Proceedings of CONF-MLA 2025 Symposium: Intelligent Systems and Automation: AI Models, IoT, and Robotic Algorithms
DOI: 10.54254/2755-2721/2025.LD25495

The Role of AI in Evolving 3D Game Worlds: From NPC Autonomy to Content Generation

Yebo Wen

Guanghua Cambridge International School, Shanghai, China
Allen317609@gmail.com

Abstract. This paper focuses on applications that enhance non-player character autonomy, intelligent opponent design, and content generation: it studies the role of artificial intelligence within 3D games. Some AI technologies, such as reinforcement learning, deep learning, and GANs, have made a substantial difference in NPC realism and interactivity; NPCs have become adaptive to game environments and different player behavior. AI-driven opponents have vastly improved the development of more dynamic and challenging gameplay. These technologies have allowed for the creation of NPCs that not only react intelligently but also evolve based on the gameplay experience. Furthermore, AI-powered content generation can change the process of game world creation now, supporting the potential of procedurally creating diversified environments, game quests, and narratives.This paper reflects on the evolution of these AI applications in 3D games and discusses potential future directions, including the development of personalized AI systems that can tailor gameplay to individual preferences, and multi - platform AI solutions that enhance cross - platform gaming experiences, making games more accessible and immersive for global users.

Keywords: NPC Autonomy, Intelligent Opponents, Procedural Content Generation (PCG), Generative Adversarial Networks (GANs)

1. Introduction

In the past decades, computer games have undergone rapid developments, especially in the field of 3D computer games (3Dgames). With the development of artificial intelligence (AI), the visual effects, gaming experience, game complexity and game interactions of 3Dgames have received significant progress [1]. AI has played crucial roles in these fields, especially in terms of non-playable characters (NPC), AI opponents and AI content generation [2]. This paper will discuss how AI has helped in 3Dgames by focusing on three aspects: AI applications in NPC autonomy-Analyzing how AI raises the intelligence and personalized interactivity of NPCs by using reinforcement learning and deep learning; Intelligent opponents and Dynamic gaming experience the application of AI in the design of intelligent opponents, especially for Dynamic Difficulty Adjustment (DDA); AI-driven content generation and world construction-Analyzing how is AI used in game content constructing and the importance of AI in generating dynamic game map and game quest. By focusing on NPCs, opponents, and content generation, this study highlights how AI can

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2025.LD25495

not only improve gameplay realism but also enhance player satisfaction and challenge. Additionally, the paper explores the future implications of AI-driven systems in making games more personalized and adaptable, pushing the boundaries of interactive entertainment. Also, this research contributes to advancing AI technology within the gaming industry, providing insights into the ongoing evolution of game design and its impact on user experiences.

2. AI applications in NPC autonomy

2.1. The basics of NPC behavior and decision-making

In early computer games, most of the behavior of NPCs is achieved by Finite State Machines (FSMs) and other simple rule systems. FSMs operate by transitioning between a finite set of states based on inputs, with each state representing a specific condition of the system [3]. These systems provide the basic logic for NPCs, for example, the NPCs in the game DOOM [1], the NPCs were able to switch states between attaching, escaping, and patrolling by using PSMs. However, these systems lack adaptability and flexibility when receiving unclassified inputs as the behavior after receiving input is pre-determined and limited.

The introduction of Behavior Trees (BT) quickly fixed the problem and has become a milestone in the field. A BT organizes behaviors hierarchically through nodes, each representing conditions, actions, or control flows. The main advantages of BTs are their modularity and reactivity, allowing complex systems to be built by combining simple tasks [4], for example, in the game The Last of Us (2013), the NPCs were able to change their behaviors by judging the change of environments and the movements of players, like hiding and calling backups [1].

2.2. Modern AI-driven NPCs

In recent years, Reinforcement Learning (RL) and Deep Learning (DL) have developed significantly and are widely used in NPC autonomy. These technologies enable NPCs to learn and optimize their behavior through interaction with the environment, to achieve more complex and intelligent reactions.

RL guides NPCs to gradually optimize their behavior strategies through a reward mechanism. For example, RL can make optimized actions in different situations after learning from many failed attempts [5]. This makes NPCs able to dynamically adjust NPC behavior in real-time game environments, enabling adaptation to different player styles, which provides a more challenging gaming environment.

Research by Georgios N. Yannakakis and Julian Togelius [1] shows that RL and DL applications have effectively increased the complexity of NPCs: in the game Red Dead Redemption 2, NPCs was not only adaptive to environments but also was able to exhibit complex emotions and social interactions.

2.3. Ways to improve

This part gives three potential thermotical ways of improving performance, adaptation, and more customized NPCs content generation.

A more efficient AI model: Developing a more efficient and specialized AI model would increase the performance of NPCs with limited computing resources. This may involve more advanced deep learning techniques or optimization algorithms based on reinforcement learning. Furthermore, as NPCs in FPS games rarely involve text analysis, the model should focus on image analysis. Hence

DOI: 10.54254/2755-2721/2025.LD25495

an available improvement could be to develop separate AI models for distinct types of NPCs depending on their functions. An AI model for FPS NPCs should develop towards pure Convolutional Neural Network (CNN), which is specifically designed to analyze image content. On the other hand, NPCs in Role Play Games (RPG) should use a language model.

Cross-platform AI models: Doing research on achieving consistent and efficient NPC autonomy in different platforms which have different computing environments and resources, like personal computers (PC), cell phones and play station consoles, could bring more fair gaming experience for different consumers. One conceivable way is to place the process of AI on to external servers to reduce the pressure on clients. However, this may demand a high-quality internet environment for the consistency of data transfer.

The greater integration of AI with storytelling: Studying how to make AI-driven NPCs integrate more with the lore of the game, for example more complex emotion effects in line with the different stages of the game lore or dynamically generate new lores based on the different gaming style for different players (Which will be discussed in following sections).

3. Intelligent opponents and dynamic gaming experience

Like NPCs in the early years, AI opponents (BOTs) were also constructed by simple sets of rules, the reactions of BOTs are limited and fixed for each individual input and cannot adapt to the change of style of players. For instance, in the classical Quake series of games, the movement passes, and the attach patterns are pre-programmed, which lacks complexity and flexibility. This may pose some gaming challenges to gamers in certain ways; however, after making attempts and working out countermeasures, gamers may discover the patterns of the BOTs, which will reduce the challenge and enjoyment of the game. Although the introduction of the FSMs and Behavior tree in the early 20 centuries do make the behavior of AI opponents more intelligent and flexible, they still lack the self-adjustment to different playing styles.

With the development of RL, Evolutionary Algorithms (EAs) and Dynamic difficulty Adjustment (DDA). AI opponents have arrived at another level. These technologies enable opponents to learn and optimize their behavior strategies, exhibiting more complex and intelligent responses. In this section, we will focus on three types of AI technology used by AI opponents.

Reinforcement learning and AI opponent: As BOTs are a subset of NPCs, they have many similarities. For example, the basic rules are applied to both, and both generate their output by analyzing the rules and the events occurring during gameplay. However, while there are two models suitable for training NPCs, it seems that RL is better than DL when it comes to training an AI opponent. This is because NPCs for dialogues and simple movements can be trained, as they face fewer game situations, but AI opponents must consider the unpredictable or more complex situations which exceeded their knowledge gained form gaming, Therefore, RL is also suitable for developing an AI opponent, because RL base on trials and errors, where the AI interacts with an environment and produce an optimized solution. This fits the characteristics of AI opponents which involves decision making and control problems.

Evolutionary Algorithms (EAs) are optimization methods inspired by the natural evolution of animals. It finds the optimal solutions by simulating mechanisms such as natural selection and genetic variation in biological evolution. EAs start with a group of potential solutions to a problem, each represented as individuals. These solutions then undergo a fitness function which generates a fitness value as the solution to the problem. Those solutions with higher fitness are likely to be selected by the algorithm to conduct crossovers (recombination) and mutations. Crossovers combine the aspect of two parent solutions to create offspring (newer and better solutions). Mutation

DOI: 10.54254/2755-2721/2025.LD25495

introduces random changes to ensure the diversity of the species (solutions). By increasing the diversity and the generations of the solution, the population evolves towards better solutions, effectively optimizing the solution to the problem [6]. EAs are particularly useful in solving complex, multi-dimensional problems in AI. Research by Marc Ponsen shows that the application of EAs in the game Wargus has enabled AI to gradually adapt to the different game tactical scenarios and adjust strategies [7].

Dynamic Difficulty Adjustment: Dynamic difficulty Adjustment (DDA) is commonly used in modern computer games. DDA aims to adjust the game difficulty in real-time based on the performance of the players, ensuring the balance of the game difficulty. DDA systems typically monitor the behavior values such as kills & death (KD) and the time for completing game tasks, to change the intelligence level of BOTs and NPCs or other game features.

4. AI-driven content generation and world construction

The concept of content generation was introduced in the late 1970s and early 1980s when computing power was limited and required developers to generate huge game worlds through procedural methods. Early Procedural Content Generation (PCG) was commonly used in creating random mazes and simple game levels, such as the game Rogue [8] and Elite [9]. These games provide different gaming experiences in every different game with simple PCG algorithms. For example, the game Rogue (1980) used a random number generator and applied to game parameters to create mazes. This method not only saves computer memory but also brings a diverse gaming experience.

In the 1990s, with the enhancement of computing power, PCG technologies were gradually applied to more complicated tasks and developers could create more detailed and dynamic game worlds [10]. For instance, in the game Diablo [11], the procedurally generated dungeons and loot systems greatly enhanced the replay-ability of the game. There were four popular algorithms at that time.

Georgy Voronoi introduced the concept of the Voronoi Diagram, a method of partitioning a plane based on the distance to a specific set of points. For a given set of seed points, the plane is divided into regions where each region contains all the points closest to one seed point. The boundaries between these regions, known as Voronoi edges, have equal distances from two or more seed points. Voronoi diagrams have various applications, including computer graphics, geophysics, and game development [12].

Fortune's Algorithm is an efficient sweep line algorithm for constructing Voronoi diagrams, operating in $O(n \log n)$ time, where n is the number of points. The algorithm works by sweeping a vertical line across the plane and maintaining a data structure that represents the beach line (the boundary of the current Voronoi regions). As the sweep line moves, the algorithm dynamically updates the beach line and the Voronoi diagram [13].

In the 1980s, Ken Perlin introduced "Perlin Noise" [14], and "Perlin Worms" are a specific application of this concept. Perlin Worms use noise functions to generate paths or tunnels, typically for cave generation, which became more prominent with the development of procedural generation techniques after 1990.

Simplex noise, introduced by Ken Perlin in 2001, is an improved version of Perlin Noise. It addresses some limitations of Perlin Noise, such as inefficiencies in higher dimensions and visible directional artifacts. Simplex noise uses a simpler, faster gradient-based approach that scales better in higher dimensions, making it particularly useful in procedural texture generation, terrain generation, and simulations.

DOI: 10.54254/2755-2721/2025.LD25495

As AI technology develops in recent years, especially for reinforcement learning and deep learning, the generating abilities of PCG algorithms are enhanced. The range of generation expands to quests, characters, narratives, and even the construction of entire worlds. One of the popular AI models is Generative Adversarial Networks which could create contents that are very alike to real world.

4.1. Generative Adversarial Networks

Generative Adversarial Networks (GANs) were introduced by Ian Goodfellow et al. in 2014 [15], this deep learning model was designed for generating realistic data, such as images, videos, and texts. In computer games, GANs could be used to generate realistic terrain, building, and character design.

GANs consist of two neural networks, a Generator, and a Discriminator. These networks are trained in a competitive and adversarial process.

In the process of GANs, the Generator will receive a random noise vector sampled from a normal or uniform distribution. It then processes this noise to generate synthetic data samples that resemble real-world data. The goal of doing this is to generate data that is hard to distinguish from real-world data, to try 'lying' and 'fooling' the Discriminator.

Once the Generator has created the fake samples, the Discriminator will try to compare and output a probability which indicates whether the input data is real or 'fake.' The goal of the Discriminator is to correctly classify the real and fake samples as accurately as possible.

The process of Adversarial Training could be regarded as a game between the Generator and the Discriminator. The goal of the training is to find a balance point between the Generator and Discriminator, where the samples generated by the Generator are statistically indistinguishable from real data.

Loss function: The typical loss function for GANs is a minimax objective:

$$\min_G \max_D V(D, G) = E_{\{x \sim p_{\{data\}(x)}\}}[\log D(x)] + E_{\{z \sim p_{z(z)}\}}[\log(1 - D(G(z)))] \tag{1}$$

$$\min_G \max_D V(D, G) \tag{2}$$

This term represents the adversarial game between the Generator (G) and the Discriminator (D).

$$E_{\{x \sim p_{\{data\}(x)}\}}[\log D(x)] \tag{3}$$

The Discriminator DDD aims to maximize this term, which is the average log probability of correctly classifying real data xxx.

$$E_{\{z \sim p_{z(z)}\}}[\log(1 - D(G(z)))] \tag{4}$$

The Discriminator also aims to maximize this term, which is the average log probability of correctly identifying fake data $G(z)$ (generated from noise z) as fake.

$$p_{\{data\}(x)} \tag{5}$$

Represents real data distribution.

DOI: 10.54254/2755-2721/2025.LD25495

$$p_{z(z)} \tag{6}$$

Represents the noise distribution used by the Generator to create fake data.

This adversarial process will continue until the Generator produces data that is indistinguishable from real data. At this point, the Discriminator can no longer distinguish between real and fake samples, theoretically leading to an optimal solution where the generated data distribution fits the real data distribution.

5. Conclusion

This paper has reviewed the role of AI in 3D gaming, focusing on such areas as NPC autonomy, intelligent opponents, and content generation. Based on the research presented, it becomes clear that AI has driven both the realism and interactivity of NPCs significantly. Technologies like reinforcement learning and deep learning have empowered NPCs to adapt dynamically to player behavior and changes in the environment, hence improving gameplay immersion. Additionally, AI techniques like reinforcement learning and evolutionary algorithms ensure game unpredictability, keeping the gamer engaged through challenging plays. Also, AI-driven procedural content creation and generative adversarial networks have also changed the very paradigm of the game world, quest, and narrative design and have opened avenues for dynamic and personalized game environments.

This study shows that AI is no longer just a tool for basic game mechanics but has become an integral part of how games are designed and experienced. However, at the same time, despite advances in AI, several challenges remain; for example, AI models still rely heavily on computational resources, and more efficient algorithms must be developed to optimize the behaviors of NPCs without overburdening hardware. On the other hand, AI-driven world generation may create diverse and large worlds, whereas in most cases, present systems are limited to rules predefined and not always as deep or coherent to fully support dynamic storytelling. There is also better integration of AI and game lore where the emotional response of an NPC and their actions would flow from the developing storyline.

Further research is suggested to increase the adaptability of these systems on other platforms and in different environmental conditions so that AI gaming experiences will be accessible to more types of players. Furthermore, with further development, AI is able to merge the techniques of machine learning with narrative design for even more personalized and context - sensitive interactions in games. Besides, research into hybrid models, combining RL with GANs and other AI technologies, can enable pure autonomous game worlds where NPCs react to the player's actions but also grow and adapt to the long-term unfolding of the story of the game.

Indeed, as AI technology advances even further, it will likely play a much bigger role in 3D gaming moving forward, which will result in more personal, fully immersive experiences on the part of the player.

References

[1] Yannakakis, G. N., & Togelius, J. (2018). Artificial intelligence and games (Vol. 2, pp. 2475-1502). New York: Springer.

[2] Hubble, A., Moorin, J., & Khuman, A. S. (2021). Artificial intelligence in fps games: Npc difficulty effects on gameplay. In Fuzzy Logic: Recent Applications and Developments (pp. 165-190). Cham: Springer International Publishing.

[3] LaMeres, B.J. (2024). Modeling Finite State Machines. In: Quick Start Guide to Verilog. Springer, Cham. https://doi.org/10.1007/978-3-031-44104-2_8

[4] Colledanchise M, Ögren P. Behavior trees in robotics and AI: An introduction [M]. CRC Press, 2018.

DOI: 10.54254/2755-2721/2025.LD25495

[5] Dobrovsky, A., Borghoff, U. M., & Hofmann, M. (2019). Improving adaptive gameplay in serious games through interactive deep reinforcement learning. Cognitive infocommunications, theory and applications, 411-432.
[6] Wong, K. C. (2017). Evolutionary algorithms: Concepts, designs, and applications in bioinformatics. In Nature-Inspired Computing: Concepts, Methodologies, Tools, and Applications (pp. 111-137). IGI Global.
[7] Ponsen, M., & Spronck, P. (2004). Improving adaptive game AI with evolutionary learning (Doctoral dissertation, Masters Thesis, Delft University of Technology).
[8] Yannakakis, G. N., & Togelius, J. (2011). Experience-driven procedural content generation. IEEE Transactions on Affective Computing, 2(3), 147-161.
[9] Hendrikx, M., Meijer, S., Van Der Velden, J., & Iosup, A. (2013). Procedural content generation for games: A survey. ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 9(1), 1-22.
[10] Li, S., He, H., & Wang, Z. (2010, December). Simulation of the temperature field of annulus multiphase flow in foam drilling. In The 2nd International Conference on Information Science and Engineering (pp. 1507-1511). IEEE.
[11] Betts, T. (2014). Procedural content generation. Handbook of Digital Games, 62-91.
[12] Aurenhammer, F., & Klein, R. (2000). Voronoi Diagrams. Handbook of computational geometry, 5(10), 201-290.
[13] Fortune, S. (1986, August). A sweepline algorithm for Voronoi diagrams. In Proceedings of the second annual symposium on Computational geometry (pp. 313-322).
[14] Perlin, K. (1985). An image synthesizer. ACM Siggraph Computer Graphics, 19(3), 287-296.
[15] Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., ... & Bengio, Y. (2014). Generative adversarial nets. Advances in neural information processing systems, 27.