Automatic Game Tuning for Strategic Diversity

Raluca D. Gaina
University of Essex
Colchester, UK
rdgain@essex.ac.uk

Rokas Volkovas
University of Essex
Colchester, UK
rv16826@essex.ac.uk

Carlos González Díaz
University of York
York, UK
cgd506@york.ac.uk

Rory Davidson
University of York
York, UK
rd553@york.ac.uk

***Abstract*—Finding the ideal game parameters is a common problem solved by game designers by manually tweaking game parameters. The aim is to ensure the desired gameplay outcomes for a specific game, a tedious process which could be alleviated through the use of Artificial Intelligence: using automatic game tuning. This paper presents an example of this process and introduces the concept of simulation based fitness evaluation focused on strategic diversity. A simple but effective Random Mutation Hill Climber algorithm is used to evolve a Zelda inspired game, by ensuring that agents using distinct heuristics are capable of achieving similar degrees of fitness. Two versions of the same game are presented to human players and their gameplay data is analyzed to identify whether they indeed find slightly more varied paths to the goal in the game evolved to be the more strategically diverse. Although the evolutionary process yields promising results, the human trials are unable to conclude a statistically significant difference between the two variants.**

I. Introduction

With the advent of powerful machines capable of processing large amounts of data in a relatively short period of time, areas of research previously not considered due to their complexity are becoming more popular. One of these areas is training artificial agents to play video games, requiring simulations of playthroughs as well as executing machine learning algorithms to improve their performance. Similarly, to alleviate the time-consuming work of video game level designers, search algorithms can be exploited to find the desired configurations of a predefined level in order to achieve a designer chosen goal. When looking at General Game Artificial Intelligence, the General Video Game AI Framework (GVGAI) [1] is gaining popularity among researchers. GVGAI aims to improve the capabilities of agents to gather knowledge, which could translate initially to other previously unseen video games and eventually to real world tasks.

Agents in commercial video games most often manifest themselves in the form of Non-Playable Characters, which the player interacts with. The main part of experience crafting through level building and agent algorithm selection rests upon the shoulders of designers. Agent based quality assurance is not a new idea and procedural content generation has spawned entire genres of games. In this paper, much like in [2], the interest is directed towards the creation of tuned game variations, providing players with the ability to complete the level in a number of strategies without disadvantaging either one of them.

The rest of this paper is structured as follows. Section II looks at previous literature in the area. Section III gives a brief background on the framework and algorithms used in this study. Section IV presents the game parameter tuning process and results, while Section V depicts the user trials and results. Section VI concludes the paper and describes several lines of future work.

II. Literature Review

Literature has previously looked at various areas of Procedural Content Generation (PCG). Togelius et al. have looked at search based PCG, with a focus on the types of content these methods are able to generate, the specific representations of the content and evaluation techniques [3]. They highlight several ways of evaluating generated content and several small successful experiments in which simulation based fitness functions return good results.

As a sub-field of Game Artificial Intelligence, automatic game design can be used to generate game content such as levels [4], [5] through various methods. Isaksen et al. [6] explore game spaces in their work, by analyzing how variations in parameter values, without altering the game rules, are able to increase or decrease the difficulty of a game and relating this to real human experience.

Togelius and Schmidhuber [7] propose a novel approach of starting from scratch and generating the game rules through evolutionary computation. They use neural networks and evolutionary algorithms to learn to play the new games generated and evaluate them based on the game-playing AI performance. Even though some of the content produced shows promise, they take a long time to compute due to the double evolution and learning involved in the process; additionally, most of the winnable games turn out to be fairly easy to beat, due to the limited capabilities of the agent used for fitness evaluation.

One simulation based method often used in literature for game or level evaluation is the measuring of skill depth. This method involves several game-playing agents with various levels of skill in playing the respective game, the aim being to maximize the difference between the agents' performance. Perez et al. apply this technique for automatic map generation in the Physical Traveling Salesman Problem [4], while Kunanusont et al. look at game parameter evolution in a classic Space Battle game [8].

Work has recently moved towards general automatic game design. Liu et al. [2] apply the same measure of skill depth in Space Battle, but using several black-box general game-playing agents from the General Video Game AI Competition

(GVGAI) [1]. They use a simple evolutionary algorithm for game tuning and their results indicate potential success, although conditioned by the resampling rate for noise reduction in this stochastic problem.

Since games are played by people, it is important to understand how players deal with the artifacts designers produce. Player Experience (PE) is a multi-factorial construct that has been researched in literature with different approaches and definitions. Engagement, immersion, flow or enjoyment are a few of the different constructs that are usually related with Player Experience [9], [10], [11]. Game Feel [12] is a specially interesting one, as it links the *feel* of a game with its aesthetics, physics simulations, controls or rules [12], [13]. It is important to take into account that while AI agents don't report many of these constructs, people do, thus it is important to design games around the different needs that a player might experience.

This study looks at a different simulation-based fitness measure focused on the difference in agent strategies, as opposed to their ability of winning the game. Although literature does not account for this aspect, there are several attempts at studying AI agent behavior and heuristic design. Perez et al. present a multi-objective optimization tree search approach, applied to GVGAI [14], in which they varied the heuristic applied to a Monte Carlo Tree Search agent, adding an exploration policy based on pheromone trails. Mendes et al. [15] took a different approach by using several GVGAI algorithms employing various strategies or methods and a hyper heuristic to choose between them. Guerrerro et al. [16] try to diversify heuristics and penalize winning to encourage different behavior, such as exploring the level more or gathering information about the various game objects; several of the agents used in this study are inspired by their work.

III. Background

A. Framework

The General Video Game AI Framework was used for this study, due to it becoming a popular benchmark in the area General Video Game Playing [17] in recent years and attracting the interest of several researchers who attempt to develop general purpose problem solvers. Therefore, a wide range of algorithms and techniques are available for automatic play testing.

The games in GVGAI are depicted in the Video Game Description Language (VGDL) [18], a declarative language which uses two text files to define a game: one file for game description (sprites used in the game, their interactions, their mapping to the level file and end conditions) and another for level definition (an ASCII matrix, each character corresponding to a game object). The specific definitions of the sprites, interactions and terminations are written in Java (or Python in the original version by Tom Schaul).

Although the games are restricted to 2D grid-physics (recently expanded to continuous physics), the simple definition allows for a large number of games to be easily described and new problems for AI agents to solve easily created. While originally focused on planning algorithms (single and two-player [19]), the GVGAI Competition has added a Level Generation track in the past year [20]. A recent expansion of the GVGAI framework has taken general PCG a step further, by permitting parameterization of games, thus making it a good choice for this study. The generality of the method used, as well as the agents and heuristics, make this experiment applicable to other problems as well.

B. Evolution

Evolutionary Algorithms are a large family of methods which improve a population of individuals over several generations, through various techniques such as mutation or crossover. Each individual encodes the solution to a problem and their evaluation is problem-dependent. There may be several end conditions, such as time or memory budget reached. In this context, the algorithms is run for a specific number of iterations. The solution chosen and applied to the game is that with the best fitness at the end of the evolutionary process.

C. Game-playing agents

All but one of the agents employed during the automatic tuning process use Monte Carlo Tree Search (MCTS) [21], specifically the implementation of the sample agent in the GVGAI Framework.

MCTS is a planning algorithm which searches the space iteratively by picking various actions available, simulating them using a Forward Model provided by the framework and building a statistical tree to make its choices. It begins with a selection step, in which it chooses one tree node not yet fully expanded and adds a new child of this node to the tree. A Monte Carlo simulation is run from this new node until a pre-defined depth is reached, then the state is evaluated with a heuristic function and the value used to update all the nodes visited during the iteration. At the end of the budget it uses the statistics gathered and returns an action to play in the game according to its recommendation policy. The recommendation policy in all agents used for this experiment selects the most visited child of the root node.

The last agent is a very simple implementation which chooses in turn each of the actions available in the current time step, repeats each a number N times and picks the action which returned the highest value according to its simple heuristic (maximizing score and aiming to win).

IV. Game parameter tuning

A. Game

The game used in this experiment is based on the popular top down action adventure *Legend of Zelda*. Figure 1 shows the layout designed for the experimental runs with the intention of the level supporting the alternative approaches with some set of chosen parameters prior to their adjustment though evolution. Certain alterations to the original *Legend of Zelda* design where introduced to enforce collection mechanics. The player can move using the arrow keys in the keyboard, and collect one or more of the pickaxes. By pressing the space bar,

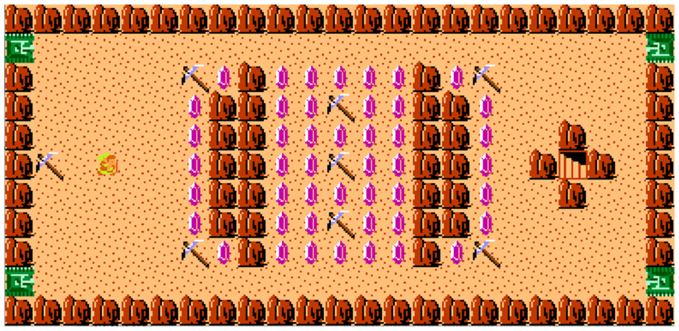
Figure 1: Experimental game level

Idx	Name	Num. dimensions	Range
1	Tank_Speed	11	0:0.2:1
2	Score_Pickaxe	4	0:5:15
3	Score_Wall_Kill	4	0:5:15
4	Pickax_Value	3	1:1:3
5	Time_Bonus	2	True:False
6	Score_Gold	4	0:5:15
7	Pickax_Limit	3	1:1:3
8	Pickax_Cooldown	4	10:5:25

Table I: Parameter list. Name, number of dimensions and range (presented as start : step : end) for each parameter evolved.

the player can throw the accumulated pickaxes to the rocks and destroy them, allowing to open new paths. Coins/ruppes can be collected by walking over them. The green tanks at the edges of the level move horizontally in the row, destroying rocks and killing the avatar instantly if they collide with the tank. Finally, the stairs at the right-end of the map represent the goal, and the game ends when reaching it.

Addressing the importance of Game Feel [12], the GVGAI framework was modified to allow the inclusion of more animations, music and sounds in the game, improving the *feel* of the game with *8-bit* melodies that fit the visual aesthetics.

B. Parameters

8 parameters of the game were tuned through evolution, see Table I for details. Most ranges included the possibility of the parameter being disabled (by setting its value to 0 or False, in case of boolean variables). The size of the resulting search space is $2.765E4$.

Each parameter impacts gameplay and strategies in different ways. For example, a bigger cooldown for pickaxes means there is a longer wait for pickaxes to respawn and then use them to destroy walls, therefore there is less interaction with environment. Additionally, the time bonus aspect adds pressure to head straight for the exit and finish the game quickly.

C. Heuristics

There were 3 heuristics used as alternative strategies when evaluating the levels. The heuristics return a normalized reward upon each roll out, which is used to determine the next action to be taken by MCTS. These heuristics were:

- Default - maximizes the game score, gains a large bonus for winning and a large penalty for losing and is used as a base for other heuristics
- Explorer - stores the locations it has visited before and gains bonus rewards for visiting areas not already seen
- Interactive - gains rewards for moving to the positions of objects it has not interacted with before, thus causing collision / interaction; if no new interactions can be triggered, it will move towards the closest object it has not collided with
- Stubborn - gets rewards for taking the same action repeatedly, but only enough to not walk into dangerous situations, providing negative score gains

D. Parameter evolution

In this study, a simple Evolutionary Algorithm was used to tune the game parameters, similar to work done in [22], run for a specific number of generations (150), with only 1 individual per population. Each gene in the individual represents a game parameter, taking values within a predefined range. One gene in the individual is chosen at random and mutated in each generation to create a new offspring (the algorithm therefore becoming a Random Mutation Hill Climber [23]). Both the offspring and the parent are evaluated at each iteration, therefore updating the statistics of the parents to reduce noise. The better individual is carried forward to the next generation, while keeping track of the worst individual discovered.

In order to evaluate an individual, 5 different agents (the simple agent, A_1, default MCTS, A_2 and the 3 specialized heuristics, H_1, H_2 and H_3, see Sections III-C and IV-C) play the game depicted by the specific set of parameters. Resampling is used to reduce noise ($r = 2$, therefore 10 games are played per evaluation). The fitness function (Equation 3) maximizes the scores of agents H_1, H_2 and H_3 and minimizes the scores of agents A_1 and A_2. Large bonuses are given for the maximization agents winning the games and large penalties are given for the minimizing agents winning the games.

$$f_A = min(Score(A_1A_2) + W(A_1A_2) \times B) \tag{1}$$

$$f_B = max(Score(H_1H_2H_3) + W(H_1H_2H_3) \times B) \tag{2}$$

$$f = f_A + f_B \tag{3}$$

The best and worst solutions found at the end of the evolution become the game variants used in the user trials (Game A and Game B, respectively). Users are therefore expected to experience greater strategic diversity and be forced to make more choices in Game A than Game B. The idea justifying the cost function lies in the fact that the game parameters are evolved so as to allow various advanced strategies to win, rather than random or a point maximizer function.

E. Game tuning results

Figure 2 presents the results of the evolutionary process. The maximum fitness achieved was 1949.8, while the lowest was -1407.625 and the final best fitness 306.

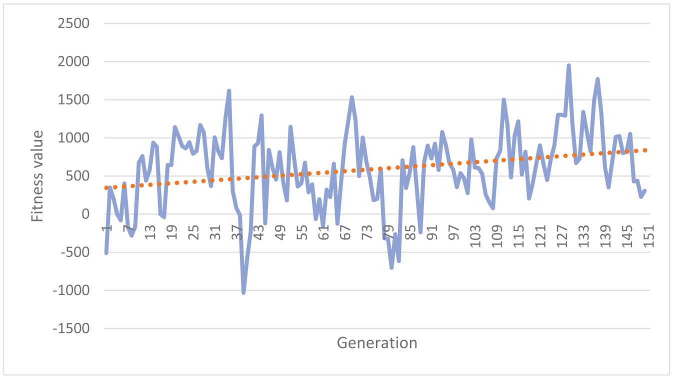

Figure 2: Evolution results.

Idx	Name	Final best	Alltime best	Worst
1	Tank_Speed	0.8	0	0.4
2	Score_Pickaxe	5	0	15
3	Score_Wall_Kill	10	15	10
4	Pickaxe_Value	1	1	3
5	Time_Bonus	False	False	True
6	Score_Gold	15	15	15
7	Pickaxe_Limit	3	2	2
8	Pickaxe_Cooldown	10	25	10

Table II: Parameter values evolved.

Although the fitness evaluation is very noisy due to the stochastic nature of the agents, as well as a small probabilistic aspect of the game (pickaxes spawning with a probability of 0.1), there is an upwards trend noticed in the average fitness. The number of generations is fairly low (150), but it is expected that, due to reevaluation of previous individuals in the evolution, the latter fitness evaluations are more accurate. Due to this aspect, the games put forward to the human player trials are the final recommendations of the algorithm. Game A is the individual with the final best fitness, while Game B takes the parameter set of the individual with the final worst fitness (see Table II for values).

The 3 final individuals were validated for more accurate values, by running 20 evaluations of each parameter set. The resulting fitness values (alltime best, final best and worst) are as follows: 1185.1, 592.89, -768.575. This confirms that although the absolute values are different to those estimated during evolution, the relative ranking of the individuals remains the same.

The points offered by the gold pick up appears to be irrelevant, taking the same values in all 3 variations obtained. This could be due to the fact that all heuristics give weight to the game score, therefore they all attempt to maximize their score, while diverging in the ways they achieve this goal.

Two conflicting parameters are the tank speed and the pickaxe cooldown, taking on values at opposite ends of the spectrum in the final best and alltime best variants. This aspect is thought to be due to the tanks and pickaxes not actually playing a big role in either of the games.

Three parameters appear to agree on similar values in strategically diverse games versus their opposite: time bonus, pickaxe value and the points offered by the pickax. The fact that low or no score is offered in “good” games, while the wall reward is kept fairly high, indicates that delayed rewards are more inviting for different strategies. Similarly, the pickaxe value takes the minimum in the “good” games, highlighting the need for the player to make more actions to get rewards, possibly exploring more of the level and interacting more with the environment. The time bonus only being offered in the “bad” game suggests that the pressure of making it to the exit quickly narrows down the exploration of strategies.

V. User trials

A. *Experimental setup*

In the evolutionary tuning phase, two versions of the game were evolved for high (version A) and low (version B) levels of strategic diversity. In order to validate this measurement, it was then necessary to investigate the connection between this AI-based measure of strategic diversity and the actual experience of human players. This connection was captured in two hypotheses. The first hypothesis was that players in the high-diversity condition would perceive more choices made during the course of play. The second hypothesis was that players would enjoy the game more in the high-diversity condition, indicating that increased choice and hence strategic diversity represents a useful metric when designing enjoyable games.

The experiment used a 2x2 within-subjects setup; the dependent variables were the game version perceived as having the most choice and the game version that was most enjoyable. In total, 25 players participated in the experiment, of which 10 were men, 7 were women and 8 were neither or preferred not to answer.

Participants were asked to play both versions of the game. In order to minimize practice and ordering effects, players were initially presented with a simple ”training” level in order to become used to the controls and understand the rules of the game, and the order in which the two versions were presented was counterbalanced. After playing both versions, players were asked which they enjoyed more, which they felt they made more choices in, and whether they had been paying attention to the game score. Players then participated in a brief questionnaire and debriefing interview.

B. *User trials results*

Despite having 25 participants, only 8 expressed noticing a difference between the two game versions in terms of choiceA one-sample t-test was carried out using only the data of those participants who expressed a difference in the levels of choice experienced in either game. The analysis indicated that while more such participants felt that game A provided more choice (6 choosing A and 2 choosing B), this difference was not significant (p = .170). A one-sample t-test was carried out on the participant preference for game version. The analysis indicated that there was no significant difference in preference between games (p = .667)

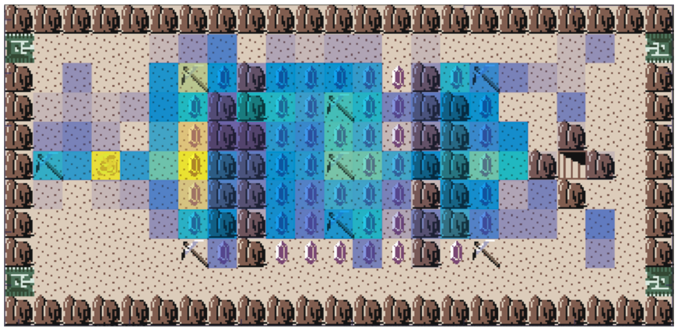
Figure 3: Game A configuration positional heatmap

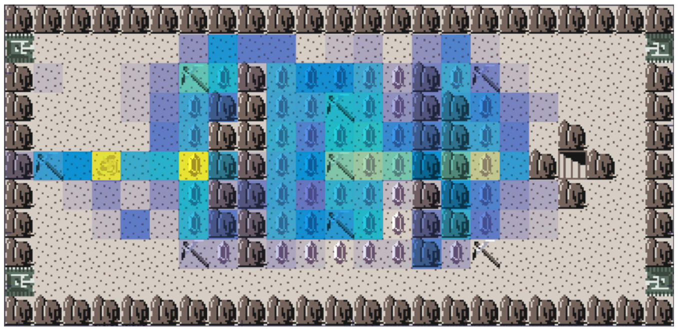
Figure 4: Game B configuration positional heatmap

Despite demonstrating the ability to use multiple MCTS bots with biased heuristics to evolve and assess game parameters for strategic depth, we were not able to validate this measure by connecting it to a difference in the actual experience of human players. One potential problem is that the AI used to evaluate levels did not necessarily correspond well to the potential approaches actually observed and applied by human players, leading to levels that AI was able to effectively approach in multiple ways, but which in practice human players did not. A more fundamental problem, however, was the lack of engagement human players had to the manipulation. The goal of the game was to maximize the score in each level, and in both versions of the game, the potential paths to the goal were constant, but the ways in which score could be maximized were changed - in particular, with the presence of absence of a time bonus that reward completion speed as well as item collection. Despite a prominent score display and instruction that the goal was to escape with as many points as possible, the overwhelming majority (21 out of 25) reported either paying no attention to the score while they were playing. Thus, it appears that the majority of players' strategies were based on factors other than the maximization of score and as such were not affected as intended by the manipulation.

We were also not able to find a difference in player enjoyment between the two versions. This is again likely due to the fact that participants were largely unaffected by the difference in scoring rules between the two versions. In order to properly test the effect of increased choice and strategic depth upon player enjoyment, it will be necessary to ensure that players are actually aware of the choices they must make.

C. Positional analysis

In addition to the data gathered from the player surveys, logs with action and positional information were collected. The positions of every player were combined to produce the heatmaps for each level version, which are presented in Figures 3 and 4. Much like with the player level preference, the positional information differences overall proved to be insignificant. The players did appear to explore more of the level and leave more of the coins untouched in the configuration A, which could be attributed to this version having faster tanks, forcing the players to be more cautious when entering their lanes.

D. Interview analysis

The quantitative results were mirrored in interview responses: not but a small minority of participants noticed any difference between the two versions of the game. When asked about the score, only 2 of the respondents stated that they were aware of it, with the majority of the participants not paying attention to it. Some of the participants stated that it *felt* good to collect coins, and that the sound was pleasant. Nevertheless, other participants enjoyed the game aesthetics and audio as well, but reported the controls were hard to master and led them to concentrate on not colliding with one of the tanks. However, some users reported the game to be easy or lack challenge, wanting more hazards to take into account. These responses indicate that the differences between the games were hard to notice, as well as the score. A better user interface design, with the score clearly stated in the screen would allow participants to potentially notice better the state of the score. Regarding controls, even thought the improvements in the aesthetics seemed to have an impact in the Game Feel, it was not good enough for certain players that were complaining about controls, conceivably interfering with the Player Experience and avoiding users to engage with the game [11], [12], [13].

VI. Conclusion

In this paper, the concept of strategic depth is introduced. Strategic depth is the ability of a level to be played by multiple AI agents biased towards different strategies, and for multiple such agents to be equivalently effective. This technique can be used both to assess a game's strategic depth as well as to evolve a design space to increase or decrease strategic depth.

To this extent, a simple Evolutionary Algorithm (EA) was used to adjust the parameter values of a Zelda inspired puzzle game. 3 different heuristics were applied to a Monte Carlo Tree Search agent and their performance in the evolved games maximized by the EA, while the performance of the default heuristic and a very simple repetitive agent was minimized for a successful evolutionary process.

However, it remains unclear how or if this affects the actual experience of the game when played by humans. There are a number of possible explanations for the strategies defined by heuristic AI play not translating into real-world differences in strategy, but the primary problem was the fact that the majority of players were not motivated by score optimization and so did not alter their strategies between the high- and low-strategic depth versions of the game. In order to properly validate this measurement as representation of a meaningful dimension in the game design space when considering real-world experience, it will be necessary to replicate the experiment with a manipulation of the strategic depth of an objective that more participants are fully conscious of. This may be accomplished either through a different manipulation, such as of survival or ability to complete the level, or by making score a more prominent feature - for instance, through the use of a participant leaderboard.

A second issue that was identified was the potential difference between human and AI strategies. If the approaches taken by the AIs being used to evolve for or assess strategic depth do not reasonably line up with the approaches a human player would take, the strategic depth of a level as measured by such a process will not correspond to the actual strategic depth experienced by human players of the same level. In particular, all of the AIs used in this investigated were MCTS bots with simple heuristic biases and as such do not correspond well to human strategies that are complex or goal- rather than reward oriented. When expanding upon this work, therefore, it would be helpful to consider other heuristics and AI approaches that more closely portray human player strategies.

The EA used for game tuning could be improved as well, looking not only at better mutation operators, but also increasing the resampling rate for more accurate fitness evaluations in the context of multiple stochastic agents, as well as running the algorithm for more generations. A better analysis of the search space prior to evolution in order to filter out the unnecessary or unimportant parameters would further enhance results.

The interviews showed that Game Feel could have potentially been an issue while experiencing the game. The *feel* of the game could be improved by performing further extensions to the GVGAI framework and modifying how input is handled, softening the movement of the avatar and allowing the use of a wider variety of input devices, such as joy-pads/game-pads.

Finally, because this study focused only on one game, in order to avoid generalization issues it would be ideal for future work to investigate the applicability of the strategic depth measurement to other games, particularly games with different control schemes for which the heuristic biases presented here would not be directly applicable - for instance, non-avatar-based puzzles.

Acknowledgment

This work was funded by the EPSRC CDT in Intelligent Games and Game Intelligence (IGGI) EP/L015846/1.

References

[1] D. Perez-Liebana, S. Samothrakis, J. Togelius, T. Schaul, S. Lucas, A. Couetoux, J. Lee, C.-U. Lim, and T. Thompson, "The 2014 General Video Game Playing Competition," in *IEEE Transactions on Computational Intelligence and AI in Games*, vol. PP, no. 99, 2015, p. 1.

[2] J. Liu, J. Togelius, S. M. Lucas, and D. P. Liébana, "Evolving Game Skill-Depth using General Video Game AI Agents," in *Proceedings of the Congress on Evolutionary Computation*, 2017.

[3] K. O. S. Julian Togelius, Georgios N. Yannakakis and C. Browne, "Search-based Procedural Content Generation: A Taxonomy and Survey," in *IEEE Transactions on Computational Intelligence and AI in Games (TCIAIG)*, vol. 3, no. 3, 2011, pp. 172–186.

[4] D. Perez, J. Togelius, S. Samothrakis, P. Rohlfshagen, and S. M. Lucas, "Automated Map Generation for the Physical Traveling Salesman Problem," *IEEE Transactions on Evolutionary Computation*, vol. 18, no. 5, pp. 708–720, Oct 2014.

[5] A. B. Safak, E. Bostanci, and A. E. Soylucicek, "Automated Maze Generation for Ms. Pac-Man Using Genetic Algorithms," in *International Journal of Machine Learning and Computing*, vol. 6, no. 4, 2016, pp. 226–240.

[6] A. Isaksen, D. Gopstein, and A. Nealen, "Exploring Game Space Using Survival Analysis," in *Proceedings of the 10th International Conference on the Foundations of Digital Games, FDG 2015, Pacific Grove, CA, USA, June 22-25, 2015*, 2015.

[7] J. Togelius and J. Schmidhuber, "An Experiment in Automatic Game Design," in *2008 IEEE Symposium On Computational Intelligence and Games*, Dec 2008, pp. 111–118.

[8] K. Kunanusont, R. D. Gaina, J. Liu, D. P. Liébana, and S. M. Lucas, "The N-Tuple Bandit Evolutionary Algorithm for Game Improvement," in *Proceedings of the Congress on Evolutionary Computation*, 2017.

[9] C. Pedersen, J. Togelius, and G. N. Yannakakis, "Modeling player experience for content creation," *IEEE Transactions on Computational Intelligence and AI in Games*, no. 1, pp. 54–67.

[10] L. Nacke and C. A. Lindley, "Affective Ludology, Flow and Immersion in a First- Person Shooter: Measurement of Player Experience." [Online]. Available: https://arxiv.org/ftp/arxiv/papers/1004/1004.0248.pdf

[11] P. Cairns, "Engagement in Digital Games," in *Why Engagement Matters: Cross-Disciplinary Perspectives of User Engagement in Digital Media*. Springer, 2016, pp. 81–104.

[12] S. Swink, *Game Feel: A Game Designer's Guide to Virtual Sensation*. Amsterdam et al: Morgan Kaufman, 2009.

[13] G. Dahl and M. Kraus, "Measuring how game feel is influenced by the player avatar's acceleration and deceleration," *Proceedings of the 19th International Academic Mindtrek Conference on - AcademicMindTrek '15*, pp. 41–46, 2015.

[14] D. P. Liebana, S. Mostaghim, and S. M. Lucas, "Multi-objective tree search approaches for general video game playing," in *IEEE Congress on Evolutionary Computation, CEC 2016, Vancouver, BC, Canada, July 24-29, 2016*, 2016, pp. 624–631.

[15] A. Mendes, J. Togelius, and A. Nealen, "Hyper-heuristic general video game playing," in *2016 IEEE Conference on Computational Intelligence and Games (CIG)*, Sept 2016, pp. 1–8.

[16] C. Guerrero-Romero, A. P. Louis, and D. Perez-Liebana, "Beyond Playing to Win: Diversifying Heuristics for GVGAI," in *Proc. of the Conference on Computational Intelligence and Games*, 2017.

[17] J. Levine, S. M. Lucas, M. Mateas, M. Preuss, P. Spronck, and J. Togelius, "General Video Game Playing," in *Artificial and Computational Intelligence in Games, Dagstuhl Follow-Ups*, vol. 6, 2013, pp. 1–7.

[18] T. Schaul, "A Video Game Description Language for Model-based or Interactive Learning," in *Proceedings of the IEEE Conference on Computational Intelligence in Games*, 2013, pp. 193–200.

[19] R. D. Gaina, D. Perez-Liebana, and S. M. Lucas, "General Video Game for 2 Players: Framework and Competition," in *Proceedings of the IEEE Computer Science and Electronic Engineering Conf.*, 2016.

[20] A. Khalifa, D. Perez-Liebana, S. Lucas, and J. T. and, "General Video Game Level Generation," in *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, 2016, p. to appear.

[21] C. Browne, E. Powley, D. Whitehouse, S. Lucas, P. Cowling, P. Rohlfshagen, S. Tavener, D. Perez, S. Samothrakis, and S. Colton, "A Survey of Monte Carlo Tree Search Methods," in *IEEE Trans. on Computational Intelligence and AI in Games*, vol. 4, no. 1, 2014, pp. 1–43.

[22] R. D. Gaina, J. Liu, S. M. Lucas, and D. Pérez-Liébana, *Analysis of Vanilla Rolling Horizon Evolution Parameters in General Video Game Playing*. Cham: Springer International Publishing, 2017, pp. 418–434.

[23] J. Liu, D. P. Liebana, and S. M. Lucas, "Bandit-Based Random Mutation Hill-Climbing," *ArXiv*, 2016. [Online]. Available: http://arxiv.org/abs/1606.06041