Enthusiasms, Challenges, and Future Trends in Procedural Content Generation in Games Using Machine Learning

Shaqayeq Saffari[1], Morteza Dorrigiv[1]* and Farzin Yaghmaee[1]

***Abstract*-- Procedural Content Generation (PCG) in video games, defined as the automated creation of game content through algorithmic processes, has experienced significant evolution over time. This advancement enables the generation of diverse game elements with minimal designer input. Historically, the development of PCG in games has utilized various methodologies, including search-based, solver-based, rule-based, and grammar-based approaches. These techniques have played a crucial role in producing a broad spectrum of content, such as levels, maps, character models, and textures. More recently, the incorporation of artificial intelligence, notably machine learning, has revolutionized the field of content generation within the gaming industry. Leveraging vast datasets and enhanced computational capabilities, machine learning algorithms offer unprecedented opportunities for creating dynamic and immersive game environments. Despite these advancements, a significant lack of comprehensive research thoroughly examines this field from crucial perspectives. This paper analyzes machine learning applications in game content generation, focusing on key issues in the PCG domain, such as enthusiasm, current challenges, and exploring potential future trends.**

***Index Terms*- Artificial intelligence, Machine learning, Game design, Procedural content generation.**

I. Introduction

PROCEDURAL Content Generation (PCG) refers to the algorithmic creation of game content. It is a significant research area and currently stands as one of the most active topics within both the software gaming industry and academic game studies. Game development generally requires assembling a team possessing diverse skills, entails substantial time commitments, and demands significant budget allocations, which culminate in a limited quantity of game content. PCG offers a solution to these challenges by enabling the automated generation of content during both the development and execution phases [1].

Research indicates that PCG is extensively employed across various game genres, though its utilization is particularly notable in action games and platformers. These genres significantly leverage PCG's capabilities, whereas other styles, such as simulation games, have yet to exploit PCG extensively. Moreover, there is no universally superior method for content generation; instead, a diverse array of approaches is adopted, depending on the specific type of content. Further research is essential to examine the methodologies used in conducting experiments and evaluating PCG's effectiveness, as well as to explore its potential applications within the broader context of game development [2].

The field of PCG extends beyond computer science, engaging a growing interest across various interdisciplinary fields. The inherent cross-disciplinary nature of PCG has established its relevance in diverse domains such as biology, architecture, urban studies, and psychology. However, coordinating these varied disciplines, along with the search for appropriate data structures, demands significant effort. Additionally, the areas of computer science that influence these domains are limited to the following elements: grammars, L-systems, shape grammars, and programming languages [3].

One of the primary expenses in video game development is content creation. Procedural generation can reduce the need for resources typically produced manually, requiring minimal or even no human input [4]. The gaming industry has evolved rapidly, becoming a multi-billion-dollar sector annually. Data from the online Steam store show that there are over 10 million active players during peak periods each day. Given the increasing demand from players for more engaging and high-quality content, research in game development, design, and the theoretical foundations of gaming plays a crucial role in addressing some of the current challenges facing the industry [5].

PCG is utilized in games for various reasons, employing diverse techniques to fulfill specific design objectives. Initially, one of the primary goals of PCG was to enhance data compression. For example, the generation of procedural galaxies in the Elite game allowed players to explore an expansive universe. Moreover, Rogue-like games use PCG to create playable experiences that maintain a relatively high degree of control over the generated content. Recent research in PCG has shifted towards developing games that align with player preferences and characteristics. Furthermore, there has been a concerted effort to enhance the replayability and adaptability of games, which has emerged as a significant motivator for ongoing research in this field [6].

The use of Artificial Intelligence (AI) in PCG enhances replay value, reduces production costs and effort, and supports autonomous generation and compression. Additionally, PCG is

1. Faculty of Electrical and Computer Engineering, Semnan University, Semnan, Iran.

Corresponding author Email: dorrigiv@semnan.ac.ir

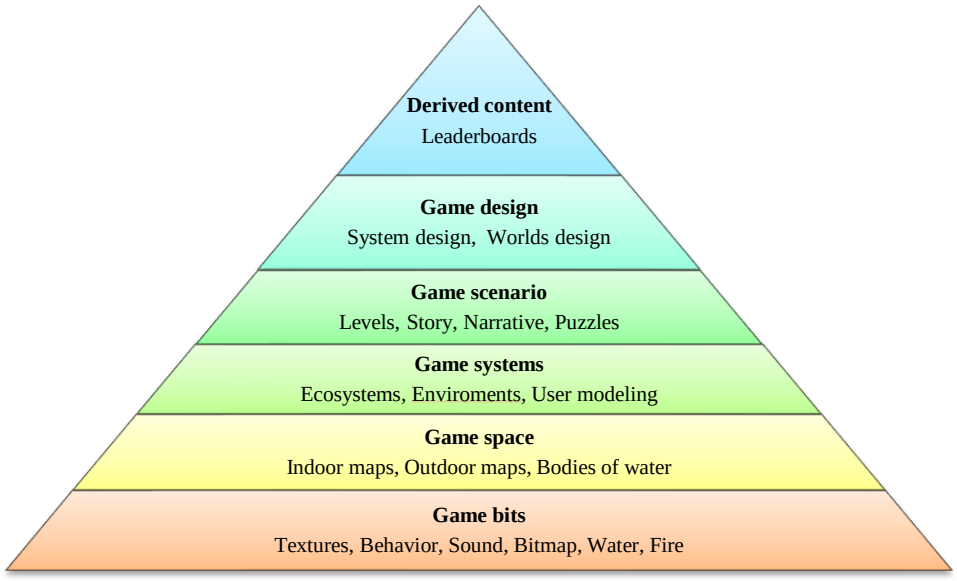

Fig. 1. Taxonomy of six levels of game content [13] and more common content generated procedurally in games using machine learning methods.

well-suited for co-creativity, mixed-initiative design, repair, critique, and content analysis [7].

Numerous intriguing and rewarding tasks can be customized to function within the unique constraints and possibilities game content offers. The structure of this content may vary significantly, and various methods, including Machine Learning, can be utilized for their generation [7].

Many machine learning methods can be utilized in a generative role. These include Autoencoders (AEs) [8], Adversarial Learning [9], Reinforcement Learning [10], Transfer Learning [11], Deep Learning [9], and others [7]. Furthermore, Evolutionary Algorithms (EAs) [12] generate content by searching within a content space, driven by specific objectives [7].

A substantial body of research has been conducted in the field of PCG. However, the existing literature primarily describes the generated content or the methods employed. Furthermore, review articles often emphasize categorizing content types [3], [13] and the methods utilized in their generation, or they focus on broad categories such as PCG via Machine Learning (PCGML) [14], [15]. These reviews tend to pay less attention to the categorization based on machine learning methodologies and their specific applications in generating procedural content [7], [9].

Therefore, this paper proposes a new taxonomy to categorize methodologies in machine learning, specifically for PCG. This taxonomy draws on current trends in research that predominantly employ specialized machine learning techniques for PCG. By categorizing these techniques based on the methodologies used during training, the paper provides a detailed insight into the evolving landscape of PCG through machine learning.

The paper [15] explores the limitations of machine learning algorithms in generalizing to new tasks and environments, proposing PCG as a promising solution. PCG can automatically generate new and diverse content, thereby exposing algorithms to a wider range of scenarios and facilitating the acquisition of more generalizable knowledge. While the paper provides a comprehensive discussion on generality in machine learning, it does not address other challenges associated with machine learning through procedural content generation. Furthermore, previous review studies [3], [7], [9], [13], [14], [15] have highlighted a limited number of significant challenges in this area, providing a general overview; however, they do not extensively discuss the important challenges of PCG in machine learning.

This paper thoroughly investigates the principal challenges faced in PCGML and pinpoints promising directions for future research. The unique challenges of PCGML stem from the fundamental differences between game development and other domains in machine learning, such as vision and audio. Specifically, datasets in game development are typically smaller, represent dynamic systems reliant on user interaction, and often lack accessible high-quality, clean data. Despite these obstacles, they can be effectively addressed using appropriate machine-learning techniques.

Moreover, existing review articles [7], [9], [14], [15] do not comprehensively explore future trends in PCG through machine learning. Consequently, this paper delineates forthcoming trends in the application of PCGML methodologies. Specifically, it addresses not only aesthetics and balance preservation in games through PCGML but also emphasizes the importance of generating unbiased content. Furthermore, this study examines works that utilize PCGML to remove bias or provide algorithmic guidance in generating targeted content.

In this study, we recognize PCG's critical importance in gaming. Specifically, we explore the various machine-learning methods that enhance PCG's effectiveness in games. We will discuss the motivations behind utilizing PCG, address the challenges encountered, and provide detailed examples of algorithmic applications.

The rest of this paper is organized as follows: Section II introduces types of game content. The types of ML methods used in game content generation are presented in Section III. Section IV discusses the motivations for using PCG in games. In addition, Section V examines the challenges of PCG in

games. Section VI discusses some future trends. Finally, Section VII draws a conclusion and Section VIII concludes the paper.

II. Content Types

Game content encompasses a variety of objects including levels, maps, items, weapons, quests, sprites, characters, and rules [7]. Additionally, these objects may be either necessary or optional, depending on the game's conditions. For example, in the Borderlands game, players can choose to discard or select specific weapons [9]. Hendrikx et al. [16] proposed a taxonomy consisting of six distinct levels of game content: bits, space, systems, scenarios, design, and derivatives. The most fundamental units, termed game bits, comprise elements such as textures, sound, music, graphic effects, vegetation, and architecture. While certain elements, like character textures, are indispensable, others, such as sound effects, may be optional. Game space refers to the environments, such as maps and terrain, where the game unfolds. The game design includes the set of rules and mechanisms that govern gameplay, whereas game scenarios organize levels into coherent plans or sequences of events, which can include puzzles, stories, and levels. Often, levels are represented as maps or mazes, and more complex environments are simulated by game systems, which can include ecosystems, road networks, urban environments, and interactions among creatures. Additionally, derivative content such as leaderboards and news items, serves to enhance the game world.

Fig. 1 depicts a taxonomy of six levels of game content as referenced in [16]. It also illustrates the types of content that are commonly generated procedurally in games through the use of AI methods.

III. Machine Learning Methods

Machine learning, a branch of AI, is primarily classified into three general categories based on the training method: supervised learning, unsupervised learning, and reinforcement learning [17].

Methods of supervised learning utilize labeled data to train models for specific tasks such as classification. In contrast, unsupervised learning methods aim to identify patterns within unlabeled data when no labeled data is accessible. Supervised learning models are predominantly employed in predictive tasks, which include predicting outcomes of events, such as the win-loss status in games [14].

Furthermore, two prevalent unsupervised learning techniques are employed in procedural content generation PCG. The first technique utilizes approaches such as AEs to learn a generalized representation of content, which can subsequently be applied to generate new content. The second technique concentrates on sequential data, aiming to understand the relationships among elements, such as words in a sentence, and leveraging this understanding to predict or generate new sequences [9].

Unlike supervised learning, reinforcement learning enables a machine learning model to learn through trial and error within an environment. The model receives information about the current state, the goal, and the available actions along with their potential outcomes. It then seeks to maximize a reward through experimentation. This approach frames content generation as a Markov Decision Process (MDP) [17].

In recent years, the application of machine learning in model training using datasets has experienced substantial growth. A significant driver of this advancement is the resurgence of neural networks, which now form the basis of the field of deep learning. Deep learning has significantly enhanced the capabilities and applicability of machine learning models, especially in the area of content generation. Among these advancements, Generative Adversarial Networks (GANs) stand out, having demonstrated success in generating realistic images, music, and speech. However, various other machine learning methods, including n-grams, Markov models, and AEs, can also be employed for generative purposes. The core principle involves training a model on a dataset that represents a specific distribution, thereby empowering the model to generate completely novel samples [9].

This paper explores the emerging field of Procedural PCGML. We propose a definition of PCGML as the process of creating game content through models that have been trained on samples of existing game content. This method distinguishes itself from search-based and solver-based PCG approaches. Although the latter may integrate machine learning models, such as trained neural networks, to evaluate content quality, the generation of content primarily occurs through a search process within the content space. In contrast, in PCGML, the model itself directly generates the content. Specifically, the model's output, whether derived from random inputs or based on partial or previous game content, is regarded as the final content. This marks a significant differentiation from search-based PCG, where the primary role of the model is to assess content [9].

On the other hand, machine learning methods involve training a model and then generating instances. On the other hand, machine learning methods involve training a model followed by instance generation. GANs, AEs, and CNNs are among the most recognized methods for leveraging machine learning algorithms to generate new content [18]. To date, PCG has achieved considerable success using machine learning [12]. For instance, notable applications include extracting physiological data from user input [19], adapting a game to player preferences by generating next-level blocks for each game level [20], and developing a system to generate game content from images designed by players [21]. Additionally, there are methods for procedural generation of Mega Man game levels by creating spaces, including rooms and connections between different areas [22], [23]. Other innovative uses involve creating WarCraft II and Super Metroid maps using distinct tiles at the pixel level, which combine symbolic constraint-solving methods with discrete neural representation learning [24]. Furthermore, Natural Language Processing (NLP) has been used to generate branching storylines in games [25], and autoregressive models based on transformers have been applied to generate levels for the Sokoban puzzle game [26].

This article introduces a novel categorization of machine learning methods, with an emphasis on procedural content generation, which predominantly employs specific machine learning techniques. The proposed categorization is based on the machine learning method used for training. Initially, the article elucidates the most prevalent machine learning methods, such as AE, adversarial learning, reinforcement learning, transfer learning, and a broader category that includes deep learning. It then provides an overview of various applications of these methods in content generation for games. Fig. 2 illustrates the taxonomy of the methods discussed.

A. Autoencoders

AEs are unsupervised learning techniques based on neural networks, focusing on learning representations of existing content and then generating new content from these representations. They have recently been utilized for image generation and modeling [27].

Specifically, AEs have been applied to the generation of game levels. One of the pioneering projects in this area, proposed by Sarkar et al. [8], involved training a variational autoencoder (VAE) to create game levels. This project incorporated the MAP-Elites quality-diversity algorithm during the training of the VAE's latent vectors. Ultimately, the MAP-Elites algorithm evaluates these vectors to explore a diverse array of levels that optimize specific objectives, such as playability.

Snodgrass et al. [27] extended the use of sample-based binary space partitioning for pattern recombination and employed VAEs in various contexts to generate previously unseen models. This approach preserves structural components and integrates different domains, thereby enabling the generation of levels with characteristics distinct from those observed in the samples.

Furthermore, Sarkar et al. [28] employed a combination of VAEs and a classifier to generate game levels of desired lengths. This approach determines the location of each generated segment and combines them to create coherent levels with variable lengths. These levels result from the implicit fusion of distinct game segments that do not share a similar orientation. This method has been successfully applied to various games, including Super Mario Bros (SMB), Kid Icarus, and Mega Man. Similarly, Thakkar et al. [29] generated game levels for the platform game Lode Runner by utilizing AEs. They produced steps based on desired features by evolving hidden layers within the AEs and represented the content using a multi-channel approach with full fidelity.

Moreover, Mori et al. [30] introduced a method that utilizes AEs to extract features based on the internal state of a one-on-one fighting game, FightingICE. This approach not only facilitates feature extraction from gameplay but also enables the recommendation of gameplay strategies to spectators.

Based on the referenced studies, it is evident that both AEs and VAEs aim to learn representations from existing content and utilize these representations to generate new content. These

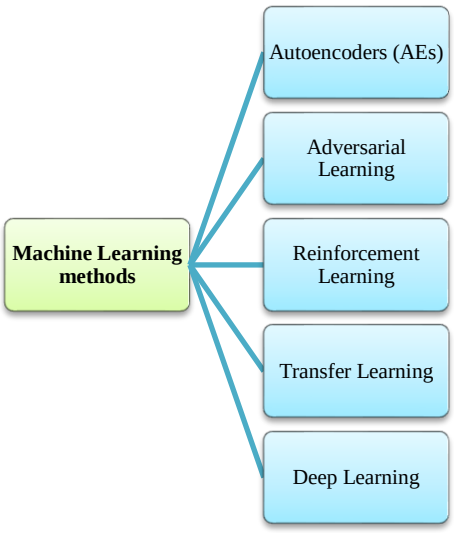

Fig. 2. Taxonomy of common machine learning methods for generation game content.

methods have been effectively applied in the domain of game-level generation, as demonstrated by various research efforts. Although AEs and VAEs share similarities, they differ significantly in their approach to learning: AEs operate under an unsupervised framework, whereas VAEs employ a semi-supervised methodology. Additionally, alternative generative techniques, such as GANs, present distinct advantages and may be preferred based on specific criteria, such as the need for realism or capabilities in imitation learning [9].

Overall, the use of autoencoders AEs in game content generation offers numerous advantages, including the ability to generate diverse levels and optimize for specific objectives such as playability. However, challenges persist, particularly in designing appropriate objectives and evaluating the quality of the generated levels.

B. Adversarial Learning

Adversarial learning is a machine learning method designed to mislead models by using malicious input specifically crafted to deceive classifiers. Adversarial learning methods are particularly well-suited for image processing, utilizing pixel-based or tile-based representations. Among the adversarial learning approaches reviewed, GANs and their variants are the most favored [9].

GANs represent a category of machine learning models composed of two distinct networks: a generator and a discriminator. These networks are trained iteratively, to generate realistic samples while effectively distinguishing between authentic and synthetic data [31].

Various adversarial learning methods have been utilized to generate diverse content. Some notable applications of GAN-based models include the creation of fantasy and science-fiction icons [32], character development in video games [33], and the production of visually symmetric patterns at a large scale in match-3 games [34]. These methods also facilitate the generation of playable Three-Dimensional (3D) terrains [35], the design of SMB game levels using a corpus of video game levels [36], and the creation of levels for the Freeway, Zelda,

and Colorescape games within the General Video Game AI (GVG-AI) framework [37].

Therefore, GANs are utilized for generating images in games. For example, Karp et al. [32] introduced a GAN-based model for generating fantasy and science fiction symbols with a high degree of realism. They employed a modified version of adaptive discriminator augmentation to train the model.

Furthermore, adversarial learning techniques are now being utilized in the generation of characters for video games. Chelliah et al. [33] introduced a novel approach for procedurally generating NPCs using Style GAN, a specialized neural network architecture. This model is capable of producing NPCs with distinct appearances and behaviors, thereby enriching the diversity and complexity of video game environments.

Investigating the ability of various techniques to accurately capture extensive symmetrical visual patterns remains an open question. Volz et al. [34] employed match-3 games as a method to assess the capability of prevalent PCGML algorithms, including GANs, in generating relevant patterns. Furthermore, Ivanov et al. [35] developed a GAN-based model to produce 3D procedural terrains in video games, utilizing satellite imagery for training purposes. Their study also evaluated the terrain's playability and the feasibility of routing between different points using a routing algorithm.

Additionally, given the widespread use of GANs in image generation, numerous studies have employed GANs to create game levels. For instance, Volz et al. [36] trained a GAN using a corpus of video game levels to generate SMB levels. The GAN applied the covariance matrix adaptation evolution strategy to produce levels that met specific goals, while simultaneously assessing the playability of these levels.

In addition, GANs have demonstrated a strong ability to achieve the desired shapes of maps while also meeting appropriate conditions for part distribution. Irfan et al. [37] employed deep convolutional GANs to generate levels, training them on levels from three games Freeway, Zelda, and Colorescape obtained through a random generator within the GVG-AI dataset. Both VAEs and GANs utilize latent variable models to generate parts, which are then integrated into the final stage to form a complete level. This method successfully produced levels that closely resembled the training levels.

Furthermore, another method utilizes a model with vector inputs that functions as a geometry-based mechanism within a layered structure known as Compositional Pattern Producing Networks (CPPNs). CPPNs have been employed to generate levels in games such as SMB and The Legend of Zelda, exploring areas that other methods may not address [38].

Overall, the reviewed literature indicates that adversarial learning techniques, especially GANs, provide several benefits for generating playable images, characters, and game levels in video games. GANs can also be employed to manipulate the detailed output of game levels. Nevertheless, these techniques present challenges related to effectively training the models and mitigating bias in the training data.

Alternative approaches to adversarial learning and GANs for content generation include VAEs, Recurrent Neural Networks (RNNs), Sequence-to-Sequence models, Transformer models, and Evolutionary Algorithms. VAEs are adept at compressing and reconstructing data, while RNNs and Sequence-to-Sequence models excel in generating sequential data, such as text. Transformer models are particularly effective in both text and image generation. Additionally, Evolutionary Algorithms, such as genetic algorithms, demonstrate efficacy in generating structured data and images within complex search spaces [39]. Each method offers distinct strengths and weaknesses, and the selection of a specific approach should be guided by the particular requirements and characteristics of the intended application.

C. Reinforcement Learning

Generative models, including PixelCNN, VAEs, and GANs, face several challenges when deployed in gaming environments due to limited training data. Reinforcement learning is another method used in this context. Particularly, when training data are insufficient for developing new games, methods based on reinforcement learning facilitate self-learning and knowledge expansion to devise solutions. In this process, an agent learns through trial and error, iteratively selecting actions that optimize the expected quality of the output. This method has demonstrated substantial success in controlled environments such as games. Nonetheless, the translation of PCG tasks into MDPs continues to be a vibrant area of research, without a universally applicable solution established to date [40].

Several studies have utilized reinforcement learning-based methods to generate content. For instance, Khalifa et al. [41] applied reinforcement learning to level generation, treating it as a sequential task. They enhanced the quality of the generated levels by using reinforcement learning to optimize future actions. Moreover, they investigated three different strategies for transforming two-dimensional level generation challenges into Markov decision processes and implemented these methods across three distinct games. Nam et al. [40] created varied levels consisting of discrete sections in turn-based RPG stages, employing a stochastic noise policy on both a deep Q-network and a deep deterministic policy gradient. Additionally, Shu et al. [42] introduced a framework for experience-driven PCG via reinforcement learning. This framework is capable of producing infinite, playable levels for SMB that vary in enjoyment while catering to specific player experiences, modeled as reward functions. Notably, this framework is adaptable to any game designed as a segment-based sequential process.

The Monte Carlo Tree Search (MCTS) is an algorithm used in reinforcement learning, designed to select actions that maximize long-term rewards. Its capability to manage complex decision-making challenges makes it a valuable method for reinforcement learning. However, its high computational complexity and dependence on the accuracy of simulations must also be considered. MCTS has been applied across various domains, including the generation and testing of game content. In their research, Liu et al. [43] explored and analyzed the building blocks of Kingdom Rush: Frontiers, a tower defense game. They assessed the playability of content generated by

MCTS, including sequences of monsters, road maps, and tower placements. MCTS was utilized to sample behaviors within a play space, and these techniques were applied in two simplified, perfect-information, adversarial game domains inspired by Scrabble and Hearthstone. Additionally, Holmgard et al. [44] employed a variant of MCTS to model players and test game content using procedurally generated archetypal characters. They developed alternative node choice criteria using EAs to replace the standard MCTS criteria.

The ability of physics-based video games to construct controllable and practical physical environments has heightened the significance of AI research in this domain. Gamage et al. [45] introduced a content generation system for Angry Birds, a physics-based video game. Their system showcased that the levels generated adhered to the designer-defined objectives and manifested distinct novelty. To achieve this, a reinforcement learning agent capable of adapting to newly generated content was employed in the experiments.

Overall, reinforcement learning-based methods have been widely employed to generate diverse and complex game content across various genres. Although challenges such as reward delay and the high dimensionality of game content persist, these methods provide an effective solution for PCG in games [42]. Additionally, alternative approaches to reinforcement learning for content generation include a variety of AI methodologies. One notable method is EAs, which mimic natural selection processes to evolve solutions for optimization problems. EAs, including Genetic Algorithms and Evolution Strategies, iteratively generate and evaluate candidate solutions, progressively refining them through successive generations. Another approach is imitation learning, where models learn from demonstrations provided by humans or other agents. Common techniques used here include behavioral cloning and inverse reinforcement learning. Moreover, hierarchical reinforcement learning decomposes complex tasks into manageable subtasks, facilitating more efficient learning and decision-making processes [46]. These methods represent diverse alternatives to reinforcement learning, each offering unique advantages and applications across different domains.

D. Transfer Learning

Transfer learning is a machine learning methodology that focuses on transferring knowledge to a new, yet similar, problem. In co-creative PCG, humans and machine learning agents collaborate to generate content. However, a major hurdle is the high cost and complexity associated with preparing large training datasets [47]. To address this, Zhou et al. [47] employed transfer learning, a technique that facilitates the adaptation of knowledge learned from one game to another. They successfully implemented this strategy using pre-trained models to generate dungeons for the game Zelda. Despite these advancements, ensuring the relevance of transferred knowledge and selecting the most appropriate transfer learning techniques remain active areas of research.

Overall, alternative approaches to transfer learning for content generation encompass a variety of machine learning methodologies. One such method, meta-learning, focuses on developing models that can quickly adapt to new tasks using minimal training data. Meta-learning algorithms, such as model-agnostic meta-learning and reptile, optimize the model's parameters across multiple tasks or domains to enhance learning efficiency. Another approach is Few-Shot Learning, which aims to train models to generalize effectively from a limited number of examples [48]. Additionally, GANs and VAEs are widely utilized in content generation tasks. These methods leverage their ability to learn complex data distributions and generate novel samples. Collectively, these methodologies provide diverse alternatives to traditional transfer learning, each with distinct strengths and potential applications across various domains.

E. Deep Learning

Unlike traditional machine learning methods that require meticulously hand-crafted features, deep learning is driven by deep neural networks. These networks comprise multiple hidden layers and advanced functionalities within their neurons, enabling the automatic extraction of meaningful patterns from data. CNNs, RNNs, distributed representations, AEs, Long Short-Term Memory (LSTM) units, and GANs are just a few examples of powerful deep learning architectures driving innovation across various fields [17].

In the field of games, deep learning assumes a pivotal role in content generation and classification, although its techniques may overlap with those of other categories. This paper clarifies the classification and unique contributions of deep learning in comparison to other methodologies, highlighting its ubiquitous presence across various categories. Key attributes of deep learning encompass its ability to process large and complex datasets, identify complex patterns, and enable transfer learning across similar fields. Deep learning is particularly effective in managing extensive and detailed datasets, such as those involving text, images, and audio.

Whereas some methodologies require precise definitions of features or the use of specific algorithms, deep learning can bypass these requirements, often resulting in enhanced performance in tasks like content generation. Deep learning excels at deciphering complex relationships within data, establishing it as a preferred method for a range of modern AI applications, including image recognition, speech translation, content generation, and medical diagnosis. Conversely, simpler learning techniques such as decision trees may be preferable when dealing with less complex data or when the available training data is scarce. Moreover, these simpler models typically offer greater interpretability, which is essential in certain applications [17].

Although deep learning has achieved impressive results, it should not be equated with true artificial intelligence, as significant challenges such as understanding natural language and human intentions remain unresolved. To enhance the capabilities of deep learning, researchers have developed various specialized architectures. These include CNNs for image data and RNNs for sequential data, each designed to process specific types of data efficiently. A fundamental advantage of these architectures is their capacity to automate the process of feature learning. This involves the automatic

identification of patterns within data, which reduces the necessity for manual feature engineering. Feature learning typically occurs in a hierarchical manner, where simpler patterns serve as the foundation for more complex structures. The specific mechanisms of this hierarchical process vary depending on the architecture employed and the nature of the data being analyzed. Deep learning is especially effective in managing large-scale, noisy, and unstructured data, prevalent in real-world scenarios [17].

Modern video games necessitate a substantial amount of high-quality media, which can be produced using deep learning techniques for creating images, designing rules, composing music, and generating various other types of content [9].

In the existing literature, Murphy et al. [47] introduced a deep learning-based approach for synthesizing face textures under defined constraints. Meanwhile, Goadrich et al. [49] employed deep learning to create pixel-based art images that were of sufficient quality for inclusion in the video game Trajes Fatais: Suits of Fate. Additionally, they generated sprites using deep learning-based asset generation techniques, which resembled the work of the artists' team.

Sirota et al. [50] proposed the use of deep neural networks to facilitate communication in inter-NPC languages and further enhanced the method by employing neuroevolution. This innovation led to the successful creation of new languages for communication. In another study, Karavolos et al. [51] linked the structural elements and rules of a shooter game with the outcomes of gameplay. Their model, which was trained on simulated two-player deathmatches, can predict the duration of a match and identify the winner based on a top-down map of the level and the class attributes of the players.

Sorochan et al. [455] utilized an LSTM network, trained on player path data extracted from gameplay videos, to generate more consistent levels in the game Lode Runner. Similarly, Liang et al. [53] applied a hybrid model combining CNN and LSTM to produce more realistic beatmaps directly from music files in the renowned rhythm-based game, OSU!.

Tanabe et al. [54] proposed a sequential encoding method that processes text data on a single level, which significantly improves the stability and diversity of levels generated for the Angry Birds game domain. This method shows notable advantages over current techniques that employ tile-based encoding and treat the data as an image.

Transformers are a type of neural network architecture that has gained popularity in recent years for NLP tasks. They are especially effective for tasks that require understanding long-range dependencies in text, such as machine translation and text summarization. Additionally, a novel approach to PCG using offline reinforcement learning and transformer networks has been introduced. This method employs an autoregressive model based on transformers to iteratively generate game levels, overcoming the limitations of traditional PCG methods, which often produce repetitive, predictable, or inconsistent content. Experimental results on the Sokoban puzzle game have shown that this approach can generate more complex and diverse game content compared to existing methods, achieving these results in significantly fewer steps. Overall, this paradigm presents a promising avenue for enhancing game design and online content generation [26].

According to the literature reviewed, it is evident that one of the challenges associated with deep learning is the requirement for substantial volumes of high-quality training data to achieve accurate outcomes. Additionally, deep learning models are computationally intensive and necessitate advanced hardware for effective operation. There is also a critical demand for skilled data scientists and engineers to design and implement specialized deep-learning models tailored for specific applications [17]. Despite these challenges, deep learning has demonstrated considerable potential in fostering the development of novel and innovative techniques for content generation and gameplay enhancement in video games [9]. Although deep learning presents several hurdles, the prospective advantages render it a compelling field of study within the gaming industry.

IV. The Motivations for Utilizing PCG

In the previous section, we provided a brief overview of various machine learning methods and their applications in generating content for games. The examples discussed underscore the importance of using machine learning in PCG for games. Beyond creating diverse content, another advantage of PCG is its capability to introduce specific characteristics within the games. Subsequently, this article discusses the motivations for employing PCG to create distinctive features in games, illustrated with examples of machine learning methods used for implementation. Fig. 3 presents a taxonomy of the motivations discussed for utilizing PCG.

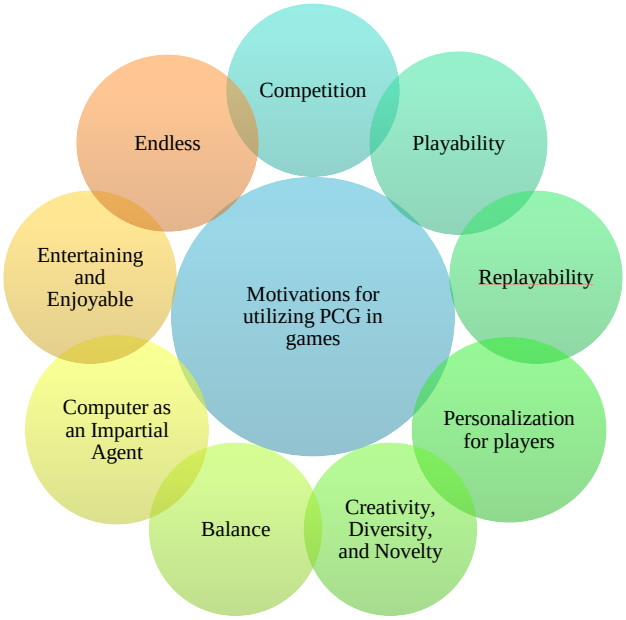

Fig. 3. Taxonomy of the motivations for utilizing PCG in games

by a DDA component, significantly improves the playability of the levels it produces [5].

A. Competition

Creating engaging game levels constitutes an art form,

involving the meticulous crafting of game spaces to provide players with a balanced mix of challenge, competition, and interaction. PCG presents a promising approach to enhancing this aspect of game design, especially in fostering competition. A recent research initiative developed a general PCG framework for a Two-Dimensional (2D) tile-based game using Unity. This framework was augmented by incorporating a Dynamic Difficulty Adjustment (DDA) component. The DDA evaluates specific criteria after each level and dynamically adjusts the difficulty to maximize player satisfaction and maintain exciting competition. This strategy employs PCG to introduce fresh challenges and optimize the difficulty curve, ultimately enhancing the gameplay experience [5].

B. Playability

Playability in games refers to the quality that ensures game levels are functional and enjoyable, allowing players to engage with them without encountering design flaws or gameplay barriers. Games exist in a domain where creativity is not limited to human authorship alone. The gaming industry, along with academic researchers and enthusiasts, has widely adopted PCG as a tool to enrich games with additional playable content and to facilitate novel gameplay experiences [55]. Research has focused on the creation of game levels that are not only playable but also designed to integrate specific game mechanics effectively. A notable instance is the development of a PCG generator for a 2D tile-based game. This generator, enhanced

C. Replayability

PCG decreases resource consumption for developers and simultaneously enhances the replayability of digital games [56]. By facilitating the automatic generation of nearly infinite content, PCG enables players to repeatedly engage with the game, allowing them to refine their strategies over time. The objective of PCG in gaming is to create replayable levels that are in alignment with the preferences of players and designers [55], [57].

One game genre that benefits from replayability is the Roguelike genre. By incorporating PCG, the automatic and random generation of game content can significantly enhance replayability. A common algorithm used in PCG is the Drunkard's Walk (DW), which generates space and content based on the path it traverses. This algorithm's use of ambiguous movements can create random spaces and content. In this context, a Roguelike game employing the traditional top-down shooting approach and the DW algorithm was developed to improve player satisfaction with the gameplay [58].

D. Personalization for Players

PCG offers exciting opportunities for creating personalized gaming experiences. A crucial aspect of personalization is ensuring that the game provides an appropriate challenge level. The implemented DDA component dynamically modifies the difficulty level after each stage based on specific criteria, thus catering to individual player skills and enhancing engagement [5]. Research has explored the integration of DDA within PCG frameworks. One recent study utilizes a general PCG framework for developing a 2D tile-based game in Unity. This framework includes a DDA component that analyzes specific criteria after each level and dynamically adjusts the difficulty accordingly. Such personalization maintains the game's excitement and accommodates individual player preferences [5]. Another study adopts a data-driven PCG approach using genetic algorithms and support vector machines. This method customizes educational game content according to individual abilities, demonstrating the potential for personalized learning experiences [58].

E. Creativity, Diversity, and Novelty

Automated game design extends beyond mere content creation; it also necessitates the capability to evaluate and guarantee the novelty, quality, and cultural relevance of the generated content. This process involves assessing elements such as surprise and the degree to which the content aligns with the specific game's design. Consequently, the integration of computational models of creativity is essential for developing tools that facilitate or automate these design processes [60].

In a recent study, an educational programming puzzle game named BOTS incorporated a pattern-based puzzle generator. Two puzzle experts independently designed their puzzles. The evaluation of puzzles generated by the automated generator, as compared to those crafted by experts, demonstrated both structural and visual innovations [61]. A primary challenge in PCG for game-level design is creating a diverse array of levels that are comparable to those devised by humans. In another study, game scene designs were initially generated randomly and subsequently refined using an illustrative genetic algorithm. This approach resulted in a diverse set of game scenes that successfully integrated the credibility of game rules with aesthetic principles [62].

F. Balance

Researchers are increasingly exploring the potential of PCG to create balanced and engaging experiences in games. In a particular study, the researchers investigate the use of PCG for crafting balanced experiences in Location-Based Games (LBGs). This approach capitalizes on players' locations and Points of Interest (PoIs) in their vicinity. It employs a directional weighted graph as a simple yet effective method to represent the relationships between these PoIs. The popularity of this technique has surged with the advent of wearable devices and the integration of the Internet of Things into LBGs. A genetic algorithm is subsequently utilized within the PCG framework to generate LBG instances. These instances are not only relevant to the player's location but also maintain a balance in terms of difficulty or challenge. Studies have shown that players report greater satisfaction when engaging in balanced games compared to unbalanced ones, even when the games are otherwise similar [63].

G. Computer as an Impartial Agent

PCG is often touted as being free from bias and error, a claim attributed to its computer-generated nature. This assertion stems from the belief that computers can offer a more impartial interpretation of game mechanics due to their lack of imagination and hidden agendas, which purportedly frees them

from the inherent biases and motivations that might influence human designers [64]. However, to fully understand PCG through Machine Learning, it is essential to scrutinize the generation process itself, along with the potential limitations and biases embedded in the algorithms [9].

PCG algorithms can inherit biases from the data on which they are trained. For example, an algorithm trained on levels that feature a specific placement of enemies may, in the future, generate repetitive and predictable content. Recent advances in deep learning facilitate the analysis and assessment of biases in PCG algorithms. This is crucial for evaluating level balance and steering content generation toward the desired content or gameplay experiences [9].

Recent advances in deep learning enable the analysis of latent spaces learned by models. These spaces capture the underlying characteristics of the training data. By leveraging these spaces, researchers can develop more semantically meaningful measures of similarity between pieces of generated content. For example, AEs can be used to categorize game levels based on style, as demonstrated in [65]. Deep learning models can also be trained to predict game outcomes based on generated levels and character classes, assisting in the assessment of level balance and guiding content generation toward desired gameplay experiences [66]. Additionally, frameworks utilizing stacked denoising autoencoders can personalize games for different player types based on their interactions within the generated content [67]. Training a CNN on human player data facilitates the prediction of player actions in unseen levels, enabling the evaluation of the diversity of player interactions within generated content [68]. Moreover, deep reinforcement learning agents can be trained to mimic human-like play behavior, providing more realistic playtesting for games, such as Mario [69].

Another approach to assess bias in PCG algorithms involves simulated playtesting using ANNs and deep learning. This method employs AI agents to interact with the generated content, and analyzing their behavior can provide insights into the features of the content and the overall generative space [70]. While these methods primarily focus on evaluating the generated content, the findings can also be applied to explore the broader generative space, identifying potential biases or limitations within the PCG algorithms.

H. Human Replacement

Early implementations of digital PCG, which facilitated the transition of tabletop role-playing systems to computer platforms, enabled individuals to engage in gaming experiences that traditionally required a social setting and the participation of multiple human players. Contemporary research transcends the mere replacement of human designers; instead, it provides innovative methods for experiencing these games [64].

I. Entertaining and Enjoyable

Does PCG enhance the engagement and entertainment value of game content? This question is explored through two applications of PCG, focusing on level creation and visual content generation. Techniques primarily involve generating maps and terrain for levels. In games where players either avoid or engage in combat with hostile entities, the player experience also includes aspects such as the placement of enemies and resources. Crafting an engaging level involves finding an optimal combination of these elements [63].

A game named Bus Runner has been developed, featuring an endless runner based on shared data. By integrating features of the physical world, such as recognizable landmarks, with the virtual game environment, the player's experience is enriched. These enhancements align with game development goals and broadly influence the generation of entertainment-based content [71]. Furthermore, Wang et al. [72] generated levels for the SMB game using a pre-trained GAN to map a latent code to a level segment, alongside a soft actor-critic-based reinforcement learning designer. This designer repairs and simulates each level post-generation and computes rewards for each segment. The levels generated are not only entertaining but also adaptable to various game styles and fundamental game conditions.

J. Endless

Endless games are designed to offer players an infinitely engaging experience. Unlike traditional games that have a defined ending, endless games utilize PCG to continually present new challenges, often accompanied by a dynamically increasing difficulty curve. This strategy maintains player engagement over extended periods as they strive to enhance their skills and surpass their high scores [73], [74].

In the field of game design, numerous studies have been conducted. One notable study introduces a new framework capable of generating endless, playable SMB levels that vary in entertainment value and differ from previous segments. This framework employs a pre-trained GAN generator for level creation. Additionally, reinforcement learning agents are used to select appropriate segments for incorporation into current levels. The use of reinforcement learning enables real-time level generation within SMB while maintaining specific standards of entertainment and variation. Generated segments are automatically refined using an Evolutionary Algorithm and assessed for playability by an A* agent [42].

K. Aesthetics

The design choices in PCG systems, when coordinated with game mechanics, create unique dynamics. These dynamics, in turn, enhance a variety of aesthetic experiences in gameplay. Three key aesthetics are highlighted: (1) Exploration: PCG facilitates exploration by providing players with new environments or procedural systems, enhancing player learning over time. This aesthetic is supported through dynamic searches in the expansive game world and generative strategies. (2) Challenge: PCG underpins challenges through memory dynamics and response strategies, fostering practice in diverse environments. Games that use PCG to induce quick player reactions present a challenge where decisions must be made moment-to-moment. Conversely, games that employ PCG to elicit player reactions, not necessarily under time constraints, introduce challenges involving unfamiliar content. Practicing in

various settings allows players to encounter new challenges by experiencing similar gameplay in multiple contexts. (3) Social Interaction: PCG promotes player interaction beyond the game environment by creating communication systems, engaging with the dynamic player community, and shaping how players discuss the game, including strategies for different content configurations [6].

V. PCG CHALLENGES IN GAMES

Previous research has identified several key challenges in the development of commercial video games [3], [7], [9], [13], [14], [15]. The most crucial of these challenges is further elaborated upon subsequently. The examples presented in earlier sections illustrate that the application of AI approaches can mitigate these issues to some extent. Fig. 4 depicts a taxonomy of the PCG challenges discussed in video games.

A. Content Generation at the Top of the Hierarchy

In the layers of systems and scenarios, PCG has not been extensively adopted in commercial applications. Its use has been primarily confined to early, simplistic games such as Rogue-likes, with a few notable exceptions like No Man's Sky. However, the application of PCG to create more complex and nuanced content, including systems (e.g., quests) and scenarios (e.g., narratives, storylines), remains limited in commercial game development [14].

B. A More Precise Generation

In many areas of game design, such as terrain and tree generation, PCG is particularly effective. However, in other areas, manually created content continues to dominate. The primary distinction between these methods lies in the level of detail they can achieve. For example, SpeedTree is capable of generating forests with a higher level of detail than those produced by even the most dedicated teams of designers. Conversely, the most advanced city-generation algorithms still yield building profiles whose textures tend to appear overly generic [14].

C. Guaranteeing Playability

One of the primary barriers to the widespread adoption of PCG in game development is the limited reliability of many search-based and machine-learning approaches. Although this issue may not affect offline or pre-computed methods, it represents a significant challenge for real-time generation. Ensuring playability requires the use of indirect representations that are carefully designed and implemented [63].

D. Personalized Content

Personalized content generation, a task that surpasses human capabilities yet is achievable through an online algorithm, involves the development of an internal player model to customize content for each user individually. This method enables games to not only generate customized content but also to evaluate and adjust it implicitly during gameplay. Successful implementations of this approach can be seen in games such as Galactic Arms Race, where it is used to generate unique weapons, and Left 4 Dead, where it tailors the dynamics of attacking waves. However, this technique has not been fully explored for generating complete game levels [14].

E. Data Scarcity

Supervised machine learning methods perform optimally

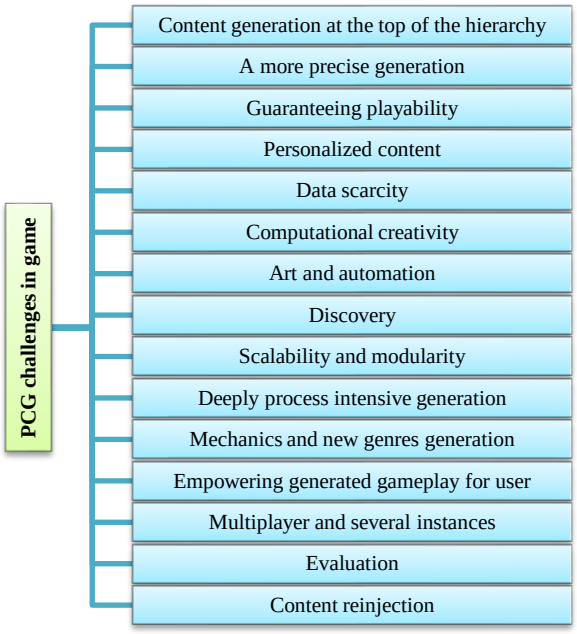

Fig.4. Taxonomy of the PCG challenges in games

when a substantial amount of data is available for training. In cases where the goal is to generate additional levels for a specific game, the training can only be conducted using the existing levels. Within the realm of image processing, various data augmentation techniques have been developed to mitigate this limitation. Additionally, some initial studies in domain transfer, which involves training on data from a similar domain, have yielded promising outcomes. Reinforcement learning is also utilized in scenarios where no data is available, enabling the initiation of systems that facilitate interactions essential for the learning processes of agents [14].

F. Computational Creativity

The demand for extensive content in video games often surpasses the capabilities of game designers. For example, developing and testing a single feature can take a week, leading game development companies to invest millions of dollars monthly in creating new content. This discrepancy between the demand and the supply of new content results in frustration for both players and developers. A promising solution is the generation of on-demand content, which allows for the customization of game elements tailored to individual player preferences. Furthermore, this method facilitates the integration of the current environmental state during production, an achievement not possible with traditional techniques. By employing developers to design rules and assets for a content generator, rather than directly creating content, not only can development costs be significantly reduced, but the time needed to produce new game content can also be cut from days to mere seconds. This approach is an application of computational creativity, employing AI to generate visually appealing game

elements such as landscapes, characters, and quests [75].

G. Art and Automation

In a research context, business discourses concerning racialization in creative industries, particularly related to caregiving practices essential for maintaining production systems, were systematically examined. Critical studies on self-automation suggest that the primary concern associated with computational automation should not focus solely on job loss. Instead, it highlights how the creative endeavors of designers are enhanced and managed with the aid of algorithms, potentially preventing the displacement of jobs within these fields [76].

H. Discovery

Discovery in procedural content generation (PCG) enhances the player's experience by offering new environments to explore and introducing innovative procedural systems for gradual comprehension. This aesthetic is supported by the dynamics of navigating expansive worlds and formulating strategies for content generation. In the former scenario, discovery occurs through traversing vast, unfamiliar landscapes. In contrast, in the latter scenario, discovery involves exploring the game's generative mechanisms. Machine learning techniques in PCG play a crucial role throughout the entire game design phase by facilitating exploration and accelerating content creation. However, a significant challenge arises in exploring a vast amount of generated content, particularly in the absence of supportive tools. The development of such tools is essential to encourage exploration while respecting the inherent nature of the design process. DesignSense, a notable tool for visual analysis, was initially developed to address similar challenges in architectural design. In a specific case study, DesignSense was utilized to analyze a dataset containing generated levels for a puzzle game [77].

I. Scalability and Modularity

Every day, hundreds of millions of individuals engage in computer games, where diverse game content—including 3D objects and abstract puzzles- serves as the focal point for entertainment. This demand necessitates the generation of a substantial volume of content. As games scale up, players' expectations for fidelity and immersion increase correspondingly. Consequently, research aimed at aiding game designers and developers to rapidly and effectively produce large volumes of content is essential [78], [79].

A comprehensive classification of game content involves six distinct layers: bits, space, systems, scenarios, designs, and derivatives. Research suggests that techniques developed for generating content in one layer may apply to others [13]. Modular programming offers significant advantages: it allows for the development of individual modules with limited knowledge of other modules, and it facilitates the reassembly and replacement of modules without the need to reconstruct the entire system [81]. The Roguelike game genre, known for its ample complexity, serves as an effective platform for investigating scalability issues. The findings demonstrate that sets of programming solutions are viable for PCG in video games. Particularly, the hierarchical implementation of these solutions effectively addresses the complexities associated with such gaming environments [78].

J. Deeply Process Intensive Generation

A process-intensive generator possesses the capability to autonomously produce content with minimal input from its creator. This type of generator transfers design knowledge from its foundational elements to the algorithm it employs, thereby actively participating in the design process and rationalizing its decisions. It evaluates the quality of its output by referencing past experiences and incorporating inspiration from external sources. Developing such a generator represents a substantial challenge in artificial intelligence, but even modest advancements promise to transform the fields of gaming and design tools. Envision a design tool equipped with a content generator that not only creates but also elucidates its creations to the user. Consider a journaling game where players provide event descriptions and photographs, and the system constructs a video game based on those inputs [14].

A study introduces a method for generating expansive dungeon environments in games. This technique employs a two-step evolutionary process: initially, it generates a high-level map overview, followed by defining specific constraints and objectives. These parameters then guide various optimization algorithms to develop detailed map segments. Compared to direct encoding methods, this modular approach enhances scalability and computational efficiency, affording users increased control over different levels of detail [14].

K. Mechanics and New Genres Generation

Some of the most engaging moments in gaming occur when a game introduces exceptional content that diverges from previously encountered themes. In these instances, players are challenged to comprehend the game's mechanics and develop new strategies. For example, consider the introduction of unique level elements, such as platforms that control player movement, in New SMB. While research has advanced PCG, these advancements often depend on predefined game elements. How can systems be engineered to autonomously generate innovations? This requires a form of player modeling that explores intricate details beyond mere numerical assessments of enjoyment and frustration. Such modeling should include a depiction of the game mechanics and their interactions with various game components. Another research avenue in interactive content generation involves the creation of distinct game genres: What factors contribute to the development of a diverse collection of games that, while sharing a common theme, are markedly different from each other? [14].

As an example, one study explores the complexity of video game development, which encompasses various creative domains. While creativity is commonly associated with fields such as art and music production, coding is typically viewed as more functional than creative. This study aims to investigate how software can be customized to facilitate creative

exploration and generate new game concepts through the automated analysis, modification, and execution of game code. It presents an illustrative system capable of analyzing a game's codebase to devise novel game mechanics. Additionally, the study demonstrates how these mechanics can be implemented in the design of game levels and in conducting playtesting [14].

L. Empowering Generated Gameplay for User

Content plays a pivotal role in many games, and user-generated content adds a substantial dimension. However, creating engaging content can be challenging for players lacking design skills. The game Spore has achieved significant success by enabling users, regardless of their design proficiency, to create high-quality content. It provides straightforward yet sophisticated tools that empower players while respecting their creative intentions. Similarly, Magic Crayons enhances user experience through interactive content. Efforts to develop PCG tools have focused on providing designers with a supportive brainstorming system, facilitating the creation of diverse and easily modifiable content. Nonetheless, the challenge remains in devising tools that help players actualize a vision they possess but are unsure how to implement [14].

What does PCG imply for multiple players actively engaging with the generated content, and how is a generator designed to cater to a multiplayer-oriented experience? The integration of PCG in multiplayer games can lead to the emergence of diverse content. In such settings, players share a common space and are empowered by mechanics that allow them to influence each other's environments. For instance, consider a cooperative multiplayer platformer where the actions of each player contribute to the creation of new content for others. In this scenario, players must communicate effectively to achieve a shared goal. Within the realm of design tools, a pertinent question emerges: How can a tool assimilate knowledge from the experiences of previous users or recognize common patterns of user interaction? [14].

M. Evaluation

The evaluation of procedurally generated content in games is crucial for various aspects, including playability. For instance, constructive algorithms play a pivotal role in rapidly developing diverse solutions for 3D game environments. However, the use of cellular automata algorithms has resulted in undesirable gaps in building generation, highlighting the necessity for an evaluation process in algorithm-driven game world creation. In response, a Space Syntax-based evaluation method was developed to assess the quality of procedurally generated game levels in terms of connectivity, integration, and mean depth. A new game called The Haunted House and three different test sets with varying room dimensions were created. From a game design perspective, implementing partial divisions in game levels may be a more effective strategy than creating a completely integrated level. The results show that utilizing the Space Syntax method allows game developers to systematically evaluate procedurally generated game levels, enabling them to test and adjust various parameters to enhance the gaming experience [81]. Moreover, StoryTec has successfully modeled digital educational games. This tool demonstrates how content can be generated and incorporated both in the authoring tool and during gameplay, showcasing which game concepts can be implemented using this method [82].

N. Content reinjection

Content reinjection refers to the process of reintroducing or modifying existing content within a system to enhance or alter its functionality and aesthetics. This technique employs machine learning algorithms to analyze, comprehend, and integrate existing game content, facilitating the generation of new content that blends seamlessly with the established game world. Content reinjection has been utilized across various disciplines, including video game research, to explore the effects of content alterations on player experience and game mechanics. For example, one study applied content reinjection to the classic game Super Metroid to examine the impact of changes in level design and enemy behavior on player experience and game balance. This research not only provides a method to maintain the relevance and playability of classic games through the addition of new content and features but also enables researchers to derive fresh insights into game design and mechanics. The methodologies employed in this study have potential applications in other game design areas, such as PCG and level design [83].

VI. Discussion and Future Trends in PCG

Numerous aspects must be considered within the landscape of PCG in computer games. For instance, integrating machine learning with PCG in games has proven beneficial for game research. Machine learning enhances our capacity to generate content; however, games simultaneously pose challenging problems for AI research. Machine learning methods not only create new opportunities for generating independent content but also have widespread applications across various games [84]. Below, we discuss some of these aspects in detail. Fig. 5 illustrates a taxonomy of the future trends in PCG for games discussed herein.

A. Mixed Reality

Mixed reality games, also referred to as hybrid reality games, occur simultaneously in both physical and virtual environments. Unlike traditional games that rely on a singular primary play space, mixed reality games operate across various platforms, including physical spaces, digital environments, or game boards. The hallmark of these games is their ability to merge physical and digital elements, thus forging a new reality. This integration is made possible through advancements in computer vision, graphical processing, display technologies, input systems, and cloud computing [84].

The integration of machine learning and PCG algorithms in mixed-reality games represents a relatively unexplored field. These algorithms hold promise for significantly enhancing player experiences in mixed-reality environments. For instance, a recent study introduced two prototype mixed-reality games

that employ PCG to design levels that capitalize on the unique features of the player's physical surroundings. This approach allows for the customization of game levels to suit individual users, thereby adjusting game difficulty and influencing the player's movements within the real-world environment [9].

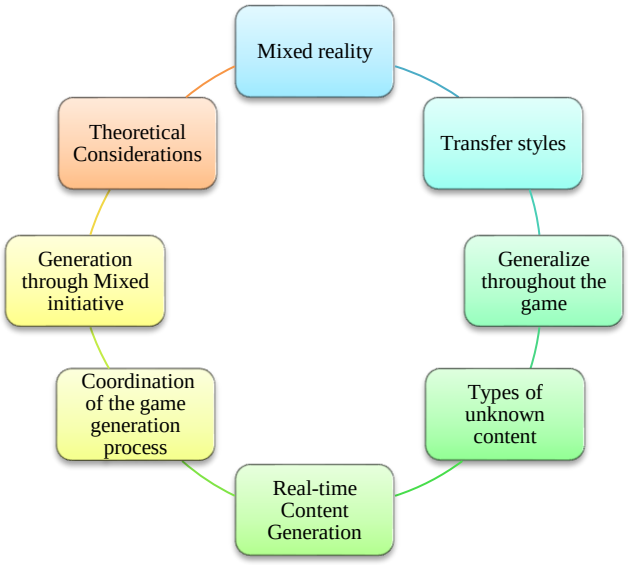

Fig. 5. Future Trends in PCG for games

B. Transfer Styles

Style transfer techniques, such as Blending, Breeding, and other methods, combined with generative models, offer significant potential for content creation in gaming, covering images, music, and sound. Moreover, deep learning approaches based on unique human-created patterns can also generate diverse input designs for these tasks. However, to date, only a few studies have explored style transfer specifically for game content. These studies have primarily focused on producing game maps inspired by terrain designs and creating artistic elements based on human-drawn sketches [9].

C. Generalize throughout the Game

Generalization in machine learning describes a model's capacity to extrapolate learned patterns to new, unseen data, ensuring robust performance in real-world scenarios. Within the realm of Procedural Content Generation (PCG), generalization involves the creation of diverse and adaptable content, such as game levels or environments, to enrich player experiences [15]. Significant research has addressed this challenge, including a study referenced in [15], where PCG techniques were employed to tackle the issue of generalization. By automatically generating varied content, PCG supplies a wealth of training data that enables models to acquire transferable skills and adapt efficiently to novel contexts. The study highlighted through demonstrations in generative models, reinforcement learning, and domain adaptation, showcases the significant role of PCG in advancing generalization capabilities in machine learning.

An effective approach to learning generative models for games, where only limited content is available, involves training on existing content from other games within the same genre. Games within a genre typically share core elements and design principles, which can be leveraged for training purposes. For example, a generator trained on First-Person Shooter (FPS) levels from games such as Quake, Halo, and Call of Duty could potentially learn to generate new levels for Half-Life. This strategy is particularly promising when applied to character models, which often share functional constraints across various open-world games. By training a generator on these existing models, it could effectively create new characters. To facilitate this, conditional generative models are utilized. These models incorporate additional inputs, such as an encoding of the target game's characteristics, along with the training data. This enables the model to discern the similarities and differences between the content it is trained on and that of the target game, thus allowing it to generate content that aligns with the specific style and mechanics of the target game. This method takes advantage of the extensive existing content across games, presenting a powerful alternative for training generative models when data for a specific game is scarce [9].

D. Types of Unknown Content

Most reviewed studies primarily focus on designing content that can be depicted through 2D images, such as tiles or pixels, which resemble elements like 2D levels, landscapes, and sprites. However, only a select few explore other dimensions of content creation, such as text and narrative generation, music and rhythm composition, and the design of weapons for FPS games, among other aspects [9].

E. Real-time Content Generation

Another underexplored area is real-time content generation. This process involves the creation of level segments during gameplay, which are dynamically adjusted based on the player's current skill level, play style, and preferences [9].

F. Coordination of the Game Generation Process

Moving forward, expanding the capabilities of PCG frameworks to encompass multiple domains of computational creativity will be a critical research area. These frameworks should integrate the six primary domains of computational game creativity: images, sound, narrative, levels, rules, and the execution process. Orchestration, which refers to the systematic coordination of these elements, ensures that outputs from two or more domain generators are harmoniously integrated to produce a complete game. Essentially, orchestration involves the management and coordination of the entire game-generation process [9].

G. Generation through Mixed Initiative

In mixed-initiative creative interfaces (MICIs), human-computer interaction unfolds as a collaborative process characterized by rapid feedback loops. These interfaces utilize algorithms to assist human designers in co-creating game content. Although current implementations of human-AI collaboration in PCG address only a subset of possible MICI applications, they demonstrate several advantageous properties. Specifically, they greatly enhance the speed of iterative

exploration within solution spaces and enable divergent exploration that would be impractically time-consuming otherwise [85].

For instance, one study, inspired by Dormans' concept of mission-space generation, explores the feasibility of PCG for creating levels based on designer-provided missions. This approach employs a generative grammar within a mixed-initiative design framework, effectively transforming a mission into a corresponding level. Its effectiveness is demonstrated through two case studies: the generation of dungeon levels for a rogue-like game and platformer levels for a Metroidvania game [85]. Additionally, another study introduces a design methodology for creating natural language interfaces for PCG systems. This methodology begins by defining a design vocabulary that can describe the generator's output, maps this vocabulary to a series of parameters, and translates natural language queries into adjustments within the generator's design space [86].

H. Theoretical Considerations

In the field of computer science, evaluating software or algorithms is crucial to ensure their effectiveness and utility. This evaluation often involves balancing performance and accuracy within PCG. Generally, algorithms that deliver precise results require greater computational power, memory, or storage capacity to manage a wider array of variations. Depending on the desired outcome, users must decide whether to prioritize performance or accuracy. A fundamental question arises: can some resources be generated with less effort than others? [3].

VII. CONCLUSION

The results derived from the reviewed sources suggest that the creation of content such as events, objectives, or personality traits, tailored to the user's skill level, can be more effectively utilized compared to other types of game content. Moreover, the real-time generation of game content that aligns with players' skills and preferences accelerates the development of personalized gaming experiences. Furthermore, based on the existing findings, the application of machine learning techniques in PCG for games with specific features has become increasingly important and beneficial. A promising direction for future research includes training game agents to collaborate with content creators, focusing on generalization across different games while considering players' motivations and addressing the challenges identified in the literature. Continued analysis and exploration in this area could resolve the current issues in PCGML.

REFERENCES

[1] L. Rodrigues, B. Robson and J. Brancher, "Procedural versus human level generation: Two sides of the same coin*?*," *International Journal of Human-Computer Studies. vol. 141,* p. 102465, 2020.

[2] M. Hafis, H. Tolle, and A. Afif Supianto, "A literature review of empirical evidence on procedural content generation in game-related implementation," *Journal of Information Technology and Computer Science. vol. 4*, pp. 308-328, 2019.

[3] J. Freiknecht and W. Effelsberg, "A survey on the procedural generation of virtual worlds," *Multimodal Technologies and Interaction. 1(4),* p.27, 2017.

[4] O. Sacco, A. Liapis, and G. N. Yannakakis, "A holistic approach for semantic-based game generation," *In 2016 IEEE Conference on Computational Intelligence and Games (CIG).* pp:1-8, 2016.

[5] S.M. Gilbert Nwankwo and J. Fiaidhi, "Procedural Content Generation for Dynamic Level Design and Difficulty in a 2D Game Using UNITY," *International Journal of Multimedia and Ubiquitous Engineering. 12(9)*, pp. 41-52, 2017.

[6] G. Smith, "An Analysis of the Role of Procedural Content Generation in Game Design," *Conference on Human Factors in Computing Systems – Proceedings*, 2013.

[7] A. Summerville, S. Snodgrass, M. Guzdial, C. Holmg˚ard, A. K. Hoover, A. Isaksen, A. Nealen, and J. Togelius, "Procedural content generation via machine learning (pcgml)," *IEEE Transactions on Games,.vol. 10*, pp. 257–270, 2018.

[8] A. Sarkar and S. Cooper, "Generating and blending game levels via quality-diversity in the latent space of a variational autoencoder," *in The 16th International Conference on the Foundations of Digital Games (FDG)*, pp. 1–11, 2021.

[9] J. Liu, S. Snodgrass, A. Khalifa, S. Risi, G. N. Yannakakis, and J. Togelius, "Deep learning for procedural content generation," *Neural Computing and Applications. 33(1)*, pp. 19-37, 2021.

[10] S. Nam and K. Ikeda, "Generation of diverse stages in turn-based roleplaying game using reinforcement learning," *in 2019 IEEE Conference on Games (CoG)*, IEEE, pp. 1–8, 2019.

[11] Z. Zhou and M. Guzdial, "Toward co-creative dungeon generation via transfer learning," *in the 16th International Conference on the Foundations of Digital Games (FDG)*, pp. 1–9, 2021.

[12] M. Gonz´alez-Hermida, E. Costa-Montenegro, B. Leger´en-Lago, and A. Pena-Gim´enez, S"tudy of artificial intelligent algorithms applied in procedural content generation in video games," *Eludamos. Journal for Computer Game Culture. 10(1)*, pp. 39–54, 2020.

[13] R.L. Cabrera, M.N. Collazo, C.C. Porras and A.J.F. Leiva, "Procedural content generation for real-time strategy games," *IJIMAI. vol. 3*, pp. 40-48, 2015.

[14] N.A. Barriga, "A short introduction to procedural content generation algorithms for videogames," *International Journal on Artificial Intelligence Tools*, 28(02), p.1930001, 2019.

[15] S. Risi, and J. Togelius, "Increasing generality in machine learning through procedural content generation," *Nature Machine Intelligence*, 2(8), pp.428-436, 2020.

[16] M. Hendrikx, S. Meijer, J. Van Der Velden, and A. Iosup, "Procedural content generation for games: A survey," *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM), 9(1)*, pp. 1–22, 2013.

[17] C. Janiesch, P. Zschech, K. Heinrich, "Machine learning and deep learning," *Electronic Markets*, 31(3), pp.685-95, 2021.

[18] A. Sarkar, and S. Cooper, "Sequential segment-based level generation and blending using variational autoencoders," *In Proceedings of the 15th International Conference on the Foundations of Digital Games*, pp:1-9, 2020.

[19] T. Georgiou and Y. Demiris, "Adaptive user modeling in car racing games using behavioural and physiological data," *User Modeling and User-Adapted Interaction. vol. 27*, pp. 267–311, 2017.

[20] Bellini, Mattia. "Towards a Model to Meet Players' Preferences in Games," *GHITALY@ CHItaly*. 2019.

[21] Z. Fang, P. Paliyawan, R. Thawonmas, and T. Harada, "Towards an angry-birds-like game system for promoting mental well-being of players using art-therapy-embedded procedural content generation," *in 2019 IEEE 8th Global Conference on Consumer Electronics (GCCE)*, IEEE, pp. 947–948, 2019.

[22] B. Li, R. Chen, Y. Xue, R. Wang, W. Li, and M. Guzdial, "Ensemble learning for mega man level generation," *in The 16th International Conference on the Foundations of Digital Games (FDG)*, pp. 1– 9, 2021.

[23] J. Osborn, A. Summerville, and M. Mateas, "Automatic mapping of nes games with mappy," *in Proceedings of the 12th International Conference on the Foundations of Digital Games*, pp. 1–9, 2017.

[24] I. Karth, B. Aytemiz, R. Mawhorter, and A. M. Smith, "Neurosymbolic map generation with vq-vae and wfc," *in The 16th International Conference on the Foundations of Digital Games (FDG)*, pp. 1–6, 2021.

[25] J. Freiknecht and W. Effelsberg, "Procedural generation of interactive stories using language models," *in International Conference on the Foundations of Digital Games*, pp. 1–8, 2020.

[26] S. Mohaghegh, M. A. R. Dehnavi, G. Abdollahinejad, and M. Hashemi, "PCGPT: Procedural Content Generation via Transformers," *arXiv preprint arXiv:2310.02405*, 2023.

[27] A. Snodgrass S. Sarkar, "A Multi-domain level generation and blending with sketches via example-driven bsp and variational autoencoders," *In Proceedings of the 15th International Conference on the foundations of digital games*, pp 1-11, 2020.

[28] Sarkar and S. Cooper, "Sequential segment-based level generation and blending using variational autoencoders," *in International Con-conference on the Foundations of Digital Games*, pp. 1–9, 2020.

[29] S. Thakkar, C. Cao, L. Wang, T. J. Choi, and J. Togelius, "Autoencoder and evolutionary algorithm for level generation in lode runner," *in 2019 IEEE Conference on Games (CoG)*, IEEE, pp. 1–4, 2019.

[30] K. Mori, A. Shinya, T. Harada, and R. Thawonmas, "Feature extraction of game plays for procedural play generation," *in 2017 Nicograph International (NicoInt)*, IEEE, pp. 86–86, 2017.

[31] V. Volz, J. Schrum, J. Liu, S. M. Lucas, A. Smith, and S. Risi, "Evolving Mario levels in the latent space of a deep convolutional generative adversarial network," *in Proceedings of the genetic and evolutionary computation conference*, pp. 221–228, 2018.

[32] R. Karp and Z. Swiderska-Chadaj, "Automatic generation of graphical game assets using gan," *in 201 7th International Conference on Computer Technology Applications*, pp. 7–12, 2021.

[33] B. J. Chelliah, V. K. Vallabhaneni, S. R. Lenkala, M. J, *and M. K. K. Reddy, "3d character generation using pcgml," International Journal of Innovative Technology and Exploring Engineering (IJITEE)*. vol. 8, pp.2278-3075, *2019.*

[34] V. Volz, N. Justesen, S. Snodgrass, S. Asadi, S. Purmonen, C. Holmg˚ard, J. Togelius, and S. Risi, "Capturing local and global patterns in procedural content generation via machine learning," *in 2020 IEEE Conference on Games (CoG)*, IEEE, pp. 399–406, 2020.

[35] G. Ivanov, M. H. Petersen, K. Kovalsk`y, K. Engberg, and G. Palamas, "An explorative design process for game map generation based on satellite images and playability factors," *in International Conference on the Foundations of Digital Games*, pp. 1–4, 2020.

[36] V. Volz, J. Schrum, J. Liu, S. M. Lucas, A. Smith, and S. Risi, "Evolving Mario levels in the latent space of a deep convolutional generative adversarial network," *in Proceedings of the genetic and evolutionary computation conference*, pp. 221–228, 2018.

[37] A. Irfan, A. Zafar, and S. Hassan, "Evolving levels for general games using deep convolutional generative adversarial networks," *in 2019 11th Computer Science and Electronic Engineering (CEEC)*, IEEE, pp. 96–101, 2019.

[38] J. Schrum, B. Capps, K. Steckel, V. Volz, and S. Risi, "Hybrid encoding for generating large scale game level patterns with local variations," *IEEE Transactions on Games. 15(1)*, pp. 46-55, 2022.

[39] L. Ciampiconi, A. Elwood, M. Leonardi, A. Mohamed, and A. Rozza, "A survey and taxonomy of loss functions in machine learning," *arXiv preprint arXiv:2301.05579*, 2023.

[40] S. Nam and K. Ikeda, "Generation of diverse stages in turn-based roleplaying game using reinforcement learning," *in 2019 IEEE Conference on Games (CoG)*, IEEE, pp. 1–8, 2019.

[41] A. Khalifa, P. Bontrager, S. Earle, and J. Togelius, "Pcgrl: Procedural content generation via reinforcement learning," *in Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, pp. 95–10, 2020.

[42] T. Shu, J. Liu, and G. N. Yannakakis, "Experience-driven PCG via reinforcement learning: A super Mario bros study," *in 2021 IEEE Conference on Games (CoG)*, IEEE, pp. 1–9, 2021.

[43] S. Liu, L. Chaoran, L. Yue, M. Heng, H. Xiao, S. Yiming, W. Li-cong, C. Ze, G. Xianghao, L. Hengtong, et al., "Automatic generation of tower defense levels using PCG," *in Proceedings of the 14th Inter--national Conference on the Foundations of Digital Games*, pp.1–9, 2019.

[44] C. Holmg˚ard, M. C. Green, A. Liapis, and J. Togelius, "Automated playtesting with procedural personas through mcts with evolved heuristics," *IEEE Transactions on Games. 11(4)*, pp. 352–362, 2018.

[45] C. Gamage, V. Pinto, C. Xue, M. Stephenson, P. Zhang, and J. Renz, "Novelty generation framework for AI agents in angry birds style physics games," *in 2021 IEEE Conference on Games (CoG)*, IEEE, pp. 1–8, 2021.

[46] J. Flimmel, J. Gemrot, and V.Černý, "Coevolution of AI and Level Generators for Super Mario Game," *In 2021 IEEE Congress on Evolutionary Computation (CEC)*, pp:2093-2100, 2021.

[47] Z. Zhou and M. Guzdial, "Toward co-creative dungeon generation via transfer learning," *in the 16th International Conference on the Foundations of Digital Games (FDG)*, pp. 1–9, 2021.

[48] J. J. Kozerawski, "Meta-learning for few-shot image classification," *University of California, Santa Barbara*, 2021.

[49] M. Goadrich and J. Droscha, "Improving solvability for procedurally generated challenges in physical solitaire games through entangled components," *IEEE Transactions on Games. vol. 12*, pp. 260–269, 2019.

[50] J. Sirota, V. Bulitko, M. R. Brown, and S. P. Hernandez, "Towards procedurally generated languages for non-playable characters in video games," *in 2019 IEEE Conference on Games (CoG)*, IEEE, pp. 1–4, 2019.

[51] D. Karavolos, A. Liapis, and G. N. Yannakakis, "Using a surrogate model of gameplay for automated level design*," in 2018 IEEE Conference on Computational Intelligence and Games (CIG)*, IEEE, pp. 1–8, 2018.

[52] K. Sorochan, J. Chen, Y. Yu, and M. Guzdial, "Generating lode runner levels by learning player paths with lstms," *in the 16th International Conference on the Foundations of Digital Games (FDG)*, pp. 1–7, 2021.

[53] Y. Liang, W. Li, and K. Ikeda, "Procedural content generation of rhythm games using deep learning methods," *in Joint Interna-National Conference on Entertainment Computing and Serious Games, Springer*, pp. 134–145, 2019.

[54] T. Tanabe, K. Fukuchi, J. Sakuma, and Y. Akimoto, "Level generation for angry birds with sequential vae and latent variable evolution," *in Proceedings of the Genetic and Evolutionary Computation Conference*, pp. 1052–1060, *2021.*

[55] Y. Zakaria, M.Fayek, and M. Hadhoud, "Procedural level generation for Sokoban via deep learning: An experimental study," *IEEE Transactions on Games*, 15(1), pp:108-120, 2022.

[56] J.A. Brown, B. Lutfullin, P. Oreshin and I. Pyatkin, "Levels for hotline Miami 2: Wrong number using procedural content generations," *Computers. 7(2)*, pp. 22, 2018.

[57] G. Smith, "Understanding procedural content generation: a design-centric analysis of the role of PCG in games," *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 917-926, *2014.*

[58] H. Goandy, "No Escape: A 2D Top-Down Shooting Roguelike Game Embedded with Drunkard Walk Algorithm," *International Journal of Advanced Trends in Computer Science and Engineering. 9(2), pp. 1045-1049*, 2020.

[59] D. Hooshyar, M. Yousefi and M. Wang, "A data-driven procedural-content-generation approach for educational games," *Journal of Computer Assisted Learning. 34(6)*, pp. 731-739, 2018.

[60] K.E. Merrick, A. Isaacs, M. Barlow and N. Gu, "A shape grammar approach to computational creativity and procedural content generation in massively multiplayer online role-playing games," *Entertainment Computing. 4(2)*, pp.115-130, 2013.

[61] Y. Dong and T. Barnes, "Evaluation of a template-based puzzle generator for an educational programming game," *In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, vol. 13*, pp.172-178, *2017.*

[62] A. Petrovas and R. Bausys, "Procedural Video Game Scene Generation by Genetic and Neutrosophic *WASPAS Algorithms," Applied Sciences. 12(2)*, p.772, 2022.

[63] L.F. Maia, W. Viana and F. Trinta, "Transposition of location-based games: using procedural content generation to deploy balanced game maps to multiple locations," *Pervasive and Mobile Computing. vol.70*, p.101302, 2021.

[64] L.F. Capasso-Ballesteros and De.la. Rosa-Rosero, "Semiautomatic construction of video game design prototypes with MaruGen," *Revista Facultad de Ingeniería Universidad de Antioquia*, vol. 99, PP.9-20, 2021.

[65] A. Isaksen, C. Holmg˚ard, J. Togelius, "Semantic hashing for video game levels," *Game & Puzzle Design*, 3(1), pp:10–16, 2017.

[66] D. Karavolos, A. Liapis, G. Yannakakis, "Learning the patterns of balance in a multi-player shooter game," *In: Proceedings of the 12th International Conference on the Foundations of Digital Games*, pp:1–10, 2017.

[67] W. Min, EY Ha, J. Rowe, B. Mott, J. Lester "Deep learning-based goal recognition in open-ended digital games," *In: Tenth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2014.

[68] S.F. Gudmundsson, P. Eisen, E. Poromaa, A. Nodet, S. Purmonen, B. Kozakowski, R. Meurling, and L. Cao, "Human-like playtesting with deep learning," *In: 2018 IEEE Conference on Computational Intelligence and Games (CIG), IEEE*, pp:1–8, 2018.

[69] M.J. Guzdial, N. Sturtevant, and B. Li, "Deep static and dynamic level analysis: A study on infinite Mario," *In: Twelfth Artificial Intelligence and Interactive Digital Entertainment Conference*, 2016.

[70] C. Holmg˚ard, A. Liapis, J. Togelius, and G.N. Yannakakis "Evolving personas for player decision modeling," *In: 2014 IEEE Conference on Computational Intelligence and Games, IEEE*, pp 1–8, 2014.

[71] G. Smith, “An Analog History of Procedural Content Generation,” *In FDG*. 2015.

[72] Z. Wang, J. Liu, and G.N. Yannakakis, “The Fun Facets of Mario: Multifaceted Experience-Driven PCG via Reinforcement Learning,” *In Proceedings of the 17th International Conference on the Foundations of Digital Games*, pp 1-8, 2022.

[73] L.T. Pereira, P.V. de Souza Prado and R.M. Lopes, “Procedural generation of dungeons’ maps and locked-door missions through an evolutionary algorithm validated with players,” *Expert Systems with Applications*. vol. 180, p.115009, 2021.

[74] M.A. Muslim, E.M.A. Jonemaro and M.A. Akbar, “Penerapan Procedural Content Generation untuk Perancangan Level pada 2D Endless Runner Game menggunakan Genetic Algorithm,” *Jurnal Pengembangan Teknologi Informasi dan Ilmu Komputer*. *3(5)*, p. 4406-14, 2019.

[75] J. Doran, “Procedural generation of content for online role playing games,” *University of North Texas*, 2014.

[76] A. Chia, "The artist and the automaton in digital game production", *Convergence*, 2022.

[77] A.M. Abuzuraiq, O. Alsalman and H. Erhan, “Shopping for Game levels: A Visual Analytics Approach to Exploring Procedurally Generated Content,” *In International Conference on the Foundations of Digital Games*, pp. 1-4, 2020.

[78] A.J. Smith and J.J. Bryson, “A logical approach to building dungeons: Answer set programming for hierarchical procedural content generation in roguelike games,” *In Proceedings of the 50th Anniversary Convention of the AISB*, 2014.

[79] R. Dey and J. Konert, “Content Generation for Serious Games,” *In Entertainment Computing and Serious Games*, pp. 174-188, 2016.

[80] S. Freiberg, “Procedural generation of content in video games,” Ph.D. dissertation, *Hochschule für angewandte Wissenschaften Hamburg*, 2016.

[81] E.A. BIYIK and E.L.İ. F.Sürer, “Developing a Space Syntax-Based Evaluation Method for Procedurally Generated Game Levels,” *Mugla Journal of Science and Technology*. *6(2)*, pp. 79-88, 2020.

[82] F. Mehm, A. Hendrich, D. Kauth and S. Göbel, “Procedural Content Generation in Educational Game Authoring Tools,” *In ECGBL2014-8th European Conference on Games Based Learning: ECGBL2014*, p. 380, 2014.

[83] R. Mawhorter, B. Aytemiz, I. Karth and A. Smith, “Content reinjection for super Metroid,” *In Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment, vol. 17*, pp. 172-178, *2021*.

[84] D. Karavolos, A. Bouwer and R. Bidarra, “Mixed-Initiative Design of Game Levels: Integrating Mission and Space into Level Generation,” *In FDG*. 2015.

[85] A. Mobramaein, and J. Whitehead, “A methodology for designing natural language interfaces for procedural content generation,” *In Proceedings of the 14th International Conference on the Foundations of Digital Games*, pp. 1-9, 2019.