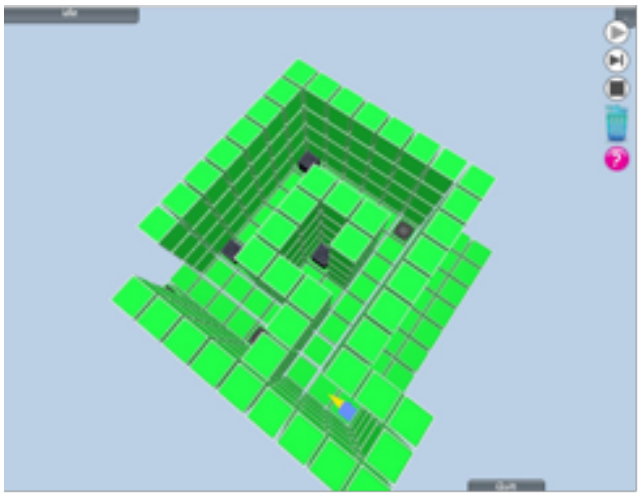

Figure 11: One of the acceptable levels which was not submitted. This level is difficult (and tedious) to solve without using loops.

4. CONCLUSIONS

We have shown that this "Solve and Submit" mechanism helps to prevent certain low-quality user-created puzzles from being submitted when compared to an open submission system. We are interested in further investigating the effect that the simple presence of a selection criteria might have on puzzle creation. However, we have only shown that this method prevents a number of unwanted puzzles from being submitted. Additionally, this mechanism also rejects puzzles which are acceptable but whose authors are unable to solve them. Proceeding forward, we hope to investigate other modifications to the puzzle submission process which may both reduce the number of false positives and increase the number of quality puzzles submitted, rather than simply filtering out the bad ones.

5. FUTURE WORK

The first step in expanding this research would be to replicate the experiment with other games, collecting more data to evaluate any effects that self-moderation versus admin moderation might have on players. Additionally, we've developed our categories of unwanted content from puzzles collected only within this game; looking at other games could help us better define those categories and provide insights into other ways of filtering those types of content. Interviews with level creators about their intentions, or speak-alouds collected while users create levels, could help provide a more solid foundation for our categories of unwanted content.

We also hope to be able to direct content generation to fill gaps in our difficulty curve and provide more personalized experiences. As cited earlier, work done by Aleahmad showed that content creators spent more time creating content when they were creating it for a specific individual's use (even if that individual was unknown or fictional). This technique could be harnessed in games by allowing content creators to build "hint" puzzles, structured to provide guidance to players who are having trouble.

Finally, we have observed that players transform the game

into a social experience even where no in-game social constructs exist. Players communicate with each other via puzzle names, challenging each other back and forth to create bigger, better things. Adding more tools for players to communicate during gameplay and puzzle creation might have a positive impact on their play experience, and could even allow for information to flow more freely between users, enhancing learning gains. It would be possible to test two versions of the game, an "isolated" version with all usernames removed, where content is still created and shared but not socially, and a "connected" version where players can view not only who made a puzzle, but who played it, and how they liked it via tags or comments. We believe that adding this level of connectedness between users would increase users' engagement and their sense of "ownership" of their content, but it may promote off-task behavior and harm learning gains.

6. ACKNOWLEDGMENTS

Thanks to the additional developers who have worked on this project or helped with our outreach activities so far, including Aaron Quidley, Trevor Brennan, Barry Peddycord, Vincent Bugica, Victoria Cooper, Dustin Culler, Shaun Pickford, Antoine Campbell, and Javier Olaya. This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. 0900860 and Grant No. 1252376.

7. REFERENCES

[1] T. Aleahmad, V. Aleven, and R. Kraut. Open community authoring of targeted worked example problems. In *Intelligent tutoring systems*, pages 216–227. Springer, 2008.

[2] M. Anderka, B. Stein, and N. Lipka. Predicting quality flaws in user-generated content: the case of wikipedia. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 981–990. ACM, 2012.

[3] R. Bartle. Hearts, clubs, diamonds, spades: Players who suit muds. *Journal of MUD research*, 1(1):19, 1996.

[4] R. A. Bartle. *Designing virtual worlds*. New Riders, 2004.

[5] J. D. Bayliss. Using games in introductory courses: tips from the trenches. In *ACM SIGCSE Bulletin*, volume 41, pages 337–341. ACM, 2009.

[6] A. Boyce, K. Doran, A. Campbell, S. Pickford, D. Culler, and T. Barnes. Beadloom game: adding competitive, user generated, and social features to increase motivation. In *Proceedings of the 6th International Conference on Foundations of Digital Games*, pages 139–146. ACM, 2011.

[7] M. Csiksczentmihalyi, C. Kolo, and T. Baur. Flow: The psychology of optimal experience. *Australian Occupational Therapy Journal*, 51(1):3–12, 2004.

[8] I. F. de Kereki. Scratch: Applications in computer science 1. In *Frontiers in Education Conference, 2008. FIE 2008. 38th Annual*, pages T3B–7. IEEE, 2008.

[9] R. Garfield. Roborally. [Board Game], 1994.

[10] K. Hullett and J. Whitehead. Design patterns in fps levels. In *proceedings of the Fifth International Conference on the Foundations of Digital Games*, pages 78–85. ACM, 2010.

[11] J. Juul. In search of lost time: on game goals and failure costs. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games*, pages 86–91. ACM, 2010.

[12] J. Juul and M. Norton. Easy to use and incredibly difficult: on the mythical border between interface and gameplay. In *Proceedings of the 4th international conference on foundations of digital Games*, pages 107–112. ACM, 2009.

[13] D. J. Malan and H. H. Leitner. Scratch for budding computer scientists. *ACM SIGCSE Bulletin*, 39(1):223–227, 2007.

[14] Maxis. Spore. [Video Game], 2008.

[15] Media Molecule. LittleBigPlanet. [Video Game], 2008.

[16] T. Murray. An overview of intelligent tutoring system authoring tools: Updated analysis of the state of the art. In *Authoring tools for advanced technology learning environments*, pages 491–544. Springer, 2003.

[17] S. Nicholson. A user-centered theoretical framework for meaningful gamification. *Games+ Learning+ Society*, 8, 2012.

[18] J. L. Plass, B. Homer, C. Kinzer, J. Frye, and K. Perlin. Learning mechanics and assessment mechanics for games for learning. *G4LI White Paper*, 1:2011, 2011.

[19] J. L. Plass and B. D. Homer. Educational game design pattern candidates. *Journal of Research in Science Teaching*, 44(1):133–153, 2009.

[20] M. Prensky. Computer games and learning: Digital game-based learning. *Handbook of computer game studies*, 18:97–122, 2005.

[21] A. Repenning, A. Basawapatna, and K. H. Koh. Making university education more like middle school computer club: facilitating the flow of inspiration. In *Proceedings of the 14th Western Canadian Conference on Computing Education*, pages 9–16. ACM, 2009.

[22] R. Smith. Helping your players feel smart: Puzzles as user interface, GDC 2009.

[23] D. Yaroslavski. LightBot. [Video Game], 2008.