Neurosymbolic Map Generation with VQ-VAE and WFC

ISAAC KARTH, BATU AYTEMIZ, ROSS MAWHORTER, and ADAM M. SMITH, University of California, Santa Cruz, USA

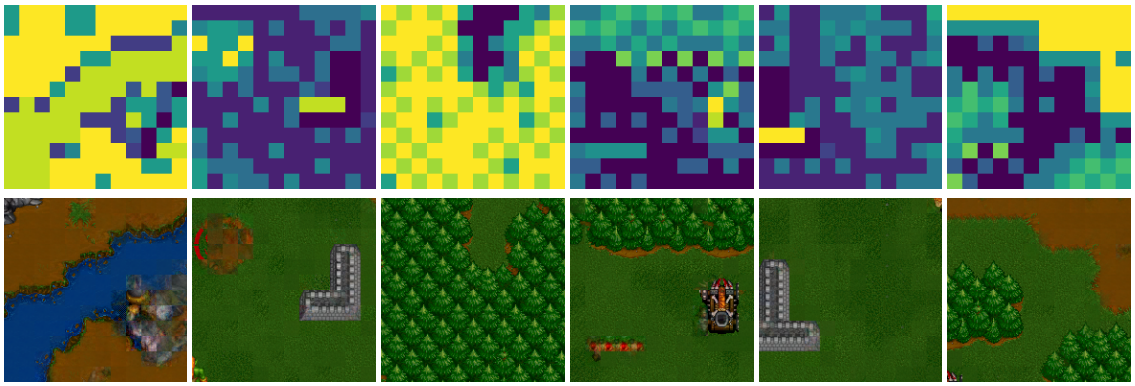

Fig. 1. Unconstrained WFC + Neural Decoder outputs. Note the directional transitions between terrain types, units spanning multiple tiles, and an imperfectly reconstructed building (only one instance of this building is seen in the training input). Top visualizes the latent tile selected at each location (from a vocabulary of just 12 latent codes) while bottom shows synthesized color images.

We introduce a hybrid neural + symbolic approach to map generation that combines neural discrete representation learning with symbolic constraint solving methods. In application to *WarCraft II* and *Super Metroid* map designs, we show how a vocabulary of directly manipulable latent tiles can be inferred from the raw pixels of design training data. Despite working with a very small tile vocabulary, our method is able to express a very large effective set of unique tiles at the level of pixel appearances. This work shows new ways of combining generative methods, resulting in directly controllable generators for domains that are primarily specified only by visual design examples.

CCS Concepts: • **Applied computing** → **Computer games**; • **Computing methodologies** → *Logical and relational learning*; *Neural networks*.

Additional Key Words and Phrases: procedural content generation, autoencoders, vector quantization, constraints

ACM Reference Format:
Isaac Karth, Batu Aytemiz, Ross Mawhorter, and Adam M. Smith. 2021. Neurosymbolic Map Generation with VQ-VAE and WFC. In *The 16th International Conference on the Foundations of Digital Games (FDG) 2021 (FDG'21), August 3–6, 2021, Montreal, QC, Canada.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3472538.3472584

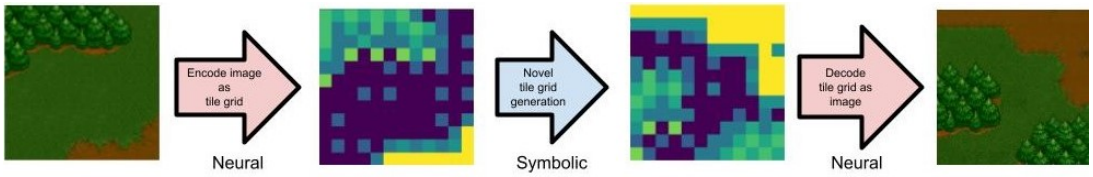

Fig. 2. Overview of our neurosymbolic map generation approach. Grid generation can accept additional constraints on the tile grid.

1 INTRODUCTION

Image generation systems based on *neural* networks have been able to create convincing, high-fidelity images, from faces [8] to animals [1]. This capability has been of interest to the game content generation community [11]. However, one common shortcoming of these learning based methods is that they are difficult to control. Simply asking designers to provide more and more examples of the desired kind of content is not a satisfying alternative [10]. Controllability is especially important when it comes to generating videogame maps. In order to support specific player experiences, a generated map must obey hard constraints [13]. *Symbolic* artificial intelligence (AI) methods such as constraint-solving offer the ability to directly enforce key properties on the output of a generator [14], but it is challenging to combine them with neural techniques.

In this paper we introduce a *neurosymbolic* [4] approach to generating videogame map images in which discrete representation learning methods (VQ-VAEs) [18] are used to produce a vocabulary of latent tile descriptors. Novel arrangements of tiles produced by a constraint-based map generator (WFC) [9] are rendered in a context-sensitive way, allowing the expression of a large effective tileset despite working with a small latent vocabulary. Figure 1 teases results of applying our method to a *WarCraft II* map setting.

2 APPROACH: VQ-VAE + WFC

Using the WaveFunctionCollapse (WFC) algorithm (a family of symbolic constraint-solving methods) to generate a new map image requires a carefully curated tileset. The input map is described using indexes into that tileset and, after generating a new map of tile indexes, they can be replaced with the corresponding tile image to create a map image. Our method (sketched in Figure 2) obviates the need for a tileset by learning tiles using a vector-quantizing variational autoencoder (VQ-VAE). The learned VQ-VAE model provides both a mapping from patches of the original image to the corresponding tile indices, as well as a (context-sensitive) mapping from the tile index grid back to image pixels.

A traditional autoencoder (AE) learns to summarize input images using a bottleneck representation consisting of a vector of continuous values. A vector-quantizing (VQ) autoencoder, on the other hand, uses discrete integers as the bottleneck representation. A trainable codebook supplies the vectors used in the decoder, and encoded vectors are compressed by assigning them the integer index into the codebook of the vector most similar to them by Euclidean distance.

In the training phase for our VQ-VAE model (shown in Figure 3), large patches of the training design image are passed through a convolutional encoder model that reduces the image into a low-resolution grid of high-dimensional vectors. Each vector on this grid is used to lookup the nearest codebook vector. These quantized vectors are assembled into a grid the same shape as the input to the quantization process. After this, a convolutional decoder model transforms the grid into a high-resolution color image. The loss function forces this to resemble the original input. For more detail

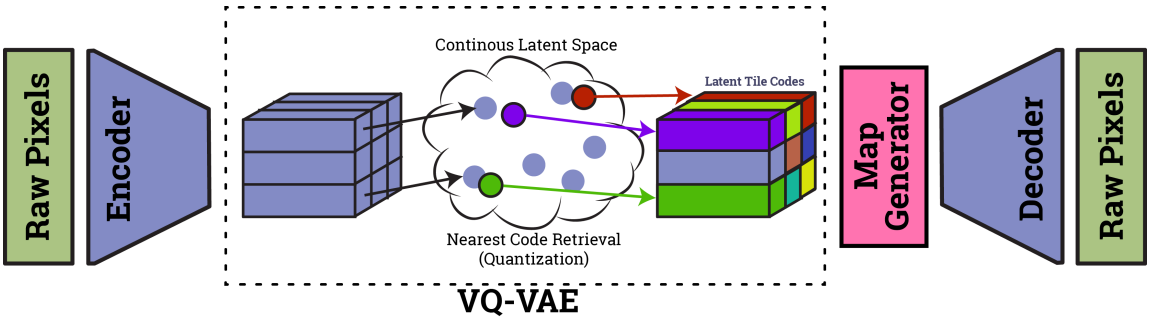

Fig. 3. Overview of our VQ-VAE used to compress continuous latent vectors into a discrete latent tile code vocabulary. The latent tile codes are then used by the Wave Function Collapse algorithm to generate novel maps.

on the use of VQ-VAE to learn tiles for image generation, the reader should consult the upstream machine learning literature [18].

Once training is complete, we can use the learned (quantizing) encoder to convert the large training design image into a tile index grid using the latent tile vocabulary. This grid is used by WaveFunctionCollapse to generate novel tile index grids. WFC consists of two parts: the adjacency learning, which translates the training data into constraint rules, and the constraint solver, which generates a new configuration of rules that satisfies the constraints. For adjacency learning, we used standard overlapping WFC with 2×2 tile patterns, which defines the constraint rules as patterns of tile neighborhoods [9]. The solver then finds a solution that satisfies the rules, generating a new grid of tile indices. Using the trained (dequantizing) decoder,[1] we can synthesize novel high-resolution pixel grids (Fig. 6).

3 DEMONSTRATIONS

To show how this method can generate novel levels, we apply it to level images of two tile-based games: *Warcraft II* and *Super Metroid*. A separate VQ-VAE model (encoder + decoder) is trained for each map.

3.1 WarCraft II

Warcraft II is a real-time strategy game released in 1995 by Blizzard Entertainment. Images of Warcraft II levels[2] were used to demonstrate VQ-VAE + WFC. Figure 4 shows the intermediate (quantized) latent tile representation of the training design image and the corresponding output of the trained VQ-VAE model in reconstructing that data. Although some of the textured detail of the tiles is missing, the overall image is quite similar to the input. Note how the light blue "land" tile next to the dark blue "shore" tile successfully renders all needed cases of a coastline.

Figure 1 shows some results of our approach. Contextualized rendering allows many important effects. The different terrain types blend together with transitions that depend on the surrounding tiles, the direction of the border, and the neighboring terrain types. Because of this, many different tile images can be represented with only 12 latent tiles.[3] Consider the two yellow tiles in Figure 4. Even though they are represented by the same index, they are reconstructed into different pixel images–one for a "left facing coast" tile and the other a "right facing coast" tile. This capability

[1] The *decoder* in an autoencoder works analogously to the *generator* in GAN models.
[2] https://vgmaps.com/Atlas/PC/WarCraftII-BeyondTheDarkPortal-Humans-Mission01-Alleria'sJourney.png
[3] A 3×3 grid with 12 possible placement in each location gives rise to billions of possible combinations, some of which are directly supervised by the training data and others to be covered by the decoder network's generalization abilities.

allows us to reconstruct maps that respect tile transitions and unit placements with a comparatively low number of distinct tile codes.

3.2 Super Metroid

Super Metroid is a platformer released in 1994 by Nintendo. As with *Warcraft II*, we use a single level image to train the VQ-VAE network, and the resulting encoded image is used to create the tile sets for WaveFunctionCollapse (see Figure 5). Fewer latent codes were used in Metroid (8 vs. 12), since the in-game image representation of any given tile does not depend on neighboring tiles. Figure 6 shows novel level images generated using WaveFunctionCollapse.

However, unlike the *Warcraft II* examples, these level images were generated under specific constraints that certain tiles be part of a door or an item pedestal. Given a partial grid of latent tile codes, WFC can correctly complete the blank regions. To control the output (placing a door or an item in a certain location), a user can first use the encoder on the original input to find the arrangement of latent tiles for a given feature. Then, the user can provide a partial grid, with those latent tiles at the desired position. These constraints influence the latent tile grid generated by WFC without changing the way that the renderer creates the level image.

4 RELATED WORK

By positioning our approach as *neurosymbolic*, we contrast it with previous symbolic and neural approaches to map generation. This section briefly relates our work to these methods.

4.1 Symbolic PCG Methods

Maxim Gumin's WaveFunctionCollapse (WFC) [5] is an obvious precedent for this work. Karth and Smith [9] interpreted WFC as an instance of constraint solving methods for PCG, describing a rational reconstruction of WFC on top of the constraint-programming technology answer-set programming (ASP). Earlier, Smith and Mateas described a general approach to procedural content generation rooted in symbolic AI. In their paradigm, symbols and rules define a *design space model* that declaratively captures the space of all designs that might be appropriate to a scenario. Constraint-solving methods are then used to sample designs from this space that satisfy all modeled constraints. Where constraints are used in other PCG systems [3, 7, 15] they relate explicitly defined symbols or predicates operating on them. In this paper, constraints over the placement of one tile next to another in the grid are handled by WFC.

Fig. 4. The autoencoder model reconstructs colored images after passing them through a bottleneck of discrete latent codes. Left, original data from WarCraft II map. Middle, the discrete latent variables in the bottleneck layer of the VQ-VAE. Right, the reconstructed image from the latent variables.

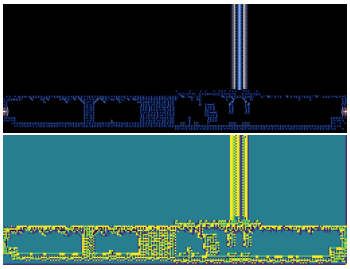

Fig. 5. Selected segment of the Super Metroid map. Left shows source pixels while right shows assigned latent codes (8 latent codes).

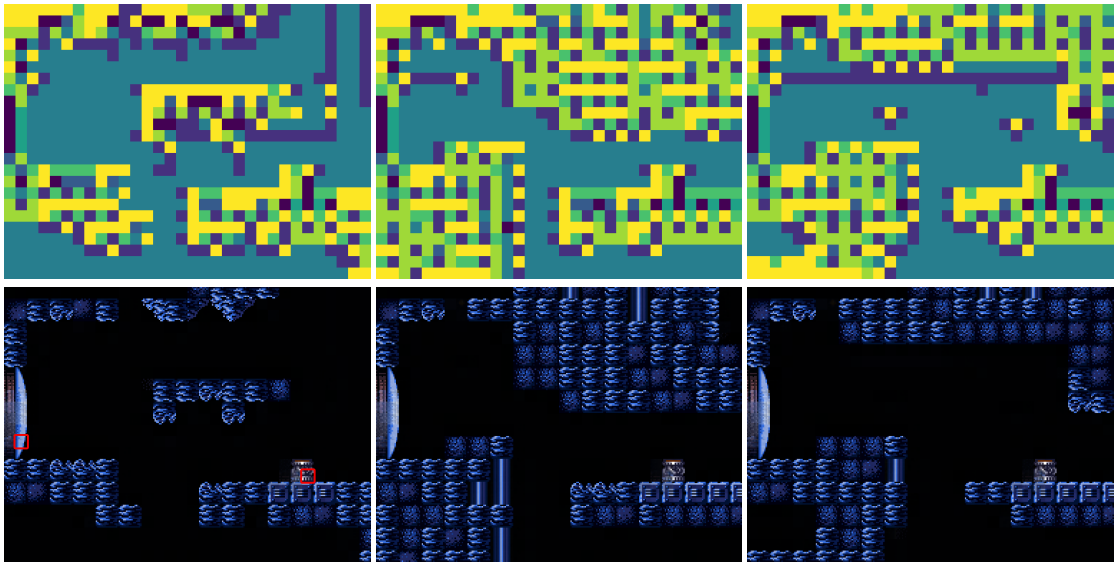

Fig. 6. Constrained WFC + Neural Decoder outputs from *Super Metroid* data. One specific grid location has been constrained to have the appearance of a door tile while the other is constrained to be a power-up pedestal (outlined in red in left image).

Karth and Smith separately interpreted WFC as an instance of machine learning methods for PCG [10]. Under the assumption that the input was already composed of recombinable tiles, WFC trivially learns which tiles may be placed adjacent to one another. The idea of using adjacencies observed in the training data to influence adjacencies that will be observed in generator's output is also present in the multi-dimensional Markov chain (MdMC) work of Snodgrass and Ontoñón [16]. By constrast, most work in procedural content generation via machine learning (PCGML) has opted to use *neural* learning techniques.

4.2 Neural PCG Methods

A recent survey of deep learning for procedural content generation [11] reviews several neural approaches to example-driven PCG. Among these, our current work makes useful contrast with past neural approaches to *Mario* level generation. Whether using a long short-term memory (LSTM) [17] or generative adversarial network (GAN) [20], these approaches directly trained a *neural generator* that used continuous representations during the sampling of discrete output tiles. While these approaches cleanly stayed within a single paradigm of AI, they limited the ability to directly and transparently apply constraints to the desired outputs.

Outside of procedural content generation for games, the larger computer graphics and computer vision communities are increasingly adopting *discrete* neural representation learning methods [18]. Even though image pixel data might be considered continuous, transforming it into inherently discrete representations allows methods originally developed for natural language processing (such as transformer networks [19]) to be cross-applied. In light of recent high-resolution image generation models that use learned discrete tile representations [2, 12], the use of WFC in our work shows constraint-solving methods as a symbolic alternative to the reasoning implicitly happening inside of transformer networks.

5 FUTURE WORK

Among a number of diverging next steps for this work, we wish to highlight two that tighten the integration between the neural and symbolic parts.

Improving manipulability and recombinability of symbols: In the history of AI [6], symbols were associated with representations that could be manipulated and recombined according to explicit rules, without consideration for the symbols' semantics. In the future, we might make the tile adjacency validity matrix (which functions as a nondeterminstic ruleset for WFC) be a trainable part of the neural model and use specific neural architectures or losses to encourage the matrix to have certain kinds of regularity or sparsity. We also might incorporate ideas from generative adversarial networks (GANs) to make sure that unseen-but-legal arrangements of latent codes produce rendered outputs that are similar to those seen in the training data. The goal is to make it so that any arrangement of symbols allowed under the extracted constraints leads to plausible renderings.

Richer symbol grounding: Beyond asking the autoencoder network to reconstruct an image of a game map design, we can ask the network to reconstruct additional data associated with the map (e.g. the precise tile data used by game engines, presence and properties of game objects on the map, historical player behavior data, etc.). This may allow reasoning about properties we want to constrain (such as reachability) using only the latent tile representation. We imagine these additional signals to be presented to the autoencoder model as additional feature channels beyond the red/blue/green pixel brightnesses currently modeled. By customizing the architecture of the decoder network (e.g. changing the receptive field of certain outputs), we can express that some inherent features of a tile should be decodable without any context whereas others might only emerge in consideration for broader regions. By grounding the latent symbols in a richer semantic frame, the resulting generator might also be able to output a high-level interpretation of a map design alongside the low-level data structures needed to represent that map in a target file format.

6 CONCLUSION

By training a VQ-VAE we can learn mappings for large source images, translating them into a latent representation. The latent representation is used as input into WFC's adjacency learning, defining the constraint rules that the solver satisfices. Once the solver has come up with an acceptable solution, we translate the new configuration of latent tiles back into the original pixel map via the VQ-VAE decoder, producing a context-sensitive rendering to the pixel image.

Constraint-based procedural content generation using symbolic methods has the advantage of being relatively easy to control, especially when the desired uses have hard constraints that can be specified as rules. In contrast, deep neural networks have demonstrated amazing generative abilities, but are difficult to control. This makes them much less useful in a production environment, particularly for videogames, in which many content types have constraints: an amazing level generator is not useful if the player cannot reach the end of the level. We demonstrate that a neurosymbolic approach, combining discrete representation learning methods with a constraint-based generator, allows the expression of large effective tilesets while offering enhanced controllability.

REFERENCES

[1] Andrew Brock, Jeff Donahue, and Karen Simonyan. 2018. Large Scale GAN Training for High Fidelity Natural Image Synthesis. (Sept. 2018). arXiv:1809.11096 [cs.LG] http://arxiv.org/abs/1809.11096

[2] Patrick Esser, Robin Rombach, and Björn Ommer. 2020. Taming Transformers for High-Resolution Image Synthesis. *arXiv preprint arXiv:2012.09841* (2020).

[3] Leif Foged and Ian D Horswill. 2015. *Rolling Your Own Finite-Domain Constraint Solver.* A K Peters/CRC Press, 283–302.

[4] Artur D'avila Garcez and Luis C Lamb. 2020. Neurosymbolic AI: The 3rd Wave. (Dec. 2020). arXiv:2012.05876 [cs.AI] http://arxiv.org/abs/2012.05876

[5] Maxim Gumin. 2016. WaveFunctionCollapse. https://github.com/mxgmn/WaveFunctionCollapse. *GitHub repository* (2016). https://github.com/mxgmn/WaveFunctionCollapse

[6] Stevan Harnad. 1990. The symbol grounding problem. *Physica D: Nonlinear Phenomena* 42, 1-3 (1990), 335–346.

[7] Ian Horswill. 2019. Imaginarium: A Tool for Casual Constraint-Based PCG. In *Proceedings of the AIIDE Workshop on Experimental AI and Games (EXAG)*.

[8] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. 2020. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. openaccess.thecvf.com, 8110–8119. http://openaccess.thecvf.com/content_CVPR_2020/html/Karras_Analyzing_and_Improving_the_Image_Quality_of_StyleGAN_CVPR_2020_paper.html

[9] Isaac Karth and Adam M Smith. 2017. WaveFunctionCollapse is constraint solving in the wild. In *Proceedings of the 12th International Conference on the Foundations of Digital Games* (Hyannis, Massachusetts) *(FDG '17, Article 68)*. Association for Computing Machinery, New York, NY, USA, 1–10. https://doi.org/10.1145/3102071.3110566

[10] Isaac Karth and Adam M. Smith. 2019. Addressing the Fundamental Tension of PCGML with Discriminative Learning. In *Proceedings of the 14th International Conference on the Foundations of Digital Games* (San Luis Obispo, California) *(FDG '19)*. ACM, New York, NY, USA, Article 89, 9 pages. https://doi.org/10.1145/3337722.3341845

[11] Jialin Liu, Sam Snodgrass, Ahmed Khalifa, Sebastian Risi, Georgios N Yannakakis, and Julian Togelius. 2020. Deep learning for procedural content generation. *Neural Computing and Applications* (2020), 1–19.

[12] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. 2021. Zero-shot text-to-image generation. *arXiv preprint arXiv:2102.12092* (2021).

[13] Adam M Smith, Eric Butler, and Zoran Popovic. 2013. Quantifying over play: Constraining undesirable solutions in puzzle design.. In *FDG*. 221–228.

[14] Adam M Smith and Michael Mateas. 2011. Answer set programming for procedural content generation: A design space approach. *IEEE Transactions on Computational Intelligence and AI in Games* 3, 3 (2011), 187–200.

[15] Gillian Smith, Jim Whitehead, and Michael Mateas. 2010. Tanagra: A Mixed-initiative Level Design Tool. In *Proceedings of the Fifth International Conference on the Foundations of Digital Games* (Monterey, California) *(FDG '10)*. ACM, New York, NY, USA, 209–216. https://doi.org/10.1145/1822348.1822376

[16] Sam Snodgrass and Santiago Ontañón. 2015. A hierarchical MdMC approach to 2d video game map generation. In *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment*, Vol. 11.

[17] Adam Summerville and Michael Mateas. 2016. Super Mario as a String: Platformer Level Generation Via LSTMs. *CoRR* abs/1603.00930 (2016). arXiv:1603.00930 http://arxiv.org/abs/1603.00930

[18] Aaron van den Oord, Oriol Vinyals, and Koray Kavukcuoglu. 2017. Neural discrete representation learning. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*. 6309–6318.

[19] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *arXiv preprint arXiv:1706.03762* (2017).

[20] Vanessa Volz, Jacob Schrum, Jialin Liu, Simon M. Lucas, Adam M. Smith, and Sebastian Risi. 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 2018)* (Kyoto, Japan). ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3205455.3205517