SELF EVOLVING PLATFORMER

Sarthak Sharma[1], Shreyash Srivastava[1], Rahul Tripathi[1], Samanvitha S[1] and K Deepashree[2]

1 Student, Department of CSE, Dayananda Sagar Academy of Technology and Management, Bangalore, India
2 Assistant Professor, Department of CSE, Dayananda Sagar Academy of Technology and Management, Bangalore, India

ABSTRACT

Self-Evolving Platformer depicts an Artificial Intelligence model of a 2 Dimension Platform Level Generator using various techniques and algorithms. In addition to the model, we aim to create a bot that will be able to navigate through the AI-Generated level to test the viability of the level. The object elements of each level will draw inspiration from input images using which it will identify a general theme and setting. To implement this part we will require the use of a sentiment classification algorithm that will detect if the inspiration image is an obstacle or a collectible.

Keywords: - *GAN, PCG, Platformer, Evolving*

1. INTRODUCTION

Video games are evolving drastically and their definition changes with every new title released. At its core, it is a platform or a piece of hardware with a variety of input methods such as controllers, joysticks and many more. Video game consoles are all-in-one integrated devices which can play demanding games and can be mass-produced by companies. The gaming industry is merely a specialized business for a specific clientele. The expansion of the gaming sector can be credited to the increasing popularity of mobile gaming and advancements in the required hardware, leading to gaming becoming a more universally embraced means of amusement. The industry's revenues have seen significant increases in recent years, going from approximately $9.5 billion in 2007 to $11.7 billion in 2008 and $25.1 billion in 2010. Here, game developers, designers, and artists work creatively to create the game. The gaming industry is no longer a specialised business for a certain consumer group. Due to advancements in mobile gaming and enhancements in the required hardware, gaming has emerged as a feasible source of entertainment accessible to individuals of diverse ages and backgrounds. In this paper we proposed a method to generate the game levels from a real-life object as the input, the input can be an image and the system will categorise the object as a collectible or an obstacle.

2. LITERATURE SURVEY

In recent years, the field of game development has witnessed remarkable advancements in the realm of procedural content generation (PCG). This technique, which involves the automatic creation of game content, has proven to be a valuable tool for enhancing player experiences, improving replayability, and reducing development costs. Within this context, this literature survey aims to explore the first plan of action that comes to mind in developing a platform for a Mario-like game that can generate game elements autonomously.

The fundamental plan entails utilizing a combination of rule-based algorithms and machine learning approaches to empower the platform with the ability to generate diverse and engaging game elements. First and foremost, the platform would employ rule-based algorithms to establish the core mechanics of the game, such as the movement

and physics systems. By adhering to the established rules, the generated elements will seamlessly integrate with the existing gameplay mechanics, ensuring a coherent and immersive experience for the players.

Additionally, the platform would leverage machine learning techniques, particularly generative models, to dynamically create levels, enemy behaviour, and power-up placements. This approach would involve training the model on a vast dataset comprising existing Mario-like games, allowing it to learn the underlying patterns, structures, and aesthetics of successful game elements. By analyzing this dataset, the generative model can then generate novel and challenging levels, unique enemy behaviours, and strategically placed power-ups, tailored to the player's skill level and preferences.

A method for procedurally creating video game levels using generative adversarial networks (GANs) is presented in the paper [1]. The authors suggest utilising the MAP-Elites optimization approach, which is intended to effectively explore a GAN's latent space and extract levels that differ across several predetermined gameplay metrics. The method is evaluated by using it to generate levels for the game Super Mario Bros and comparing the resulting levels to human-authored ones. Furthermore, the researchers carried out an online user study to analyze the impact of various mechanics employed in participants' subjective assessments of the mechanically created stages on their apparent complexity and aesthetic attractiveness. The authors of this paper are proposing a method for using (GANs) to procedurally generate levels for video games in a way that allows human designers to specify certain attributes or characteristics that they want the generated levels to have. They call this problem "latent space illumination" (LSI). The authors propose using a class of optimization algorithms called quality diversity (QD) algorithms to solve the LSI problem. These algorithms are intended to uncover a wide variety of exceptional responses in continuous search areas. The authors compare the performance of several state-of-the-art QD algorithms on this problem, including MAP-Elites, MAP-Elites (line), and CMA-ME (Covariance Matrix Adaptation MAP-Elites). They discover that these approaches vastly surpass competing techniques, such as random search and CMA-ES (Covariance Matrix Adaptation Evolution Strategy), in uncovering various high-quality levels while preserving stylistic closeness to human-authored instances. Specifically, the researchers discover that CMA-ME exhibits superior performance compared to the other algorithms evaluated, demonstrating greater diversity and superior quality in the generated levels. Additionally, the authors undertake a user study to investigate the impact of the various mechanics employed in the generated levels on subjective ratings concerning their difficulty and appearance. The study reveals that the incorporation of a diverse set of level mechanics yields distinct subjective ratings, thereby underscoring the potential of QD algorithms to generate a wide range of high-quality content by exploring the latent space of GANs. This paper gives us the baseline to conceptualize the working model of the Platformer game where real-life images can be used as the game objects.

The paper [2], presents a method for using generative adversarial networks (GANs) to generate realistic game elements. The researchers devised their approach by incorporating GAN enhancement techniques and Adaptive Discriminator Augmentation and assessed the quality of the produced images using the Fretchet Inception Distance (FID) metric alongside a visual evaluation conducted by 50 observers. The outcomes indicated that "69% of the generated icons were perceived as authentic, comparable to the 70% for hand-drawn images". These findings imply that the generated illustrations possess high quality and are practically indistinguishable from manually created graphics. Moreover, this capability holds the potential to substantially reduce asset creation expenses for designers while enabling the creation of unique experiences for each player through the utilization of a vast array of generated visuals.

Generating whole new levels can be a bit tricky part. The paper [3] presents a method for generating procedural content generators (PCGs) that can be used to automatically generate levels for the video game Super Mario Bros. A PCG, which stands for Procedural Content Generation, refers to a software tool utilized for the automatic creation of various game elements, including levels, rules, textures, and items. In their proposal, the authors suggest the adoption of an interactive evolutionary algorithm for evolving agent-based level generators. These generators are essentially PCGs meticulously tailored to produce levels specifically designed for the Super Mario Bros game. The

human user is responsible for making aesthetic judgments about which generators to prefer, based on views of the generated levels and estimates of their playability. The writers examine the properties of the developed levels and how equivalent or diverse the levels and generators are from one another. The PPLGG uses a genetic algorithm to evolve a population of inner generators, which are collections of agents. Each agent has a set of attributes and behaviours that allow it to move around the level and modify it by adding or removing blocks. The agents move independently and concurrently, and their behaviour is not deterministic, allowing for a wide range of possible levels to be generated. The genetic algorithm uses a fitness function to evaluate the playability of the generated levels and to determine which generators should survive and be used as parents for the next generation. The PPLGG also includes a graphical user interface (GUI) that allows a user to select and visualize the generated levels in different ways, including a "cloud map" that shows the average level generated by each generator, a playability rating, and sample levels that can be played within the tool. The user can also export the generated generators for use in other runs of the algorithm.

In the paper [4] the authors present an approach to generate levels for a platformer game, like Super Mario Bros, using three levels of abstraction: micro-patterns, meso-patterns, and macro-patterns. Micro-patterns encompass specific design elements within a game, such as a particular type of platform. Meso-patterns, on the other hand, constitute combinations of micro-patterns that give rise to more extensive structures, such as a sequence of platforms arranged in a specific manner. Macro-patterns further extend this concept by combining meso-patterns to form even larger structures, such as a series of platforms followed by a succession of enemies. In this particular approach, micro-patterns serve as fundamental building blocks within a search-based procedure aimed at identifying macro-patterns that emulate the macro-patterns that have been seen in a few layers of input. The authors previously created a framework for analyzing 2D platformer game levels., which involves categorizing the content into three hierarchical levels of abstraction: micro-patterns, macro-patterns as well as meso-patterns. Vertical slices of the level are represented by micropatterns, aspects requiring player interaction are represented by mesopatterns, and the placement of power-up mushrooms is represented by macro patterns. The objective of this approach is to generate levels that mimic the style of human-designed levels while retaining a diverse range of content. The resulting levels effectively replicate the macro-patterns of the chosen input levels, thereby showcasing an automated method for analyzing and replicating style in level design, as argued by the authors.

3. CONCLUSIONS

We have proposed a method for generating game levels using real-life objects as input, to categorize the objects as collectibles or obstacles. The authors also reviewed previous research on using generative adversarial networks and optimization algorithms to procedurally generate levels for video games. The performance of various state-of-the-art quality diversity algorithms was compared, with CMA-ME found to be the most effective in finding diverse, high-quality levels that maintained stylistic similarity to human-authored examples. A user study was also conducted to examine the effects of different level mechanics on subjective ratings of difficulty and appearance. The results showed that the automatically generated levels received similar ratings to human-authored levels, indicating the potential for this method to be used in game development.

4. REFERENCES

[1] Matthew C. Fontaine1 et al - Illuminating Mario Scenes in the Latent Space of a Generative Adversarial Network. 21 Jun 2021.

[2] Karp, Rafał & Swiderska, Zaneta. (2021). Automatic generation of graphical game assets using GAN. 7-12. 10.1145/3477911.3477913.

[3] Kerssemakers, Manuel & Tuxen, Jeppe & Togelius, Julian & Yannakakis, Georgios. (2012). A procedural level generator generator. 2012 IEEE Conference on Computational Intelligence and Games, CIG 2012. 335-341. 10.1109/CIG.2012.6374174.

[4] Dahlskog, Steve & Togelius, Julian. (2014). A Multi-level Level Generator. 10.1109/CIG.2014.6932909.

[5] Awiszus, M.; Schubert, F.; and Rosenhahn, B. 2020. TOADGAN: coherent style level generation from a single example. In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, volume 16, 10–16.

[6] Manuel Au-Yong-Oliveira, Ramiro Gonçalves, José Martins, and Frederico Branco. 2018. The social impact of technology on millennials and consequences for higher education and leadership. Telematics and Informatics 35, 4 (2018), 954–963.

[7] L. Johnson, G. N. Yannakakis, and J. Togelius, "Cellular Automata for Real-time Generation of Infinite Cave Levels," in Proceedings of the ACM Foundations of Digital Games. ACM Press, June 2010.

[8] J. Togelius, N. Shaker, and M. J. Nelson, "Introduction," in Procedural Content Generation in Games: A Textbook and an Overview of Current Research, N. Shaker, J. Togelius, and M. J. Nelson Eds. Springer, 2014