Towards Automatic Personalized Content Generation for Platform Games

Noor Shaker, Georgios Yannakakis and Julian Togelius
Center of Computer Games Research
IT University of Copenhagen, Copenhagen, Denmark
{nosh, yannakakis, juto}@itu.dk

Abstract

In this paper, we show that personalized levels can be automatically generated for platform games. We build on previous work, where models were derived that predicted player experience based on features of level design and on playing styles. These models are constructed using preference learning, based on questionnaires administered to players after playing different levels. The contributions of the current paper are (1) more accurate models based on a much larger data set; (2) a mechanism for adapting level design parameters to given players and playing style; (3) evaluation of this adaptation mechanism using both algorithmic and human players. The results indicate that the adaptation mechanism effectively optimizes level design parameters for particular players.

1 Introduction

Procedural content generation (PCG) is an important technique for computer game development, and is likely to be of ever greater importance in the future; both offline, for making the game development process more efficient (design of content such as environments and animations now consume a major part of the development budget for most commercial games) and online, for increasing replay value, adapting games to particular demographics and enabling new types of games based on player-adapted content. The literature on personalized and player-adaptive PCG is so far scarce, as it is a new research direction (Togelius et al. 2010).

Emotions and player experience are critical in game design. Players may become frustrated when they don't progress well, pleased with themselves when they beat a level, or turn away in despair when encountering a seemingly insoluble problem. Emotions can be triggered by gameplay events (e.g. finding a treasure), by the behavior of a game character, or by interaction with the game (e.g., frustration when the game is too difficult) (Hudlicka 2008).

Successful computer games are typically fun and immersive. Fun can emerge from challenges met in the game and goals accomplished, which may vary from a victory in a scenario, to the accumulation of an asset, or to the right to move to the next level (Schuytema 2007).

Many theories exist regarding why we play games and what makes computer games fun (Bateman and Boon 2006; Isbister and Schaffer 2008; Koster 2004). However, these theories are qualitative rather than quantitative and tend to apply to games in general rather than to specific aspects of games. We still have to make several auxiliary assumptions if we want to develop algorithms that design or adapt games automatically. Therefore we need empirical research on particular games to acquire such models.

Optimization of game aspects based on empirically derived models has so far mostly focused on the impact of non player character (NPC) behavior (Andrade et al. 2005a; Missura and Gärtner 2009) and the adjustment of NPC behavioral parameters for maximizing satisfaction in games (Yannakakis and Hallam 2009). The focus of most research has been on dynamic game balancing which algorithmically changes parameters of the game to prevent players feeling frustrated because the game is too hard or bored because the game is too easy. The proposed algorithms are often restricted to specific game genres; for example, dynamic scripting is suitable only for games that are either scripted or imply storytelling. Lee and Jung (2006) work on dynamic scripting for a shooter game using a Gaussian Mixture Module that models the player's reaction pattern. Spronck et al. (2006) implement dynamic scripting through reinforcement learning to control the movement of the NPC. Andrade et al. (2005b; 2005a) also used reinforcement learning to modify NPC behaviors. They did not, however adjust the difficulty of the game level during play.

While procedural content generation of various kinds has been around in the game industry for decades, published attempts at generating coherent game segments (like levels) are scarce. When it comes to platform games, only a few attempts at generating levels can be found in the literature (Compton and Mateas 2006; Smith et al. 2009). Adaptive or personalized procedural content generation is a new research direction. Recent publications include attempts at optimizing tracks for car racing games (Togelius, De Nardi, and Lucas 2007); weapons for space shooter games (Hastings, Guha, and Stanley 2009); and rulesets for board games and predator-prey games (Marks and Hom 2007; Browne 2008; Togelius and Schmidhuber 2008).

A related direction focuses on incorporating the players' emotions into the game in a closed-loop manner. Here, player emotion is actively manipulated to ensure engagement (Hudlicka 2008). Existing work (Yannakakis and

Maragoudakis 2005; Charles and Black 2004) demonstrates the power of using affective player models to in-game situations of high interest and satisfaction for the players.

The focus and main contribution of this paper is the design of an online game adaptation mechanism that maximizes the player's fun value in platform games. The approach proposed extends and draws upon earlier work on modeling player experience in the same game (Pedersen, Togelius, and Yannakakis 2010). We extend this work through (1) reconstructing the computational model of player experience based on a new, larger data set, (2) designing an online adaptation mechanism based on the constructed models and (3) testing the model and mechanism using both human subjects and controllers taken from the 2009 Mario AI Competition.

2 Testbed Platform Game

The testbed platform game used for our study is a modified version of Markus Persson's *Infinite Mario Bros* (see Figure 1) which is a public domain clone of Nintendo's classic platform game *Super Mario Bros*. The original Infinite Mario Bros and its source code is available on the web.

Figure 1: Infinite Mario Bros game screenshot

The gameplay in Super Mario Bros consists of moving the player-controlled character, Mario, through two-dimensional levels. Mario can walk and run, duck, jump, and shoot fireballs. The main goal of each level is to get to the end of the level. Auxiliary goals include collecting as many coins as possible, and clearing the level as fast as possible. For more details about the game and our modifications the reader may refer to (Pedersen, Togelius, and Yannakakis 2010).

While implementing most features of Super Mario Bros, the standout feature of Infinite Mario Bros is the automatic generation of levels. Every time a new game is started, levels are randomly generated by traversing a fixed width and adding features according to certain heuristics as specified by placement parameters. In our modified version of Infinite Mario Bros we concentrate on a few selected game level parameters that affect game experience.

3 Data Collection

Data of three types was collected from 327 players.

1. Controllable features of the game: These parameters are used for level generation, and affect the type and difficulty of the level. This set contains three features that are related to gaps: number of gaps, average width of gaps, and gap entropy, as well as a switching feature that defines the percentage of the level played in the left direction.
2. Gameplay characteristics: Statistical features of how the user plays the game such as how often the player jumped, ran, died, how much he spent moving left, and how many enemies he killed for the different type of opponents. These features cannot be directly controlled by the game as they depend on the player's skill and playing style.
3. Player experience: After playing a set of four games, divided into two pairs played in both orders, players were asked to report the prefered game for three emotional dimensions; fun, challenge and frustration, through a 4-alternative forced choice questionnaire protocol.

A detailed description of which features were collected for each type of data as well as the modeling process can be found in (Pedersen, Togelius, and Yannakakis 2010). The analysis presented in this paper is based on 654 game pairs (1308 game sessions) played by 327 players, more than twice as many as in the previous paper. The collected data has been preprocessed to remove the pairs with unclear preferences (those pairs where both games are equally liked or disliked). After this step, 458, 463, and 463 pairs remain for fun, challenge, and frustration respectively.

4 Modeling of Player Experience Preferences

Initially, we use single layer perceptrons (SLPs) to approximate the affective state of the players. Sequential feature selection is used to choose the input subsets for the SLPs. All features (both gameplay and controllable) are investigated at this stage. After selecting the feature sets that maximize the accuracy in predicting the players preferences, we evolve the topology and weights of multi layer perceptrons (MLPs) to match reported player preferences.

Single Layer Perceptron Models: Feature Selection

A set of 58 features was extracted during gameplay (e.g. features related to time, player's actions, cause of death etc). We would like our model to be dependent on as few features as possible, not only to make it easier to analyze; but also to make it more useful for incorporation into future games for purposes of e.g. online game adaptation; and to save computational effort. Therefore, sequential forward feature selection (SFS) is used to extract the minimal feature subsets that yield the highest performance.

The performance of each player model is measured through the average classification accuracy of the SLP in three independent runs using 3-fold cross validation.

The set of features that yields the highest prediction accuracy differs for each emotional state investigated (+/- in parenthesis signifies positive or negative correlation). The selected feature subset for fun consists of four features: time

	Fun	$Frustration$	$Challenge$
MLP Topology	7-10-1	10-4-2-1	9-3-1
$\text{MLP}_{previous}$	74.21%	91.33%	79.37%
MLP	69.66%	89.33%	74.66%

Table 1: MLP performance for the previous and current model

needed to complete the level (+), time spent in large mode (+), time spent running (+), and number of unleashed shells (+). For frustration, a larger set of seven features were selected, consisting of: time needed to complete the level (+), total number of cannons fired in the level (+), number of power blocks destroyed (+), number of times the player shifted mode (+), average width of gaps (+), number of coin blocks destroyed (+), and number of jumps executed (-). Challenge can be predicted using a set of six features: number of collected coins (+), time needed to complete the level (-), average width of gaps (+), number of times the player fires (+), total number of cannons in the level (+), and number of times the player was killed because of a cannon (+).

Using SLPs with the selected features, we are able to predict fun, frustration and challenge with 64%, 84.66%, 70%, respectively.

Multi Layer Perceptron Models

Since our main aim is to automatically generate game content that is tailored to player experience in real-time, we need to be able to predict emotions, at least partly, from controllable features. For this purpose, all remaining controllable features which are not already included in the selected feature subset are forced into the input of MLP models and the topologies of the networks are optimized for maximum prediction accuracy.

As can be seen in table 1, the performance of the MLPs trained on the larger dataset is slightly lower than the one obtained previously (Pedersen, Togelius, and Yannakakis 2010). Overall, however, the approach proposed is able to predict the player's preferences with an acceptable accuracy and is able to generalize well across larger sets of data. Frustration achieves the highest performance 89.33% and maintained the smallest difference from the old model. We observed a drop in about 5% when predicting fun and challenge with a prediction accuracy of 69.66% and 74.66% compared to 74.21% and 79.37% for fun and challenge respectively.

It's also worth noting that the networks vary in size and topology. The topology for the challenge model is the smallest, consisting of one hidden layer that includes three neurons. Frustration is predicted with a network of two hidden layers while the network for fun consists of one hidden layer with ten neurons.

5 On-line Game Adaptation Mechanism

The aim of this study is to dynamically adapt level generation parameters based on player experience models, in order to optimize player experience. Our approach to on-line game adaptation is based on performing exhaustive search in the space of controllable features to find the combination of controllable features that, taken together with observed gameplay features, maximize the MLP output value.

The search space consists of four features; number of gaps, average width of gaps, gap placement and number of direction switches with value ranges of [4,10], [10,30], [0,1] and [0,1], respectively. The search space is explored by starting from the minimal possible values and at each step the values are increased by 1, 1, 0.1, and 0.1 respectively. Each configuration of controllable features is fed (together with the recently observed gameplay features) into the MLP; the combination that maximizes network output is chosen to generate the next level. With such a small search space (12000 configurations) we can find the optimal configuration almost instantly, allowing real time level generation.

6 Testing the On-line Game Adaptation

The adaptation mechanism is capable of generating a new level that optimizes some aspect of predicted player experience almost instantaneously given the playing style of the *previous* level. However, as the playing style of the same player on the *next* level is likely to differ from that of the previous level (because the level is different), the effect of the controllable features on player experience is rather indirect. It is therefore important to study the behaviour of the adaptation mechanism over time. In this paper, we test the adaptation mechanism using both humans and AI players. The reason for using AI players is that they behave consistently over time, and have more time to spend on playing computer games than the average human test subject.

Two different AI controllers were used to test the adaptation mechanism. Both controllers were submitted to the 2009 edition of the Gameplay track of the Mario AI Competition; a competition about designing controllers that play Infinite Mario Bros as well as possible, in the sense of completing as many levels as possible. See (Togelius, Karakovskiy, and Baumgarten 2010) for details about the rules of the competition, API and controllers submitted.

We used two different controllers in our experiments:

1. The agent that won the competition, submitted by Robin Baumgarten. This agent is based on an A* search algorithm in state space and simulates the future trajectory of both itself and enemy NPCs for each considered action. It performs very well on the type of levels generated by the level generator, as evidenced by it managing to finish all levels in the competition.
2. The competition entry of Sergio Lopez. This agent is based on a relatively simple heuristic function that decides when to jump and how high, and otherwise walks left. While performing less well than Robin's agent, it could still complete most of the levels in the competition.

The reason we chose these two controllers is that they differ not only in performance and architecture, but also in playing style. While Robin's agent runs through the levels, almost continuously jumping and shooting fireballs, Sergio's agent strolls along at a more leisurely pace, only jumping when necessary and never firing. In general, Sergio's agent looks rather more human-like and lacks the creepy exactness of Robin's agent. These behavioural differences are

directly reflected in metrics used by the player experience model, allowing us to test the adaptation mechanism with automated playthroughs of significantly different styles.

In the following sections we will discuss a number of experiments that have been done to test the model performance and the validity of the proposed approach. In the first experiment we test the performance of the mechanism by optimizing levels to playing characteristics of a specific player. In the second experiment, we test the generality of the mechanism by optimizing levels for changing playing styles. In both experiments, all agents played in real time while gameplay features were recorded.

7 Experiment 1: Optimizing Player Experience for a Fixed Playing Style

The experiment presented in this section is focused on generating optimized levels for a specific player. The experiment is done in the following steps:

1. An initial level is generated with random parameters.
2. An AI agent plays the level while gameplay features are recorded.
3. Using the set of recorded gameplay features, the values for the controllable features that optimizes the player experience are chosen.
4. A new adapted level is generated based on the optimized controllable features.

The two AI agents have each played a set of 50 levels with the first level generated randomly, followed by a set of adapted levels that aim at maximizing player experience based on gameplay during the previous level. The experiment is repeated 10 times for each agent starting from a different random level each time.

For comparison purposes, 50 random levels were generated by assigning random values for the controllable features instead of searching for the optimal set of values. The results depicted in figure 2 show the performance of the proposed approach against the randomly generated levels: our adaptation mechanism is able to generate levels with higher predicted fun level than the baseline random levels. (In the discussion in this section, "fun" always refers to *predicted* fun levels. We do not claim that the algorithms had fun.)

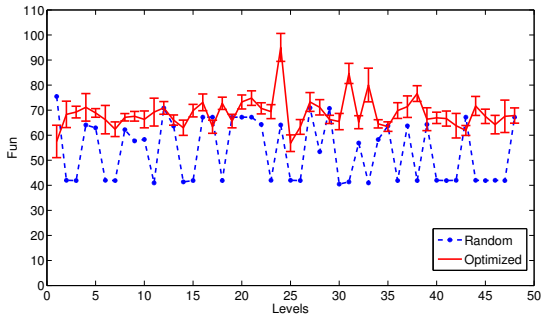

Figure 2: Optimized fun levels vs. random levels for Robin's agent

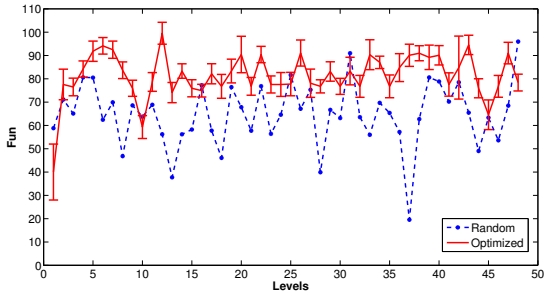

Figure 3: Optimized fun levels vs. random levels for Sergio's agent

We were able to generate fun levels for Sergio's agent that are 81.31% fun, while the optimized fun levels for Robin's agent have a slightly lower fun value with an average 68.71% (see figures 2, 3, 4 and 5). This difference is related to the agent's playing style. The observation of each agent playing characteristics showed that Sergio's agent plays in a more human-like style; he moves at an average speed so it is able to complete the levels approximately at the same time as a human player would, and he jumps only when necessary. Robin's agent, on the other hand, is able to clear the levels fast, he keeps jumping and he fires even when unnecessary.

We were able to construct levels with higher fun values for the agent who plays in a human like manner than we could for the other agent. Since the model has been trained on data collected from human players, we suspect that this is due to the nature of the data our fun model was built on. Sergio's agent playing style may matches a pattern of behavior existent in human players.

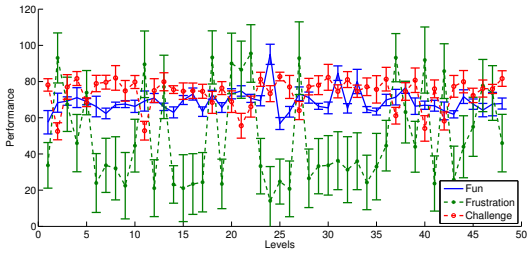

Figure 4: Optimized fun levels for Robin's agent

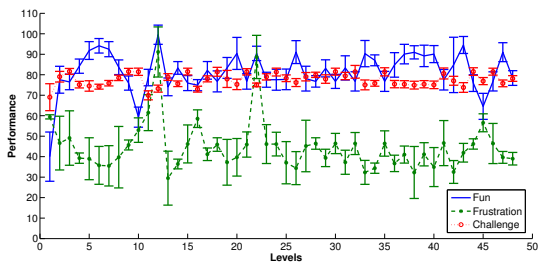

Figure 5: Optimized fun levels for Sergio's agent

Statistical Analysis

This section describes testing the correlations between the three emotions that we are investigating.

To check whether optimizing for one reported emotional state affects the others, we generated levels to optimize the fun value while monitoring how the values change for frustration and challenge. We performed an analysis for exploring statistically significant correlations between these emotions. We use the correlation coefficient as described in (Pedersen, Togelius, and Yannakakis 2010). A statistically significant effect ($p < 0.05$) is observed between challenge and frustration ($p = 3.57 * 10^{-20}$) for Robin's agent. The negative correlation (-0.66) between these two emotions shows that for players like Robin's agent challenging levels are predicted to be less frustrating.

The statistical analysis for Sergio's agent showed no significant correlation between any of the emotions investigated. This shows the sensitivity of the model to the type of player and the player's playing style. We also observed that the levels optimized for fun induced high levels of challenge for both agents.

8 Experiment 2: Dynamic Adaptation to Changing Playing Styles

In this experiment we test the model's ability to generalize over different types of players. For this purpose, the two AI agents were set to play in turns while monitoring how the fun value evolves.

The experiment starts from a randomly generated level. The agents play a set of 100 levels and every 20 levels the playing agent is switched. The result in Figure 6 shows the changes in fun value over 100 levels. The figure shows that the fun value ranges around 70% in the first 20 levels where Robin's agent is playing, and when we switch the agent to Sergio's agent in the following 20 levels, it increases to 80% approximately, and it drops again to 70% when Robin's agent is set back to play. These results provide evidence for the model's ability to adapt to the types of players.

For further investigation, we did the same experiment on human players playing a smaller set of 12 levels. The outcome of this experiment is illustrated in figure 7, which shows the evolution of fun over 48 levels played by four different human players. The results obtained is similar to the ones obtained with the AI agents with fun value averages 80.65%, 93.88%82.11%,85.49% for first, second, third and fourth player respectively. As seen from Figure 7 the adaptation mechanism is robust enough to adapt to a particular player and to generalize over different types of players. Participants in this experiment along with 6 other participants were asked to report their preferences between the randomly generated level and the first adapted level, results show that 60% of the participants enjoyed the adapted level more than the random level.

9 Discussion

For the experiments presented in this paper, we only defined four controllable features, three of which are related to gaps

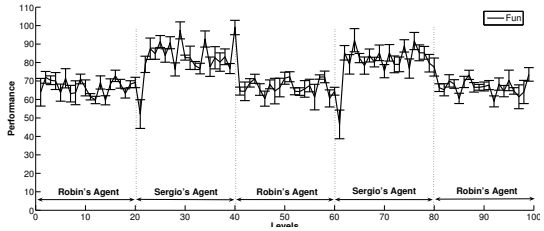

Figure 6: Optimized fun levels for two AI agents

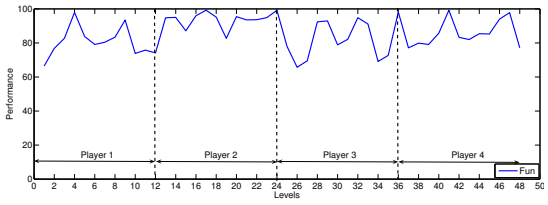

Figure 7: Optimized fun levels for four human players

in the level. Although we were able to predict the emotional states with a relatively high accuracy and adapt well to player's playing style and characteristics, for our future work we would like to investigate many more features.

Although the methodology proposed shows promising results in terms of optimizing player experience and adapting to changing playing styles, the results presented are somehow suggestive in that they are based on experiments on two AI agents and four human players. Future work will include further improvement and testing for the model.

We believe that our approach will generalize well to other games, at least first- or third-person action games, since the features we define have straightforward analogues in those games. Not only do many such games have platform elements, but also in a typical FPS like *Halo* concepts such as average movement speed, speed of progress, number of shots fired, entropy of object placement, and number and concentration of enemies/items/obstacles are clearly defined and can be assumed to have impact on player experience.

One of the limitations of the proposed approach is post-experience. Subjective techniques only generate data when a question is asked, and interrupting game play to ask a question can be intrusive. There exist solutions for both testing emotional models over different time windows as introduced in (Yannakakis and Hallam 2009) and capturing the association between gameplay dynamics and emotional responses via e.g. recurrent neural networks. In the future, we will combine these techniques with our approach to adaptive content generation; we will also investigate other sources of information, like physiological measures. Naturally, when better models have been derived, the efficacy of the adaptation mechanism will be validated with human players.

In the experiments reported here, new game content was generated at the end of each level. The next step towards more responsive on-line adaptation to the player is to replace the level unit with a smaller time window in which we assume that the entertainment preference is constant. This

will allow us to include features that are based on ordering in time and will result in more fine-grained adaptation.

10 Conclusion

The work reported in this paper introduces an on-line game adaptation mechanism that can be used to effectively optimize player experience. We saw that using the proposed approach we were able to generate levels tailored to specific players. While our experiments were successful in the sense that the predictors achieved acceptable accuracy and the adaptation mechanism shows promising results, there are many exciting ideas for further improvement.

Acknowledgements

We would like to thank the participants in the experiments. The research was supported in part by the Danish Research Agency, Ministry of Science, Technology and Innovation; project "AGameComIn" (274-09-0083).

References

Andrade, G.; Ramalho, G.; Santana, H.; and Corruble, V. 2005a. Automatic computer game balancing: a reinforcement learning approach. In *AAMAS '05: Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems*, 1111–1112. New York, NY, USA: ACM.

Andrade, G.; Ramalho, G.; Santana, H.; and Corruble, V. 2005b. Challenge-sensitive action selection: an application to game balancing. In *IAT '05: Proceedings of the IEEE/WIC/ACM International Conference on Intelligent Agent Technology*, 194–200. Washington, DC, USA: IEEE Computer Society.

Bateman, C., and Boon, R. 2006. *21st century game design*. Charles River Media.

Browne, C. 2008. *Automatic generation and evaluation of recombination games*. Ph.D. Dissertation, Queensland University of Technology.

Charles, D., and Black, M. 2004. Dynamic player modeling: A framework for player-centered digital games. In *Proc. of the International Conference on Computer Games: Artificial Intelligence, Design and Education*, 29–35.

Compton, K., and Mateas, M. 2006. Procedural level design for platform games. In *AIIDE'06: Artificial Intelligence and Interactive Digital Entertainment International Conference*.

Hastings, E.; Guha, R.; and Stanley, K. 2009. Demonstrating automatic content generation in the galactic arms race video game. In *Artificial Intelligence for Interactive Digital Entertainment Conference*.

Hudlicka, E. 2008. Affective computing for game design. In *GAMEON-NA'08: Proceedings of the 4th Intl. North American Conference on Intelligent Games and Simulation*, 5–12.

Isbister, K., and Schaffer, N. 2008. *Game Usability: Advancing the Player Experience*. Morgan Kaufman.

Koster, R. 2004. *A theory of fun for game design*. Paraglyph press.

Lee, S., and Jung, K. 2006. Dynamic game level design using gaussian mixture model. In *PRICAI'06: Proceedings of the 9th Pacific Rim international conference on Artificial intelligence*, 955–959. Berlin, Heidelberg: Springer-Verlag.

Marks, J., and Hom, V. 2007. Automatic design of balanced board games. In *AIIDE*, 25–30.

Missura, O., and Gärtner, T. 2009. Player modeling for intelligent difficulty adjustment. In *DS '09: Proceedings of the 12th International Conference on Discovery Science*, 197–211. Berlin, Heidelberg: Springer-Verlag.

Pagulayan, R. J.; Keeker, K.; Wixon, D.; Romero, R. L.; and Fuller, T. 2003. User-centered design in games. In *The human-computer interaction handbook: fundamentals, evolving technologies and emerging applications*, 883–906. Hillsdale, NJ, USA: L. Erlbaum Associates Inc.

Pedersen, C.; Togelius, J.; and Yannakakis, G. N. 2009. Modeling player experience in super mario bros. In *CIG'09: Proceedings of the 5th international conference on Computational Intelligence and Games*, 132–139. Piscataway, NJ, USA: IEEE Press.

Pedersen, C.; Togelius, J.; and Yannakakis, G. N. 2010. Modeling player experience for content creation. *IEEE Transactions on Computational Intelligence and AI in Games* 2(1):54–67.

Schuytema, P. 2007. *Game Design: A Practical Approach*. Charles River Media.

Smith, G.; Treanor, M.; Whitehead, J.; and Mateas, M. 2009. Rhythm-based level generation for 2d platformers. In *Proceedings of the 2009 Int'l Conference on the Foundations of Digital Games*.

Spronck, P.; Ponsen, M.; Sprinkhuizen-Kuyper, I.; and Postma, E. 2006. Adaptive game ai with dynamic scripting. *Mach. Learn.* 63(3):217–248.

Togelius, J., and Schmidhuber, J. 2008. An experiment in automatic game design. In *CIG'08: IEEE Symposium on Computational Intelligence and Games*, 252–259.

Togelius, J.; Yannakakis, G. N.; Stanley, K. O.; and Browne, C. 2010. Search-based procedural content generation. In *Proceedings of EvoGames*.

Togelius, J.; De Nardi, R.; and Lucas, S. M. 2007. Towards automatic personalised content creation in racing games. In *Proceedings of the IEEE Symposium on Computational Intelligence and Games*.

Togelius, J.; Karakovskiy, S.; and Baumgarten, R. 2010. The 2010 mario ai competition. In *Proceedings of the Congress on Evolutionary Computation*.

Yannakakis, G., and Hallam, J. 2009. Real-time Game Adaptation for Optimizing Player Satisfaction. *IEEE Transactions on Computational Intelligence and AI in Games* 1(2):121–133.

Yannakakis, G., and Maragoudakis. 2005. Player modeling impact on players entertainment in computer games. In *10th International Conference on User Modeling*, 74–78.