Novelty Search for Evolving Interesting Character Mechanics for a Two-Player Video Game

Eirik Høgdahl Skjærseth
NTNU
eirik.skjaerseth@gmail.com

Harald Vinje
NTNU
haraldvinje@gmail.com

Ole Jakob Mengshoel
NTNU & CMU
ole.j.mengshoel@ntnu.no

ABSTRACT

Computational generation of video game content, often referred to as procedural content generation (PCG), holds much promise for generating character mechanics. Character mechanics refers to how characters are allowed to move and behave in a computer game, rather than aesthetics such as graphics and audio. In this paper we study how to generate character mechanics automatically, by means of novelty search. Our results show that some of the auto-generated characters are, by human subjects, perceived as more interesting than built-in game characters.

CCS CONCEPTS

• **Computing methodologies** → *Modeling methodologies*; **Randomized search**; • **Applied computing** → **Computer games**;

KEYWORDS

novelty search, evolutionary algorithms, procedural content generation, video game, fighting game, character mechanics, user studies

ACM Reference Format:
Eirik Høgdahl Skjærseth, Harald Vinje, and Ole Jakob Mengshoel. 2021. Novelty Search for Evolving Interesting Character Mechanics for a Two-Player Video Game. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 2 pages. https://doi.org/10.1145/3449726.3459444

1 INTRODUCTION

Procedural content generation (PCG) is an essential part of computer game development [5]. Search-based methods are often used, mainly evolutionary algorithms [8]. *Novelty search* is a variant of evolutionary algorithms that heavily promotes exploration [2], and has shown promise for PCG [4]. Liapis et al. combine novelty search with a feasible-infeasible two-population genetic algorithm. They propose two variants; *feasible-infeasible novelty search* (FINS) and *feasible-infeasible dual novelty search* (FI2NS).

Evaluating game content with the objective of human player enjoyment by play-throughs with AI agents is commonly performed when a direct evaluation function is hard to define [1, 7]. We refer to this approach, which we use, as *simulation-based evaluation.*

This paper presents SolEA, a PCG system for a two-player video game, Sol [6]. Sol is an open-source, novel two-player video game, inspired by fighting games. It was developed by two of the authors. Each player controls a single character, by moving and using three unique abilities (projectile or melee attacks) that knock the enemy backward. The stage area is bounded by holes and walls, see Figure 1. A character is eliminated upon intersecting a hole. Each character has three lives. Sol has four built-in characters, with different strengths and weaknesses. Two agents (computer controlled players) were designed to play Sol, A_s for simulation-based evaluations and A_t for user studies [6]. Both are rule based reflex agents, with A_s having better attack accuracy than A_t.

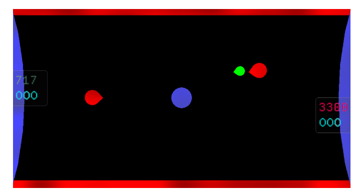

Figure 1: Sol graphics. Characters (red circles), walls (blue), holes (red), ability (green).

Our goal with SolEA is to generate interesting character mechanics through the use of evolutionary algorithms. *Interesting* character mechanics are enjoyed by human players. Specifically, we define interesting as: *novel*, *fun*, and *balanced* (not being superior to another). This definition is closely related to concepts in computational creativity [3].

We consider this research question (**RQ**): Can constrained novelty search yield character mechanics that humans find interesting (novel, fun, and balanced)? Through our case study on generating interesting character mechanics for Sol, we contribute to the use of PCG in video games. Further, we estimate player experience through simulation-based evaluation in Sol. To our knowledge, these two areas have not been integrated in previous research.

2 EVOLUTIONARY ALGORITHMS FOR SOL

A software system, SolEA,[1] was created to generate fighting game characters, with the proposed method [6]. SolEA (Figure 2) consists of two main modules, *Evolution* and *Constraints*. Evolution is based on constrained novelty search. First, an *initial population* of characters is generated. Each character is then evaluated for feasibility in the Constraints module, and based on the outcome, included in either the feasible population X_F or the infeasible population X_I. Novelty search is performed by either FINS or FI2NS, with offspring boost. The search completes after a fixed number of iterations.

The genotype of a character's mechanics is represented by 44 properties describing behaviour. A (physical) radius and movement speed describes the "body" of a character (body properties) and

[1] Source code for SolEA is available at: https://github.com/sol-ai-master/solai_project

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07.
https://doi.org/10.1145/3449726.3459444

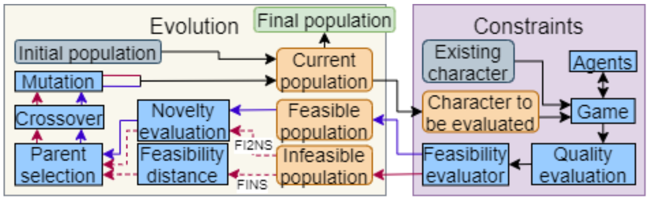

Figure 2: Architecture of SolEA.

three abilities are each given by 14 properties. Thus the genotype $\boldsymbol{x}$ is an n-tuple or sequence of genes $(x_1, \ldots, x_n)$, where $i \in \mathbb{N}$ and gene $x_i \in \mathbb{R}$ for $1 \leq i \leq n$. All properties are upper- and lower bounded. The phenotype is given by a character loaded into Sol.

Constraints are based on a quality measure (fun and balance), given by character measurements over multiple game simulations. An individual is loaded into the game, and played 10 times against each of the four built-in characters. Both characters are played by agent A_s. Five criteria are given by average measurements over simulations: *Game balance (1):* Average number of wins. *Game length (2):* Average length of simulations. *Stage coverage (3):* The area of the stage used. *Character balance (4):* The ratio between hits from the three abilities. *Lead change (5):* Number of times the lead changes, given by state evaluations. A criterion is met if its average measurement falls within a pre-computed *feasibility range* [6].

For novelty search, the novelty of an individual is given by the distance to the k closest individuals, and the m most novel individuals of each generation are included in the novel archive. The distance is computed by a *distance function*: $d(\boldsymbol{x}_1, \boldsymbol{x}_2) \rightarrow \mathbb{R}$. For SolEA, d is defined on the genotype, by the Euclidean distance between $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, not considering ability order.

Two methods for generating the initial population are studied. The *random generator* uniformely samples the search space, while the *heuristic generator* copies existing characters with mutation. Novelty proportionate selection with replacement is used for parent selection. A crossover operator swaps one ability (14 genes) between two individuals, and produces two offspring. A custom mutation operator mutates a gene, x_i, with a probability, p_M. A gene is mutated by sampling from $U(-x_i m_s, +x_i m_s)$, where m_s is the mutation strength.

3 EXPERIMENTAL RESULTS

The experiment is a combination of quantitative analysis of evolution runs and qualitative analysis through human play testing of the outputs from evolution runs. A total of 7 subjects (all students age 20-29) participated in the user studies. Everyone had at least some prior experience with fighting games. The following parameters were used for SolEA: $m = 5$, $k = 15$, $p_M = 0.3$, $m_s = 0.2$ (body properties) or 0.5 (ability properties).

Goal. The goal of the experiment is to test whether constrained novelty search can yield character mechanics that humans find interesting (see **RQ**).

Method and data. After a total of 120 evolution runs, we found no significant difference with regards to feasbility and diversity between FINS and FI2NS, using a random generator or heuristic generator to create the initial population. We sampled the most diverse population of 6 feasible characters from the evolution runs. To evaluate these characters, we compared them to 6 human designed characters. A total of seven human subjects, P_1-P_7, played Sol as each of the 12 characters. The 12 characters were split into three batches, each batch with 4 characters. After playing through each batch, the players ranked, based on their enjoyment, the 4 characters from 1 to 4. Here, 1 is most enjoyable and 4 is least enjoyable. We further asked them to comment on unique or interesting aspects of a character, as well as balance. This left us with feedback regarding the novelty and balance of the characters.

Table 1 shows the character identifier and genesis of each character in the experiment, as well as the three batches of characters. We denote human designed characters C_{hi} and computer generated characters C_{ci}. Each of the three batches contains two SolEA-generated characters and two human-designed characters.

B	C	S	R	B	C	S	R	B	C	S	R
1	C_{c1}	*12*	*2*	2	C_{c3}	10	3	3	C_{c5}	*11*	*2*
1	C_{h1}	9	3	2	C_{h3}	9	4	3	C_{h5}	10	3
1	C_{c2}	**14**	**1**	2	C_{c4}	*11*	*2*	3	C_{c6}	**13**	**1**
1	C_{h2}	8	4	2	C_{h4}	**12**	**1**	3	C_{h6}	8	4

Table 1: The 12 characters (C) used in the experiment, presented according to their batch (B), score (S), and rank (R) within its batch.

Results and discussion. The performance of the characters, based on the responses from the human subjects P_1-P_7, is summarized in Table 1. Two characters, generated by SolEA, were pointed out by several subjects to be intriguing and quite different from the other characters, namely C_{c4} and C_{c6} (see Table 1). C_{c4} had one ability that was especially interesting, as pointed out by the subjects. It was a projectile ability with a relatively large size, really slow speed and long persistance. The subjects noted that this ability could be used to "lay mines" that could be used to trap the opponent, see Figure 3.

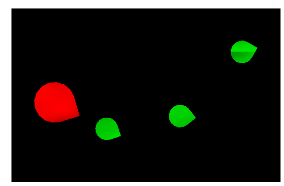

Figure 3: Character C_{c4} (red), generated by SolEA, along with 3 "mines" (green).

REFERENCES

[1] C. Browne and F. Maire. 2010. Evolutionary Game Design. *IEEE Transactions on Computational Intelligence and AI in Games* 2, 1 (2010), 1–16.

[2] J. Lehman and K. O. Stanley. 2011. Abandoning Objectives: Evolution Through the Search for Novelty Alone. *Evolutionary Computation* 19, 2 (2011), 189–223.

[3] A. Liapis, G. N. Yannakakis, and J. Togelius. 2014. Computational Game Creativity. In *ICCC*.

[4] A. Liapis, G. N. Yannakakis, and J. Togelius. 2015. Constrained Novelty Search: A Study on Game Content Generation. 23, 1 (2015).

[5] N. Shaker, J. Togelius, and M. J. Nelson. 2016. *Procedural content generation in games*. Springer.

[6] E. H. Skjærseth and H. Vinje. 2020. *Evolutionary algorithms for generating interesting fighting game character mechanics*. Master's thesis. NTNU, NTNU 7491 Trondheim, Norway. https://hdl.handle.net/11250/2689497

[7] J. Togelius and J. Schmidhuber. 2008. An experiment in automatic game design. In *2008 IEEE Symposium On Computational Intelligence and Games*. 111–118.

[8] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne. 2011. Search-Based Procedural Content Generation: A Taxonomy and Survey. *IEEE Transactions on Computational Intelligence and AI in Games* 3, 3 (July 2011), 172–186.