2D Map Generation for Games Using Generative Adversarial Networks

Victor Le Roy Matos[1], **Rogério M. Gomes**[1], **João Gabriel Gama Vila Nova**[2]

[1]Departamento de Computação
CEFET-MG*– Belo Horizonte, MG – Brasil.

[2]Departamento de Computação
UNIMA Afya – Maceió, AL – Brasil

victorlrmatos@gmail.com, rogerio@cefetmg.br, jggvilanova@gmail.com

Abstract. Introduction*: Video games have evolved from simple pastimes into a culturally significant art form, driven by technological advances that have enabled more realistic graphics and immersive gameplay.* ***Objective****: This study explores the use of Generative Adversarial Networks (GANs) for generating maps in 2D games, focusing on Super Mario Bros.* ***Methodology****: The methodology includes the use of a Wasserstein GAN (WGAN), combined with a fragment selection algorithm and gameplay evaluation performed by an A* agent.* ***Results****: The results indicate a significant improvement in map quality, with a 25% increase in the rate of playable fragments and a 15% reduction in the agent's average evaluation time, ensuring greater adherence to the characteristics of the original game.*

Keywords *Map Generation, Geração de mapa, GANs, Super Mario Bros., PCG.*

1. Introduction

Since their inception, electronic games have evolved significantly, transitioning from simple forms of entertainment to a widely recognized cultural and artistic expression [da Manhã 2023]. This evolution has been driven by technological improvements, such as more realistic graphics, immersive gameplay, and innovations in artificial intelligence (AI), which enable more sophisticated interactive experiences [Reis e Andrade 2023]. Furthermore, the growing popularity of online communities and streaming platforms has amplified the cultural impact of games [Siqueira 2023] [Andrade 2023] [Machado 2022].

Among the various areas of innovation, Procedural Content Generation (PCG) stands out for its potential to create unique and personalized experiences for players. In this context, Generative Adversarial Networks (GANs) have emerged as a promising solution, enabling the generation of game maps and levels that respect critical aspects such as playability and creative design [Goodfellow et al. 2020] [Cheigh 2023]. However, applying GANs to 2D games presents specific challenges. One is ensuring that the generated maps are coherent in terms of structure and gameplay. Another challenge is the need to adhere to the specific rules of each game, such as block layout and the distribution of interactive elements [Volz et al. 2018] [Awiszus et al. 2020]. These issues necessitate exploring new architectures and algorithms to optimize the results obtained by GANs.

[1]The authors would like to thank the Centro Federal de Educação Tecnológica de Minas Gerais (CEFET-MG) for the institutional support and the conditions offered for the development of this research.

Among 2D games, Super Mario Bros. is frequently used as a case study due to its combination of graphical simplicity and structural complexity [David J. Malan 2021] [Aloupis et al. 2015], offering an ideal testing ground for level generation algorithms [deWinter 2015]. This study directly confronts the problem of structural incoherence in GAN-generated levels by proposing a modular, filter-based pipeline. Unlike end-to-end models like TOAD-GAN that build constraints into the architecture [Awiszus et al. 2020], our approach decouples generation from validation. Our central contribution is the systematic analysis of this post-hoc filtering methodology, where a rule-based selection algorithm and an A* agent are used to enforce design constraints. We demonstrate that this method significantly improves the structural integrity and playability of levels generated by a standard WGAN.

2. Related Work

PCG gained prominence with the 2010 PCG competition, which used Super Mario Bros. as a case study [Shaker et al. 2011] [Khalifa 2021]. This initiative spurred various academic studies focusing on AI-based approaches, especially GANs, for creating levels with specific attributes. Among relevant studies, TOAD-GAN [Awiszus et al. 2020] used multiple scales to preserve essential map features, while works based on Variational Autoencoders (VAEs) [Kingma e Welling 2019] demonstrated the possibility of blending elements from different games.

In the context of GANs, Silva, Torchelsen, and Aguiar [e Silva et al. 2023] compared different architectures applied to generating dungeon maps. The analyzed architectures included Vanilla GAN, Deep Convolutional GAN (DCGAN), and Wasserstein GAN (WGAN). Their results showed that applying Spectral Normalization significantly improved the playability and cohesion of generated levels.

The TOAD-GAN model, proposed by Awiszus, Schubert, and Rosenhahn [Awiszus et al. 2020] and based on the work of Shaham, Dekel, and Michaeli [Shaham et al. 2019], introduced a multi-scale training approach. Qualitative analysis showed that 65% of its generated levels were completed by an A* agent implemented according to Baumgarten [Togelius et al. 2010].

Schrum et al. [Volz et al. 2018] [Schrum et al. 2020] also advanced the field. Their first study [Volz et al. 2018] used a DCGAN and latent space evolution to optimize Super Mario levels, evaluated with an A* agent from Shaker [Shaker et al. 2011]. Their second study [Schrum et al. 2020] focused on interactive map evolution using WGANs.

The existing literature shows two dominant trends: modifying the GAN architecture itself (e.g., TOAD-GAN) or evolving within the GAN's latent space (e.g., Volz et al.). Our work proposes a complementary, more modular alternative. We focus on a post-hoc rule-based filtering and assembly methodology. Instead of altering the core generative model, we investigate the impact of applying explicit design constraints after generation. This approach has a practical advantage: it can be applied to various pre-trained generative models without the need for complete retraining, providing greater flexibility and efficiency in extracting usable content from a generator's raw output.

3. Methodology

Our methodology focuses on a modular pipeline for 2D map generation using GANs and a post-hoc selection filter.

3.1. Data and Preprocessing

We used the 150 Super Mario Bros. levels from the Video Game Level Corpus (VGLC) [Summerville et al. 2016]. Maps were encoded numerically (Table 1) and subdivided into 28x14 block fragments, which were then padded to 32x32 for the network.

Table 1. Encoding of game blocks with visualization.

Block	Symbol	Identity	Visualization
Solid block/Ground	X	0	
Breakable block	S	1	
Empty space (passable)	-	2	
Filled question block	?	3	
Empty question block	Q	4	
Enemy	E	5	
Top-left pipe	<	6	
Top-right pipe	>	7	
Left pipe	[	8	
Right pipe	]	9	

3.2. Map Generation and Selection

A Wasserstein GAN (WGAN) [Arjovsky et al. 2017] [Gulrajani et al. 2017] was trained on the fragments to generate 10,000 new level fragments. A selection algorithm filters these fragments using a fitness function that penalizes structural flaws. The three core criteria are: pipe integrity, enemy placement, and hole size. The final fitness is a weighted sum of these scores. The top 10% (1,000) of fragments with the best fitness scores are selected for the next stage. The penalty coefficients of the three core criteria were defined empirically with a value of 0.2. This value was determined through iterative executions of the selection process, where hyperparameters were tuned to achieve satisfactory performance in filtering out structurally flawed fragments was achieved.

3.3. Assembly and A* agent Evaluation

To construct a full level, 10 fragments are assembled sequentially through horizontal concatenation. A two-step validation process ensures playability: first, a quick geometric check for ground-level alignment between fragments, followed by a full-path validation using an A* agent. This agent, based on the framework by Khalifa et al. [Khalifa 2021], attempts to traverse the assembled map within a 120-second time limit. If a fragment renders the map unplayable, it is discarded. For our experiments, we created two sets of fragments: Group 1 (only fragments individually completable by the agent) and Group 2 (all 1,000 filtered fragments). In our evaluation, a 'jump' is counted each time the agent's vertical position increases. The goal of minimizing jumps, as explored by Schrum et al. [Schrum et al. 2020], serves as a proxy for level efficiency and flow. An 'unnecessary' jump, in this context, is one occurring in a poorly designed section where a simpler, flatter path could exist. A high jump count often correlates with convoluted or vertically noisy level design.

4. Results and Discussion

To evaluate our method, we conducted two experiments: one using the raw WGAN output and another using the fragments filtered by our selection algorithm. For each, we assembled maps using fragments from both Group 1 and Group 2.

The results show that the selection algorithm significantly improved the generation success rate. When using only pre-vetted, playable fragments (Group 1), the success rate for assembling a full, 10-fragment map rose from 17.6% (pure WGAN) to 64% (WGAN + Selection). This indicates that the selection algorithm is effective at identifying fragments that are not only are internally valid but also combine well with others. The statistical details are presented in Tables 2 and 3.

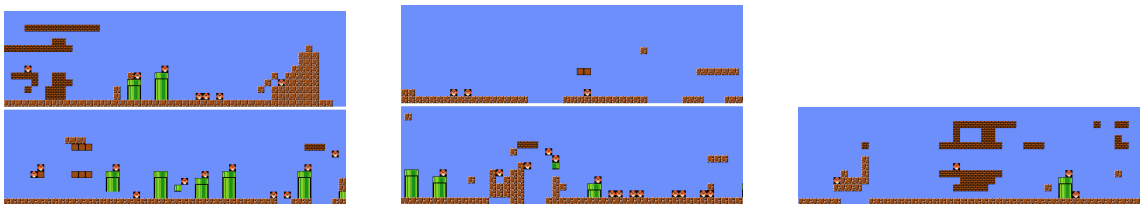
Table 2. Number of generated maps (WGAN + Selection) by difficulty.

	Easy (0-0.3)	Medium (0.3-0.7)	Hard (0.7-1)	Total
Group 1	36	27	1	64
Group 2	11	9	1	21

Table 3. Statistical results for the WGAN with selection experiment.

	Elapsed time (s)				Enemies Eliminated			
	Max	**Min**	**Mean**	**STD**	**Max**	**Min**	**Mean**	**STD**
Group 1	26	12	13.2	1.8	10	1	4.5	2.5
Group 2	28	12	13.57	3.26	8	1	4.28	1.95
	Jumps				**Difficulty (0-1)**			
	Max	**Min**	**Mean**	**STD**	**Max**	**Min**	**Mean**	**STD**
Group 1	36	10	19.5	0.14	0.91	0.069	0.29	0.142
Group 2	47	10	20.24	6.67	0.77	0.08	0.3	0.153

Empirically, the filtered fragments show far fewer structural errors. Figure 1 illustrates this improvement, showing how the selection algorithm corrects issues like illogical pipe placement and floating enemies, although some nonsensical structures may still appear.



(a) Logical pipe placement.

(b) Reduced structural errors.

(c) Remaining illogical structure.

Figure 1. Side-by-side comparison of map fragments improved by the selection algorithm.

4.1. Discussion and Limitations

Our results confirm that post-generation filtering is a viable method for improving the quality of GAN-produced levels. However, the study has limitations. Our evaluation does

not include a direct comparison with state-of-the-art models like TOAD-GAN or a simple search-based heuristic. Furthermore, the method is tailored to tile-based 2D platformers. Adapting this method to a different genre, such as a scrolling beat-'em-up, would require substantial changes. The tile semantics would shift from platforms to enemy spawn points, and the A* agent's goal would change from path-completion to combat survival, necessitating a completely new fitness function and evaluation framework. Finally, the evaluation relies solely on a single A* agent, whose performance might not fully reflect human playability, and our difficulty metric was not validated against human player studies.

5. Conclusion and Future Work

This paper presents a modular pipeline for improving GAN-based level generation[2]. Our results show that a post-hoc filtering approach is a viable and effective method for improving the structural quality and playability of generated levels. By fixing design flaws after generation, we significantly increase the yield of usable content from a standard WGAN.

Future work will address the limitations of this study. A crucial next step is to perform a direct quantitative comparison of our method's output against both a state-of-the-art model like TOAD-GAN and a simpler, non-learning baseline, such as a search-and-repair heuristic, to rigorously benchmark its effectiveness. We also plan to refine the fitness function to promote a wider range of difficulty levels and to incorporate human player feedback to create a more robust and generalizable validation process.

[2]Source code and supplementary materials for this paper are available at the project's GitHub repository. [Le Roy Matos 2025]

References

Aloupis, G., Demaine, E. D., Guo, A., e Viglietta, G. (2015). Classic nintendo games are (computationally) hard. *Theoretical Computer Science*, 586:135–160. Fun with Algorithms.

Andrade, E. S. (2023). Games e a indústria cultural: O impacto sociocultural dos jogos eletrônicos. *Repositório Institucional - Universidade Federal de Uberlândia: Home*.

Arjovsky, M., Chintala, S., e Bottou, L. (2017). Wasserstein generative adversarial networks. In Precup, D. e Teh, Y. W., editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 214–223. PMLR.

Awiszus, M., Schubert, F., e Rosenhahn, B. (2020). Toad-gan: Coherent style level generation from a single example. *ArXiv*.

Cheigh, J. (2023). Generating images using vaes, gans, and diffusion models. https://towardsdatascience.com/generating-images-using-vaes-gans-and-diffusion-models-48963ddeb2b2. [Accessed 09-03-2024].

da Manhã, D. (2023). 4 importantes transformações tecnológicas nos jogos eletrônicos. https://www.dm.com.br/cotidiano/4-importantes-transformacoes-tecnologicas-nos-jogoseletronicos-127327. [Accessed 19-02-2024].

David J. Malan, D. L. (2021). Csci e-23a - cs50. https://cs50.harvard.edu/extension/games/2021/fall/syllabus/. [Accessed 09-03-2024].

deWinter, J. (2015). *Shigeru Miyamoto: Super Mario Bros., Donkey Kong, The Legend of Zelda*. Influential Video Game Designers. Bloomsbury Publishing.

e Silva, D. F., Torchelsen, R., e Aguiar, M. (2023). Dungeon level generation using generative adversarial network: an experimental study for top-down view games. In *Anais do L Seminário Integrado de Software e Hardware*, pages 95–106, Porto Alegre, RS, Brasil. SBC.

Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., e Bengio, Y. (2020). Generative adversarial networks. *Commun. ACM*, 63(11):139–144.

Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., e Courville, A. (2017). Improved training of wasserstein gans. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, page 5769–5779, Red Hook, NY, USA. Curran Associates Inc.

Khalifa, A. (2021). Mario-ai-framework. https://github.com/amidos2006/Mario-AI-Framework?tab=readme-ov-file#copyrights. [Accessed 27-07-2024].

Kingma, D. P. e Welling, M. (2019). An introduction to variational autoencoders. *Foundations and Trends® in Machine Learning*, 12(4):307–392.

Le Roy Matos, V. (2025). mario-maps-generation-gans-selection: 2d map generation for games using generative adversarial networks. https://github.com/vmleroy/mario-maps-generation-gans-selection.

Machado, R. P. T. (2022). Jogos eletrônicos e e-sports: Um fenômeno cultural. https://ciencia.ufla.br/todas-opinioes/860-jogos-eletronicos-e-e-sports-um-fenomeno-cultural. [Accessed 19-02-2024].

Reis, M. d. L. e Andrade, K. d. O. (2023). Áreas de pesquisa e técnicas de inteligência artificial em jogos digitais. *Revista Tecnológica Fatec Americana*.

Schrum, J., Gutierrez, J., Volz, V., Liu, J., Lucas, S., e Risi, S. (2020). Interactive evolution and exploration within latent level-design space of generative adversarial networks. In *Proceedings of the 2020 Genetic and Evolutionary Computation Conference*, GECCO '20, page 148–156, New York, NY, USA. Association for Computing Machinery.

Shaham, T., Dekel, T., e Michaeli, T. (2019). Singan: Learning a generative model from a single natural image. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 4569–4579, Los Alamitos, CA, USA. IEEE Computer Society.

Shaker, N., Togelius, J., Yannakakis, G. N., Weber, B., Shimizu, T., Hashiyama, T., Sorenson, N., Pasquier, P., Mawhorter, P., Takahashi, G., Smith, G., e Baumgarten, R. (2011). The 2010 mario ai championship: Level generation track. *IEEE Transactions on Computational Intelligence and AI in Games*, 3(4):332–347.

Siqueira, M. S. (2023). Esports e o ambiente acadêmico: uma perspectiva sobre o cenário esports. *Revista UFF*.

Summerville, A. J., Snodgrass, S., Mateas, M., e Villar, S. O. (2016). The VGLC: The video game level corpus. *CoRR*, abs/1606.07487.

Togelius, J., Karakovskiy, S., e Baumgarten, R. (2010). The 2009 mario ai competition. In *IEEE Congress on Evolutionary Computation*, pages 1–8.

Volz, V., Schrum, J., Liu, J., Lucas, S. M., Smith, A., e Risi, S. (2018). Evolving mario levels in the latent space of a deep convolutional generative adversarial network. In *Proceedings of the Genetic and Evolutionary Computation Conference*, GECCO '18, page 221–228, New York, NY, USA. Association for Computing Machinery.