Innovative Pixel Art: Generation and Augmentation with Pixray

1st Marina Adriana Mercioni
Applied Electronics Department
Politehnica University Timisoara
Timisoara, Romania
marina.mercioni@upt.ro

2nd Robert Bogdan Cazacu
Department of Computer Science
Politehnica University Timisoara
Timisoara, Romania
robert.cazacu@student.upt.ro

3rd Stefan Holban
Department of Computer Science
Politehnica University Timisoara
Timisoara, Romania
stefan.holban@cs.upt.ro

2024 International Symposium on Electronics and Telecommunications (ISETC) | 979-8-3503-9086-5/24/$31.00 ©2024 IEEE | DOI: 10.1109/ISETC63109.2024.10797335

***Abstract*—The recent surge in the popularity of indie games has highlighted a growing trend towards pixel art as a favoured approach. Simultaneously, advancements in artificial intelligence have enabled the development of tools that can generate, modify, and enhance images, opening new possibilities for game art creation. This paper explores the intersection of these trends by examining the use of state-of-the-art AI tools to generate and augment pixel art. The objective is to identify a set of practical, accessible AI tools that can assist game developers in the creative process, from concept art to in-game assets. By evaluating the effectiveness, ease of use, and quality of outputs produced by various AI models, this study aims to provide actionable insights into how these technologies can streamline the art development pipeline in indie game creation. The findings aim to empower developers, regardless of their artistic skill level, to leverage AI-driven methods for enhancing their game's visual appeal and creative process.**

Keywords— artificial intelligence; CLIP; generating art; pixel art; pixray; video games; VQGAN.

I. Introduction

Indie games [1] are typically developed by individuals or small teams, often characterized by a limited scope and lower resource requirements in terms of time and budget. In recent years, the indie game industry has experienced significant growth, fuelled by access to various grants, free game development engines like Unreal Engine [2] and Godot [3], and online distribution platforms such as Steam [4] and Itch [5]. Additionally, the rise of new platforms, including mobile and Virtual Reality (VR) [6], has further expanded the opportunities for indie game developers.

Indie games often take a more experimental approach, exploring new genres, innovative mechanics, and introducing fresh ideas to the industry. In contrast, AAA games—high-budget, high-profile titles—tend to play it safe, focusing on established features and proven gameplay formulas.

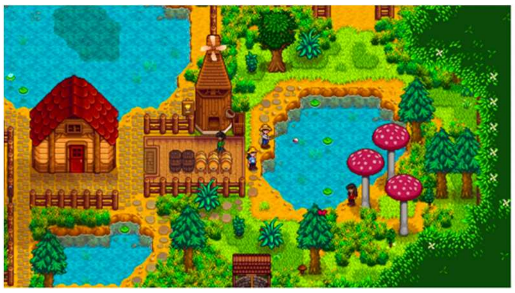

Fig. 1. Stardew Valley in-game screenshot (Source: https://www.destructoid.com/update-stardew-valley-multiplayer-comes- to-switch-this-week/)

This paper focusses on 2D pixel art indie game development, which is requiring extensive effort and numerous assets. Crafting intricate environments, characters, and animations in pixel art is time-consuming and artistically demanding, presenting a significant challenge for developers and small indie teams.

Variations of these assets are crucial for creating more realistic and immersive worlds. Notable global successes in this genre include games like Stardew Valley (Figure 1) [7] and Terraria (Figure 2) [8].

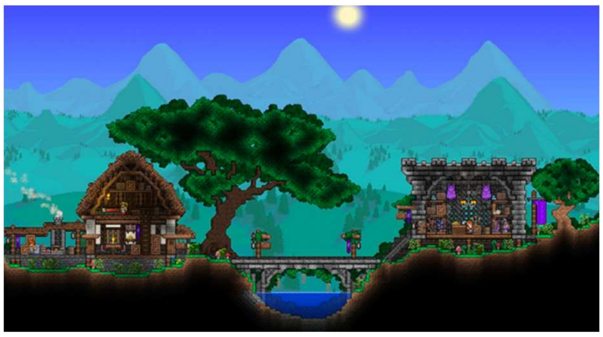

Fig. 2. Terraria in-game screenshot (Source: https://www.nintendolife.com/games/nintendo-switch/terraria)

In game development, the components used to build a video game are commonly known as "*assets*." These assets can include various file types, such as graphical elements, sound files, code snippets, and more.

In video game development, it is crucial that all assets work together harmoniously to create the desired atmosphere and effect. Consequently, a key focus of this paper is ensuring that the use of artificial intelligence (AI) allows for control over the generation and modification of various assets to maintain the intended artistic direction.

Pixel art [9] is a digital art style where each pixel is deliberately and individually placed. This art form originated with low-resolution graphics on early 8-bit and 16-bit computers, driven by hardware limitations of the time. Despite the advanced capabilities of modern hardware, pixel art remains a popular style, particularly within the indie games industry.

The primary aim of this work is to identify a set of tools and frameworks that can be practically used in real-life scenarios to generate or enhance video game assets at a reasonable cost in terms of time, resources, and required expertise.

A few general limitations on the tools and frameworks have been established to guarantee the achievement of the aforementioned objective. This is to make it simple to exclude those who don't meet the requirements. The terms "*tools*" and "*frameworks*" are used interchangeably in the lines that follow.

1. The tools need to be available to the public and have a friendly licensing strategy.

2. The cost of using the tools must be low (especially lower than hiring a professional to do the same task).

This paper is organised as follows: Section 2 presents the related work. Section 3 introduces the contribution. Section 4 discusses the results. Finally, Section 5 concludes the paper.

II. Related Work

This chapter provides a brief introduction to the tools that will be used throughout this work. There are various tools available for generating art in the form of images. However, the tool that best suits our chosen task is Pixray.

Pixray [10] is a system designed for image generation tasks. It provides an almost ready-to-use setup with a suite of tools that assist in generating images from text inputs. It utilizes perception engines [11] capable of creating visuals that fit within specific categories. Additionally, Pixray employs a combination of Contrastive Language–Image Pre-training (CLIP) [12] and Vector-Quantized Generative Adversarial Network (VQGAN) [13] to produce results similar to those achieved by DALL-E [14].

DALL-E, developed by OpenAI in 2021, is a neural network capable of generating images from text. It operates using a 12-billion parameter version of GPT-3 [15], trained on a dataset that includes pairs of text and images. Its ability to create art is truly remarkable. While it may not be suitable for tasks that demand a high level of realism, it aligns perfectly with the objectives of this paper. Concept art and pixel art, which often prioritize creativity and embrace a certain level of chaos over realism, are ideal applications for DALL-E's capabilities.

Unfortunately, DALL-E was not released in its entirety; instead, only the underlying model, CLIP, was made available. CLIP is a neural network designed to classify images, functioning as a broad-spectrum classifier that even surpasses specialized classifiers like ImageNet ResNet-101 on various benchmarks. These impressive results were achieved not by optimizing CLIP for specific benchmarks but by training it on a wide range of image classification tasks. Additionally, CLIP matches the performance of ResNet50 [16] on ImageNet zero-shot classification. Simply put, CLIP evaluates how well a text (caption) corresponds to an image and vice versa.

VQGAN is an innovative neural network architecture capable of generating images. It achieves this by building on the concepts of the Vector Quantised-Variational AutoEncoder (VQ-VAE) [17].

VQ-VAE is a variational autoencoder that generates a discrete latent space. It achieves this by constructing a codebook (a list of vectors) that essentially outlines the latent space. The output of the encoder is mapped to the nearest vector within this codebook. The codebook itself is learned through traditional backpropagation by incorporating additional parameters into the loss function. VQ-VAE serves as the foundation for DALL-E.

In VQGAN, a convolutional neural network (CNN) [18] is employed in a manner similar to a traditional VAE. However, the authors introduced the idea of using a transformer on the codebook to also capture long-range interactions. This approach enables VQGAN to learn not only the visual components of an image but also the relationships between those components.

Once both CLIP and VQGAN became publicly available, an intriguing outcome was observed when they were combined (Figure 3), producing results similar to those of DALL-E. This process operates like a typical GAN: VQGAN generates images, while CLIP evaluates how well these images match the given caption. This creates a feedback loop that continues until CLIP's confidence level is sufficiently high to be considered acceptable.

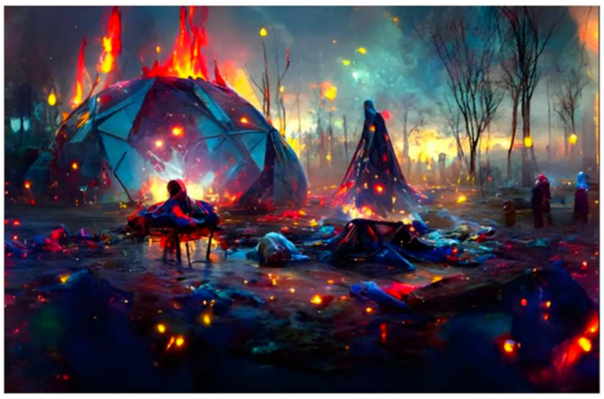

Fig. 3. Multiple text captions. "*campfire filled with stars*". (Source: https://creator.nightcafe.studio/creation/4n3AagpqyeSE8IWxA0tT)

Pixray is a Python library that integrates all the previously mentioned technologies. It combines CLIP and VQGAN, along with various other perception engines, to generate different styles of art from text prompts.

Pixray employs a sophisticated licensing mechanism. For commercial use, royalties must be paid to the library's creator; otherwise, it is free for open-source and educational purposes. Negotiable and undisclosed are the terms of the royalties. Pixray is still the greatest tool for our use-case, despite this.

III. Contribution

A. Experimenting with Pixray

An open-source Python library is called Pixray [10]. This indicates that it functions properly on local computers and is also compatible with Google Colaboratory. Given that the aim of this study is to reduce expenses associated with the video game development process, we have opted to conduct direct experiments on Google Colaboratory by utilising their most affordable membership (Google Colab Pro). This gives us access to a respectable GPU-equipped workstation that we can utilise for parallel processing, freeing up the local development system for other uses. Moreover, Google Colab functions as an online Jupyter notebook [19], and it is rather user-friendly.

The main parameter for generating images is a text caption (prompt) which describes the image to be generated and the drawer to be used (e.g., VQGAN, pixel, fft). Apart from this, there are several fine-tuning parameters, the most of which are self-explanatory.

Utilising the tool for pixel art is somewhat difficult because there is very little information in the documentation other than how to get started, and most of the examples aren't fully functional. This has led us to identify two distinct Pixray ways of handling pixel art.

After putting several drawers to the test, the "*fft*" drawer—which appears to function best with pixel art—stands out the most. While learning takes longer and iterates more quickly, the outcomes are more in line with our objectives.

Working with pixel art requires using a vast number of settings to obtain satisfying results. Finding those settings is often difficult due to the long feedback cycle and the lack of documentation. However, there is a wonderful discord community [20] that can share some helping tips and tricks (special thanks to Mike "Kkringg" Yates).

B. Generating pixel art

Generating pixel art is a difficult task. The primary problem is the excessive amount of noise that is produced around the target object and is not easily eliminated by adjusting the parameters. Owing to this limitation, we have developed a three-step process for creating art assets using Pixray:

1. Create a simple initial image to enforce the layout.

2. Generate a pixel image using Pixray.

3. Post-process the picture to isolate the asset and remove the noise.

The parameters listed in Table 1 have been applied to the image in Figure 4.

TABLE I. WORKFLOW DIAGRAM - SHARK IMAGE - PIXRAY OPTIONS

Option	Value
Text prompt	fft
Quality	supreme

To sum up, creating art from scratch is really challenging. Although Pixray may be used to start with a picture and try to simply pixelate it, there are more advanced tools available for these kinds of projects. It may be used to create pixel art, but as the workflow necessitates a lot of management and post-processing, the cost is comparable to hiring an artist to create the art assets directly.

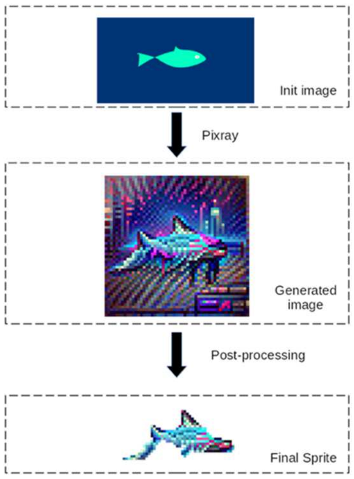

Fig. 4. Generating pixelart workflow diagram

C. Augmenting pixel art

Even though augmenting pixel art sprites requires a more complex set of options to pass to Pixray, we have managed to obtain good, consistent results when it comes to adding variations to sprites.

Asset variations are an important part of the game development process, as often, a world is made of the same building blocks but with small variations. This lowers the cost of building every asset by itself, improves the way those assets fit together, and allows for natural-looking environments.

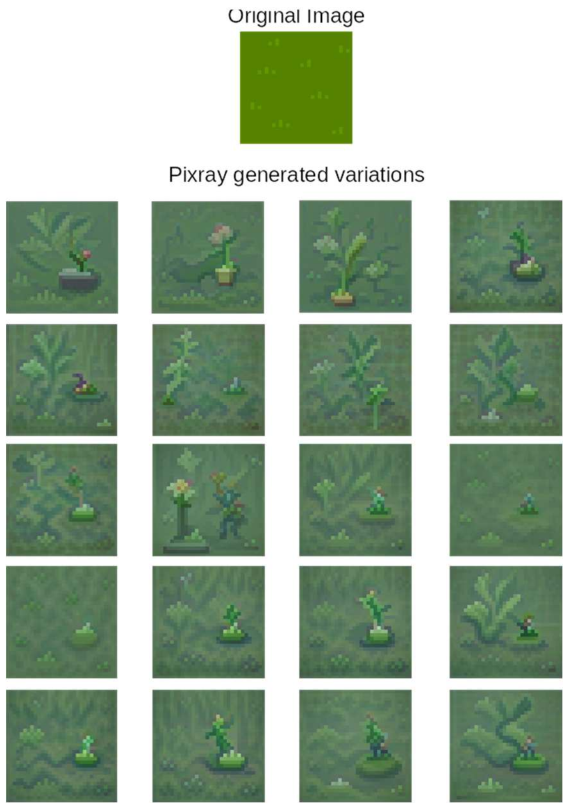

Fig. 5. Grass sprite augmentation using Pixray

It has emerged that the following set of parameters produced the greatest results when adding variants to sprites [Table 2]:

TABLE II. WORKFLOW DIAGRAM - SHARK IMAGE - PIXRAY OPTIONS

Option	Value
Drawer	fft
Quality	supreme
init_noise	none
batches	1
num_cuts	2
learning_rate_drops	101
learning_rate	0.01
Iterations	1000

When augmenting sprites, we have only been able to work with one sprite at a time since the tool doesn't handle sprite sheets effectively. Figures 5 and 6 show examples of augmented sprites (grass and dirt) created using the original image as the *init_image* and applying the settings outlined in the table above.

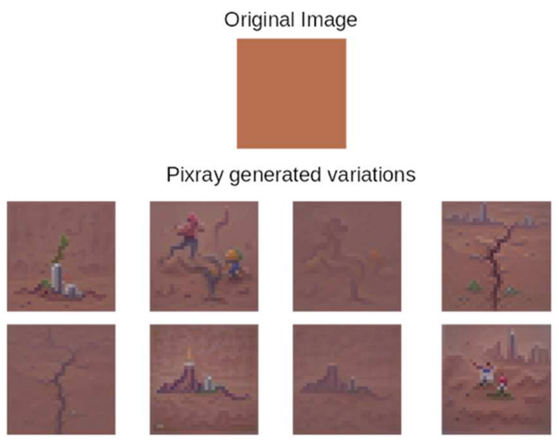

Fig. 6. Dirt sprite augmentation using Pixray

AI-based tools like Pixray excel at rapidly generating pixel art, enabling non-experts to create quickly with minimal effort. However, the results may lack the refined style of a skilled artist. Classical pixel art, on the other hand, is ideal for precision, control, and unique stylistic expression. A hybrid approach could offer the best of both worlds: using AI for fast idea generation, followed by artist refinement for a personal, polished touch.

IV. CONCLUSIONS

Pixray introduces a revolutionary method for pixel art by efficiently generating and augmenting assets. It excels at creating a variety of sprite variations from initial images, simplifying the process, and minimizing the need for manual sprite creation. This capability can notably reduce costs and accelerate development, making it highly beneficial for artists with limited resources. Iterating on pixel art character sprite sheets is a crucial part of game development. However, the process can be labor-intensive, requiring considerable effort to produce final versions that include various poses and animation clips [21].

Pixray has limitations, especially in handling full sprite sheets, requiring each sprite to be processed individually. Despite this, the tool provides significant efficiency gains and cost savings. While the generated variations are generally detailed enough for many projects, some might still need additional post-processing to improve the final results. In [22][23][24], the authors explore the use of deep learning for asset generation as a means to reduce the costs associated with the asset creation pipeline, addressing a significant concern for game development teams. Current state-of-the-art technologies still have space for improvement in both quality and ease of use. While they may be suitable for video game jams [25] or short-term projects, they might not yet be ideal for professional indie game development due to the potential quality trade-offs and the substantial trial-and-error required with these solutions. The technology effectively adds variations to existing assets, which can significantly reduce an artist's workload by allowing them to create a few default sprites and use Pixray to generate variations.

However, it requires some micromanagement as it doesn't handle full sprite sheets well. While a post-processing step can be beneficial, it is often not needed. This approach can also reduce costs, especially in pixel art where pricing is based on pixel count. Overall, using this technology can lower development costs and speed up the game's time to market.

The paper introduced two distinct workflows for using Pixray to generate or augment pixel art. Although these workflows are functional, they would benefit from further refinement. Nevertheless, they represent a significant advancement in game asset generation.

REFERENCES

[1] Fred Dutton, "What is Indie?" Accessed: Aug. 26, 2024. [Online]. Available: https://www.eurogamer.net/what-is-indie
[2] Unreal Engine, "Unreal Engine | The most powerful real-time 3D creation tool." Accessed: Aug. 26, 2024. [Online]. Available: https://www.unrealengine.com/en-US
[3] G. Engine, "Godot Engine - Free and open source 2D and 3D game engine." Accessed: Aug. 26, 2024. [Online]. Available: https://godotengine.org/
[4] Steam Store, Accessed: Aug. 26, 2024. [Online]. Available: https://store.steampowered.com/
[5] Itch, "Download the latest indie games." Accessed: Aug. 26, 2024. [Online]. Available: https://itch.io/
[6] J. Bailenson, "Experience on Demand: What Virtual Reality Is, How It Works, and What It Can Do.", 2018.
[7] Stardew Valley, Accessed: Aug. 26, 2024. [Online]. Available: https://www.stardewvalley.net/
[8] Terraria, Accessed: Aug. 26, 2024. [Online]. Available: https://www.terraria.org/
[9] J. Dawe and M. Humphries, "Make Your Own Pixel Art: Create Graphics for Games, Animations, and More!," *Illustrated edition. San Francisco: No Starch Press*, 2019.
[10] Pixray, Accessed: Aug. 26, 2024. [Online]. Available: https://github.com/pixray/pixray
[11] Dribnet, "Perception Engines." Accessed: Aug. 26, 2024. [Online]. Available: https://github.com/dribnet/perceptionengines
[12] OpenAI, "CLIP: Connecting Text and Images." Accessed: Aug. 26, 2024. [Online]. Available: https://openai.com/blog/clip/
[13] "Taming Transformers for High-Resolution Image Synthesis." Accessed: Aug. 26, 2024. [Online]. Available: https://compvis.github.io/taming- transformers/
[14] OpenAI, "DALL·E: Creating Images from Text." Accessed: Aug. 26, 2024. [Online]. Available: https://openai.com/blog/dall-e/
[15] T. B. Brown *et al.*, "Language Models are Few-Shot Learners," 2020, *arXiv*. doi: 10.48550/ARXIV.2005.14165.
[16] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," 2015, *arXiv*. doi: 10.48550/ARXIV.1512.03385.
[17] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, "Neural Discrete Representation Learning," 2017, *arXiv*. doi: 10.48550/ARXIV.1711.00937.
[18] K. O'Shea and R. Nash, "An Introduction to Convolutional Neural Networks," 2015, *arXiv*. doi: 10.48550/ARXIV.1511.08458.
[19] "Project Jupyter." Accessed: Aug. 26, 2024. [Online]. Available: https://jupyter.org
[20] Discord, "Join the pixray Discord Server!" Accessed: Aug. 26, 2024. [Online]. Available: https://discord.com/invite/x2g9TWrNKe
[21] F. Coutinho and L. Chaimowicz, "Generating Pixel Art Character Sprites using GANs," 2022, *arXiv*. doi: 10.48550/ARXIV.2208.06413.
[22] Ygor Rebouc¸as Serpa and Maria Andreia Formico Rodrigues, "Towards Machine-Learning Assisted Asset Generation for Games: A Study on Pixel Art Sprite Sheets," in *SBC – Proceedings of SBGames 2019*, ISSN: 2179-2259, 2019.
[23] M. Brocchini *et al.*, "MONstEr: A Deep Learning-Based System for the Automatic Generation of Gaming Assets," in *Image Analysis and Processing. ICIAP 2022 Workshops*, vol. 13373, Cham: Springer International Publishing, 2022, pp. 280–290. doi: 10.1007/978-3-031-13321-3_25.
[24] A. Tilson and C. M. Gelowitz, "Towards Generating Image Assets Through Deep Learning for Game Development," in *2019 IEEE Canadian Conference of Electrical and Computer Engineering (CCECE)*, Edmonton, AB, Canada: IEEE, May 2019, pp. 1–4. doi: 10.1109/CCECE.2019.8861965.
[25] Global Game Jam, "What is a game jam." Accessed: Aug. 26, 2024. [Online]. Available: https://globalgamejam.org/what-game-jam.