Wiley
International Journal of Computer Games Technology
Volume 2025, Article ID 3609613, 12 pages
https://doi.org/10.1155/ijcg/3609613

WILEY

Research Article

Evolving Camouflages: A User-Centric AI Approach for Game Aesthetics

Rasmus Ploug,[1] **Emil Rimer**,[1] **Anthon Kristian Skov Petersen**,[1] **Marco Scirea**,[1] **and Joseph Alexander Brown**[2]

[1]*The Maersk Mc-Kinney Moller Institute, University of Southern Denmark, Odense, Denmark*
[2]*Computing Science Department, Thompson Rivers University, Kamloops, British Columbia, Canada*

Correspondence should be addressed to Joseph Alexander Brown; josbrown@tru.ca

Received 27 January 2025; Accepted 24 April 2025

Academic Editor: Davide Gadia

Artificial intelligence (AI) can create icons, skins, and camouflages for games. An optimal implementation of such a concept might provide new and more advanced features that benefit the user experience. This project investigates the use of an evolutionary algorithm for texture generation and allows users to choose and manipulate camouflage patterns. This enables users to create camouflage patterns that could theoretically be implemented in a video game. This study is supported by user testing to gather insight into usability and the users' ability to replicate a target pattern. The result is an evaluation of gathered data showing user tendencies and how they engage with the system. These tendencies include significantly different completion times for target patterns varying in complexity. Additionally, participants mostly agreed that the tool is helpful for future games and objects other than camouflage skins. The findings suggest potential applications for AI in enhancing user customization and design flexibility. Further research is needed to address technical limitations and explore broader game industry implications. A brief introduction to the system described in this paper was published as a short paper in the IEEE Conference on Game (CoG) (Ploug et al. 2024).

1. Introduction

The application of artificial intelligence (AI) has become a common practice in video game design due to its ability to create dynamic gameplay elements [1–3]. One such element is the customization of the player's in-game character appearance, which often offers only a predetermined number of options. Integrating an evolutionary algorithm enables the exploration of the space of possible camouflages. The combination of AI and player-driven camouflage generation is currently underrepresented in games. This makes it a relevant topic to explore, particularly for bridging user-centric approaches with procedural camouflage generation.

Camouflage in games provides several aesthetic and mechanical values for the player. Regarding the aesthetic elements, there are two rationales for camouflage. Firstly, camouflage is a clear marker of the player's allegiance to a faction within the game as a matter of distinction. Secondly, it allows players to customize their unit within the game space. Game designers who worked on Titanfall 2 [4] highlight the variation of camouflage as being the personal choices of mercenary units, fitting into the game's narrative while allowing for player character customization [5].

In a multiplayer setting, camouflage can mechanically provide effective concealment from opponents when the outfit is well-matched to the environment. However, this paper focuses on the aesthetics of camouflage skins rather than their practicality.

Beyond aesthetic and mechanical features, there is also the question of monetization. Many games have used player skins as a monetization feature. The ability to define a personalized look for player characters would be a desirable feature, either included in the base game or for an additional cost.

This paper explores implementing a system made in the Unity[1] engine that utilizes an evolutionary algorithm to dynamically create different unique camouflage patterns. The algorithm employs different evolutionary principles to refine each generation of camouflage designs. It further enables players to customize their camouflage through user preferences and choices, increasing their agency. The paper also outlines the methods used to create the evolutionary algorithm. Other similar AI systems that inspired this project are also mentioned. An experiment has been conducted to measure the tool's effectiveness and usability. The experiment included a target-replication task and a questionnaire.

Finally, this paper discusses various considerations based on user feedback to enhance the tool for future development. Therefore, the objective of this paper is to create a tool that enables players to dynamically make their personalized camouflage pattern for a generic game, utilizing an evolutionary algorithm.

2. Background

Creating tools to generate icons, skins, or camouflages has precedents within the game industry. An example of this was the ability to develop a personal emblem for the player card in Call of Duty: Black Ops 1 [6]. This emblem was seen by the opponent each time the player killed them. This incentivized the players to express themselves via this emblem, as they could not reliably use the chat in the action-packed game. The emblem editor worked by providing a limited amount of shapes that could be freely rotated and placed. In conjunction with a limited color palette, the combinations were numerous. Many users sought to recreate a logo from their favourite series, team, or brand. However, this quickly led to players abusing this system, creating hate symbols and other profanity. Even though Treyarch announced that such behaviour would be punished with temporary or permanent bans, it made for a game with inappropriate content for younger players. It was removed after the successor, Call of Duty: Black Ops 2 [7]. This concept has since been adopted by several other games, such as the Battlefield game series [8]. However, electronic arts changed the approach by letting the players customize their emblems online on their web pages instead of using an in-game editor. This allowed their emblems to be used across multiple Battlefield games. Another point of interest is the emergence of third-party websites. These websites helped players copy already-designed emblems for personal use. This reduced some of the self-expression, as players would copy perfected emblems made by the community. In return, this also allowed players to make clan emblems and ensured that every clan member could use the same emblem regardless of their design ability.

These tools partly inspire this project, allowing players to create something on their own to express themselves within video games. However, users are constrained to creating only camouflage-like patterns through this system. The system, therefore, helps navigate a pattern space that does not allow for clear symbols to form.

2.1. Texture Generation. Texture generation is a field that encompasses computer vision, pattern recognition, materials science, and AI with a rich history spanning decades [9, 10]. The field has seen significant advancement in recent years, especially within the application of convolutional neural networks (CNNs) [11] and generative adversarial networks (GANs) [12]. Additionally, there has been research expanding the domain to many extensions of pure 2D texture generation, such as 3D surface texture synthesis [13], rotation-invariant textures [14], hierarchical variational autoencoders for texture synthesis [15], wave function collapse [16, 17], quality-diversity approaches [18, 19], mixed-initiative systems using evolutionary systems [20, 21], control mechanisms based on aesthetic metrics or content locking [22, 23], style transfer [24], and image generation from a text description (such as Dall-E 2 [25] and StableDiffusion[2]).

2.2. Evolutionary Algorithms (EAs). Genetic algorithms (GAs) are a type of EA that was initially introduced by Holland [26]. They are general optimization algorithms that simulate processes inspired by Darwinian evolution to explore solution spaces, explicitly employing a natural selection process based on environment fitness. Usually, a GA depends on a few operators: a *fitness function* that assigns a *fitness score* to each chromosome, a (usually binary) crossover operator, and a (typically unary) mutation operator. Through an iterative process of fitness evaluation, selection, and variation, the population of data structures converges towards optima in the search space.

Generating images and textures through evolutionary means has a rich history [27–29] but has seen limited application in commercial video games. Few have explored the possibilities within this field. Gonzalez et al. created a prototype that was able to create silhouettes of monsters using a word-conditioned variational autoencoder [30]. Tools on creating pixel art for character sprites have also been experimented with using GANs [31]. Yoon et al. [32] also present one of the few evolutionary systems designed explicitly for producing textures for games.

While generative AI, particularly deep neural networks (DNNs), has become one of the most popular methods for generating textures and images, GAs can offer certain advantages. GAs can provide more control over the optimization process and can be fine-tuned with greater precision than deep learning models. This makes GAs particularly helpful as users tend to prefer controllability over expressivity [33]. This is especially the case when implementing an interactive approach where the user plays an active role in the guiding process.

Another relevant approach is Picbreeder[3] [34], an example of image generation through interactive evolution [35] and a variation of EAs where the fitness function is replaced by human evaluation. Picbreeder allows a user to create abstract pictures by evolving compositional pattern-producing networks (CPPNs) using the NeuroEvolution of Augmented Topologies (NEAT) algorithm [36]. From the user experience point of view, Picbreeder shows the user a set of generated images that the user can evaluate (through selection) to guide the next generation. When the user is satisfied with their creation, they could publish this, along with a title, on the Picbreeder website to get reviews from other users. The system described in this paper is greatly inspired by the user interface and functionality of Picbreeder, being another example of interactive evolution; however, Picbreeder was not made to be a tool for creating skins for video games but rather to create interesting abstract pictures.

Brown and Scirea wrote an article about evolving woodland camouflages for video games, focusing on the effectiveness of the camouflages in a forest setting [37]. Creating these camouflages included tweaking specific values within a shader based on the fitness from the previous generation. This would be an inspiration and a starting point for this project as camouflage is a typical pattern to see in games. Although both projects use an EA to achieve the desired result, the goal for each project is not the same. Where Brown and Scirea's objective is to create effective camouflage in an arbitrary environment, this project focuses on empowering the user to create desired patterns easily. For this project, the effectiveness of the camouflage is unimportant as the focus is purely on aesthetics.

2.3. AI-Assisted Design Tools. AI-assisted design tools are a growing multidisciplinary field that requires collaboration across various domains, including ethical, philosophical, legal, and technical areas [38]. The development of AI-assisted tools for game design has seen significant advancements, particularly in creating mixed-initiative tools such as Liapis' "Sentient Sketchbook" [39]. This tool, designed for creating top-down strategy games, emphasizes resource management and base conquest, showcasing the potential for AI to support game designers in creating complex game environments [40]. Additionally, using AI-assisted map design tools has been highlighted as a valuable approach for real-time strategy games, providing abstracted map representations and aiding the design process [41]. These tools, including the Sentient Sketchbook, Tanagra [42], and Ropossum [43], demonstrate the increasing integration of AI in game development, amplifying creative inputs and providing analysis to support human designers in the creation of game levels and maps [44]. Developing such AI-assisted tools reflects the growing interest in intelligent systems that actively participate in the game development process, offering valuable support and creative input to designers [45].

3. Methodology

The following section is aimed at explaining the implementation of a vanilla EA into the tool.

3.1. Genome Representation and Texture Generation. In our EA, the genome is implemented as a Unity ScriptableObject, serving as a blueprint for generating and manipulating the camouflage patterns. This genome encodes various properties that define the visual characteristics of these textures, which are then processed through Unity's Shader Graph to produce the final material.

3.1.1. Genome Structure. The genome comprises several key attributes grouped into categories, each influencing different aspects of the generated camouflage:

- Color settings: The backgroundColor and foregroundColor define the base color scheme of the texture. These colors provide the foundational hues, which are later integrated with noise and distortion patterns to create visually complex textures.
- Pattern settings: The noiseScale and center values dictate the spatial distribution of the texture's noise pattern. The noiseScale adjusts the level of detail in the texture, where a larger scale leads to more granularity. The center is a vector that controls the origin of the pattern, allowing for positional shifts in the generated design.
- Twirl distortion: The isTwirl flag determines whether a twirl distortion effect is applied. When enabled, the twirlStrength controls the intensity of the swirling deformation, which warps the texture around a central point, introducing nonlinear distortions.
- Radial shear: The isRadialShear flag enables or disables a radial shearing effect. When active, the radialShearStrength vector dictates the degree and direction of the shearing, stretching the texture outward from a focal point.
- Spherize effect: The isSpherize setting activates a spherical distortion, wherein the spherizeStrength determines the magnitude of the effect. This introduces a rounded deformation, simulating the texture being projected onto a spherical surface.

3.1.2. Texture Generation. These attributes control the procedural generation of textures within the Shader Graph, where gradient noise is modified by a series of distortion nodes such as twirl, spherize, and radial shear. The texture's complexity arises from the interaction between these distortions and the color settings, which are further fine-tuned by the EA in conjunction with the users to evolve towards preferred camouflage patterns.

3.2. Evolution. The process of evolving these textures into new distinct camouflages involves iteratively generating populations of candidate solutions based on user selections. These solutions undergo crossover and mutation operations to produce unique offspring over multiple generations. The user selects a variety of camouflages of their liking from the population shown on screen (see Figure 1). Each camouflage created for the next generation is, therefore, based on the variety and quantity of textures the user selects. This ensures that the next generation of camouflage aligns with the user's selections.

Additionally, the algorithm memorizes all previous user selections to influence future offspring. However, the

(a) (b)

Figure 1: The user interface of the program. (a) Forty-nine squares, each with a unique camouflage pattern, are displayed. Below, the evolve button and the adjustable settings can be accessed. (b) The model of choice can be placed. In this example, a soldier wearing a uniform is chosen model for this project. Below the model is the interface for the target-replication task. Left clicking the camouflages will select them, adding them to the user-selection. Right clicking the camouflages will apply the camouflage pattern to the soldier's uniform.

likelihood of using camouflages from earlier generations decreases with the age of the generation, that is, a first generation of camouflage will be weighted lower than the second generation. Therefore, the most recent user selections take priority over older selections.

3.3. Crossover. Each new generation of camouflage is created through a crossover operation. To create each new camouflage pattern, two parent camouflages are systematically selected from the user selection, after which the offspring is formed by randomly mixing attributes from both parents.

The process starts with determining the backgroundColor and foregroundColor of the offspring. A random choice is made between the backgroundColor and foregroundColor of the two parents. The child's color is determined by this random selection to ensure diversity in appearance.

If the pattern is not locked, the offspring inherits genome attributes from both of the parents through a random selection process, like the color selection. Attributes like twirlStrength, radialShearStrength, and spherizeStrength are passed on from one of the parents, ensuring a combination of features from both parents. Additionally, slight tuning is introduced in certain properties, such as the center values. This prevents offspring from being too similar to their parents.

The crossover algorithm is aimed at maintaining diversity in the offspring, producing unique camouflage designs with mixed attributes from both parent solutions.

3.4. Mutation. Another feature is the random chance of mutations occurring during the crossover operation. Mutation serves as a mechanism for introducing novel variations of camouflages into the population, thus increasing diversity and exploration while simultaneously preventing premature convergence.

The mutation operation depends on the two variables M_{prob} and M_{range}. Respectively, these variables are responsible for the probability of a mutation occurring and the scale at which this mutation alters the attributes of the camouflage. The user freely adjusts these variables, which can be set to a decimal value within an interval of [0, 1], with 0 meaning no mutational impact and 1 meaning maximum mutational impact. As shown in the first example of Figure 2, no mutations occur when the M_{prob} has been set to 0. In comparison, the second example in Figure 2 depicts a population in which theM_{prob}has been set to 0.5 (meaning there is a 50% chance of a mutation occurring) andM_{range}has been set to 0.5 (meaning the range of mutation impact is set to 50%), resulting in some of the offspring drastically deviating from the user selections in both color and pattern.

The tool adjusts for the mutations after the crossover operation has happened. Mutations are additional variations to the attributes, where values are adjusted by weighted factors, influenced by M_{range}.

3.5. Fitness Evaluation. The fitness evaluation is a critical component in guiding the evolutionary process towards a suitable candidate for the next generation of camouflages. Unlike a traditional fitness evaluation based on objective criteria, this project's approach incorporates subjective measurements based on the camouflages provided by the user

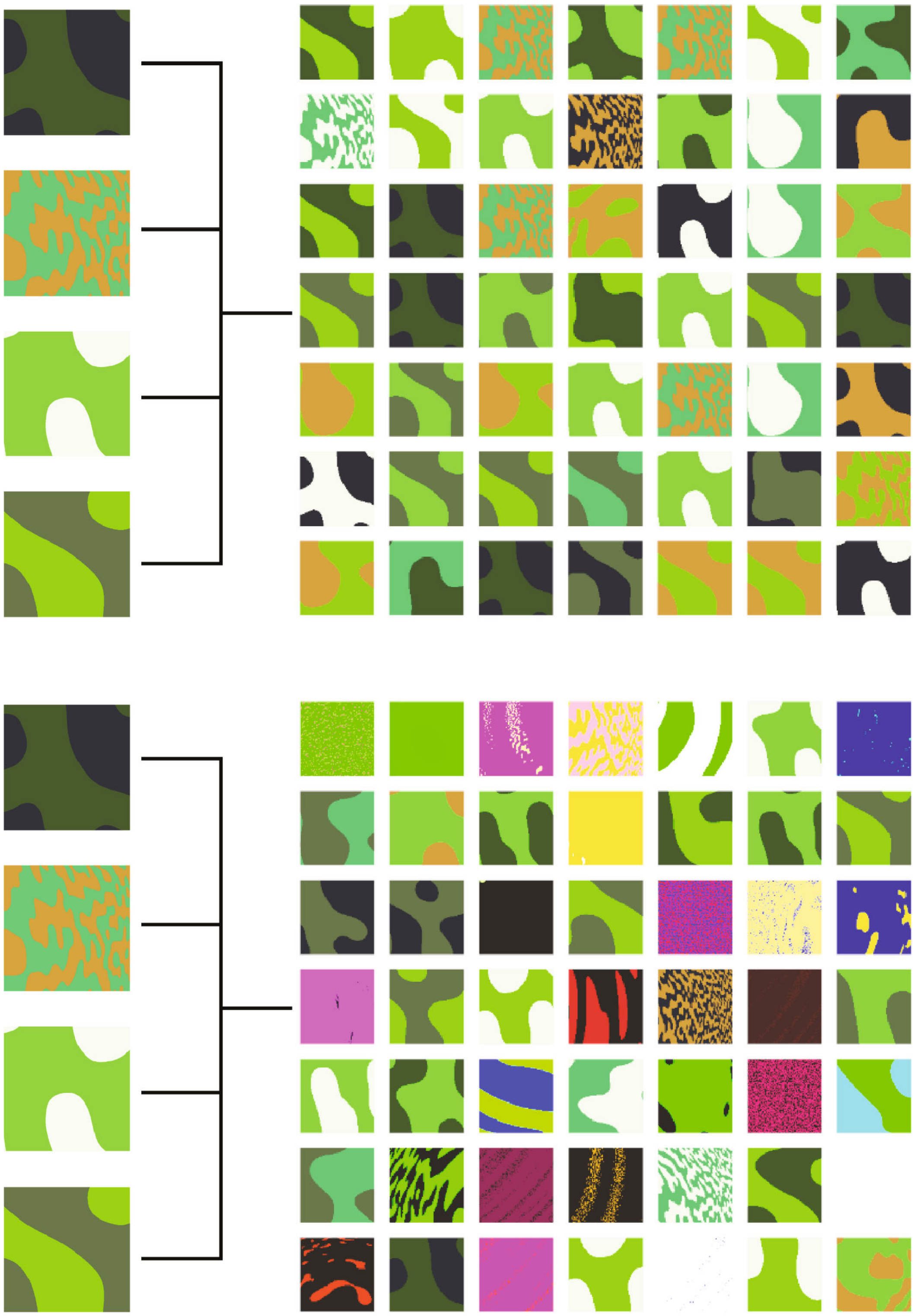

Figure 2: The selection of four camouflages with similar colors and patterns. In the first example, the offspring closely resembles the selection, as no mutations occur. Settings: $M_{\text{prob}} = 0$, $M_{\text{range}} = 0$. In contrast, the second example has both mutation probability and range set to 0.5, resulting in half of the offspring mutating. Settings: $M_{\text{prob}} = 0.5$, $M_{\text{range}} = 0.5$.

selections. This approach enables the algorithm to adapt based on personalized user preferences, resulting in generations of camouflage textures that align closely with the user's expectations.

3.5.1. Color and Pattern Locking. A feature in the fitness evaluation process is the user's ability to exert direct influence over the evolutionary direction by preserving a specific color or pattern for a subsequent generation of camouflage textures. This feature of locking a particular color and/or pattern ensures that the specific camouflage attributes become immutable, guaranteeing the retention of said attributes for the next generation (unless a mutation occurs). Color locking entails preserving a specific camouflage's backgroundColor and/or foregroundColor color for the upcoming offspring. This is shown in Figure 3, where the primary color is locked in, resulting in the following generation adhering to the specific color palette. Similarly, the pattern-locking feature enables the user to safeguard a particular pattern configuration to preserve the following camouflage textures' structural integrity and visual coherence. Figure 4 visualizes this pattern-locking feature. Finally, it should also be mentioned that these locking features can be combined, meaning that multiple attributes can be locked simultaneously.

4. Experiment

This section presents the details of the experiment conducted to investigate the dynamic creation of personalized camouflages using this tool. The experiment was designed to test the program's usability and gain insight into the participants' ability to replicate a target pattern. This section will also present an overview of the experiment design, including the test procedure. Additionally, this section provides information on the participants' demographics involved in the study and if different patterns affect the participants' replication task.

4.1. Experiment Design. The experimental design consists of a testing phase followed by a statistical analysis. This experimental design is aimed at keeping a linear approach, allowing for optimal testing and data gathering. The experiment for this project is noniterative, as the testing phase has only been conducted once. However, this approach enables the implementation of an iterative process, allowing for future testing should this be relevant. The user testing phase consists of three parts. The three parts are a free-flow task, a narrow-specific task, and a questionnaire. The implementation of these three parts is inspired by Bernhaupt ([46], 19–25). The source of inspiration is based on evaluation in the context of game design, which is relevant for this project. However, according to Bernhaupt, the overall context of this tool can be viewed as a partial game. The free-flow test is an unguided task where the participants are encouraged to engage and learn by themselves. The test for this project starts with a short introduction by the facilitators, explaining the basics of the program. Hereafter, the participant conducts the free-flow task to engage with and learn the mechanics without input from a facilitator unless a critical question or complication should occur. The period of the free-flow task is 5 min. Afterwards, the participant conducts the narrow-specific task. This task requires the participant to achieve or experience a limited or particular goal. In the context of this experiment, it will be referred to as the target-replication task. Here, the participants will be tasked with creating a replica of a random selection of three predetermined patterns varying in complexity (see Figure 5). The context of this task was to imitate what players might have done in a real scenario, by copying another player's appearance. The target-replication task is conducted directly after the free-flow task to allow the participants to utilize newly learned skills and mechanics. During this task, the participants can freely adjust the different parameters such as the mutation and locking settings to ensure a satisfactory result. The period for the target-replication task is user determined, as the participant will end this task when they feel the goal has been achieved. The result of the target-replication task will collect data on the end result of the camouflage pattern, completion time, amount of generations, and the total amount of camouflages chosen. When both phases have been conducted, the participants will fill out a questionnaire. The choice of using this approach is also inspired by Bernhaupt ([46], 69–70), which states that within game design, questionnaires are common and practical as they allow for collecting large volumes of self-report data. In addition, they also tend to gather insight into the participants' value judgment, which will help facilitate a deeper understanding of the target demographic. The overall purpose of a questionnaire within this experiment has been to gather quantitative insights. The questionnaire layout consists of a short introduction, a few introductory questions followed by usability questions and an open question on ideas for improvement. An overview of the questionnaire can be seen in Table 1. The introduction describes the purpose of the experiment as well as consensus on GDPR rules [47] and formality when participating in the survey. The purpose of the introductory questions is to gain basic information on the target audience. These are followed by questions about usability, which are based on a 7-point Likert scale [48] allowing for the participants to attach a numerical value on whether they agree or disagree with the given statements. The next part of the questionnaire contains questions regarding the tool evaluation. These questions help to evaluate the effectiveness of the tool and gather knowledge on the target-replication task. The questionnaire ends with an open question, aimed at gathering qualitative feedback on potential improvements.

4.2. Demographics. The focus of finding and recruiting participants is to gather individuals who had some prior knowledge and experience with video games. This prior knowledge will help the participants contextualize the concept properly without being hindered by missing knowledge of how the concept could be applied in games. The expected participants will therefore be recruited from academic grounds with a technical educational background. In addition, all participants will also be expected to have engaged or played video games prior to participating in this survey. Before the user testing, it was also determined that the population should be composed of 30 or more participants (10 per camouflage pattern) in order to gain a significant sample size and therefore increased support for conducting parametric tests.

5. Results

5.1. Data Analysis of the Target-Replication Task. To investigate whether different patterns had any effect on the

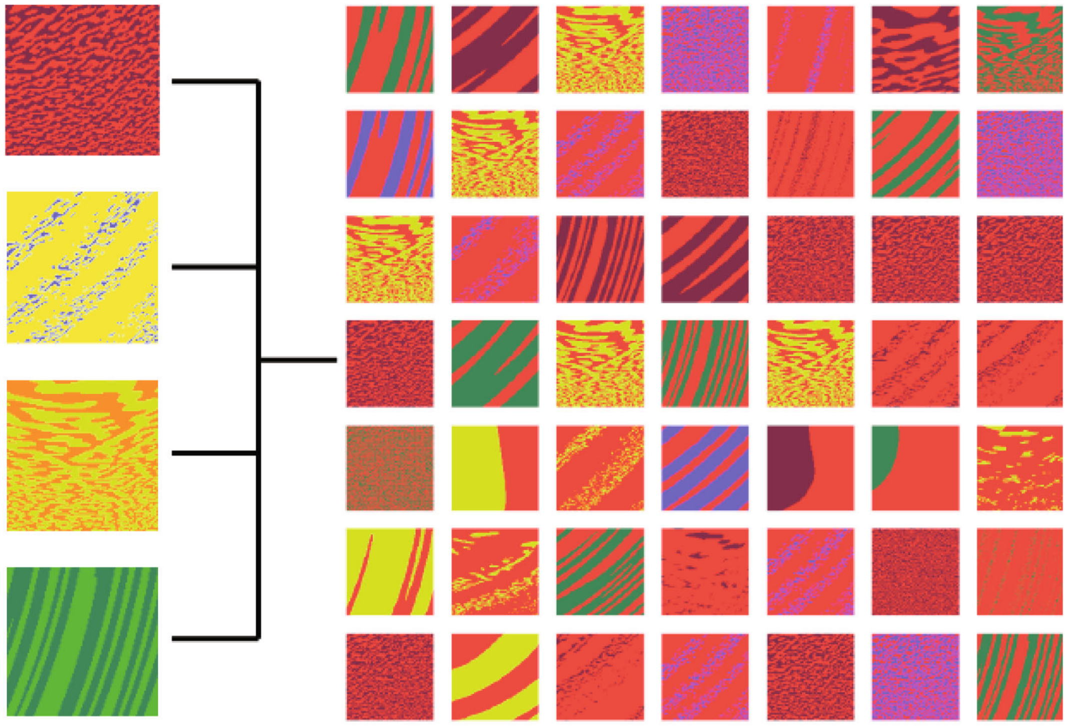

Figure 3: The selection of four different camouflages. In this example, the primary color of the top selection has been locked, ensuring that the future generation only contains camouflages of the same red primary color. This also means that all secondary colors will consist of the secondary colors of the other selections, given no mutations occur. Settings: $M_{prob} = 0$, $M_{range} = 0$.

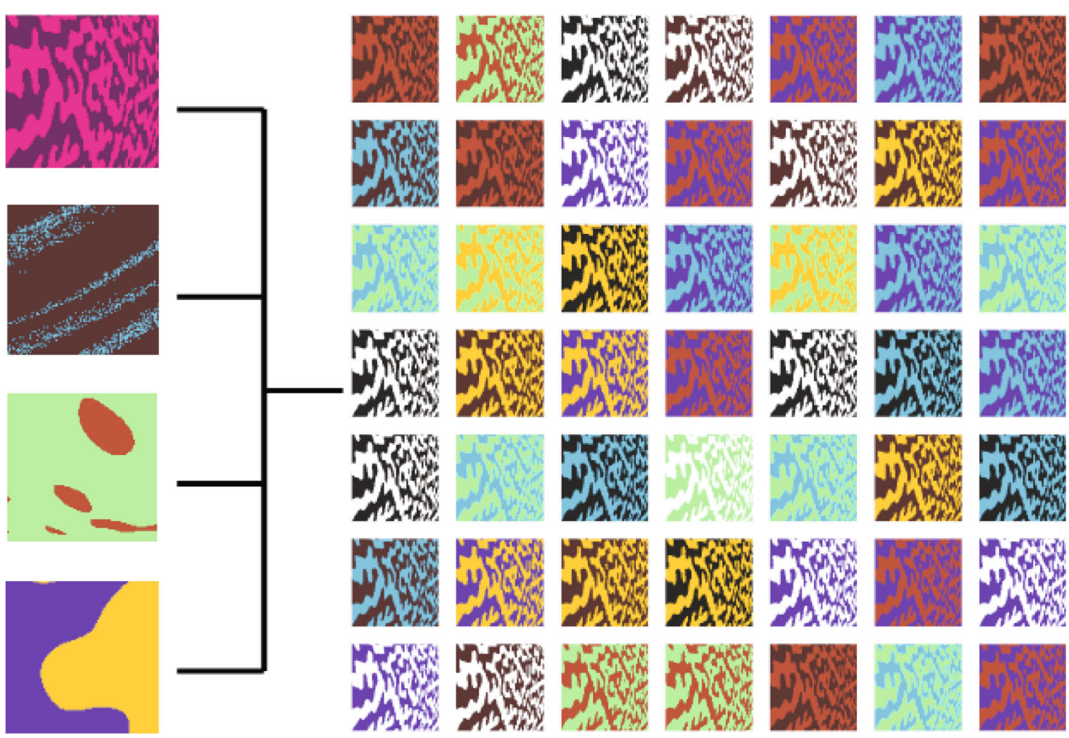

Figure 4: The selection of four completely different camouflages, not sharing a single trait. In this selection, the pattern of the top selected camouflage is locked, ensuring that all of the camouflages of the next generation will include this pattern, given no mutations. Settings: $M_{prob} = 0$, $M_{range} = 0$.

Target 1 Simple | Target 2 Moderate | Target 3 Complex

Figure 5: The experiment involved three different target designs, ranging from simple to complex. These targets were randomly assigned to participants.

TABLE 1: Experiment questionnaire.

Category 1: Personal information
1. What is your gender?
2. What is your age?
3. How often do you play video games?
4. I have knowledge within image/texture crafting[a].
Category 2: Usability
5. I think the overall user interface was intuitive to use[a].
6. I think the color locking was intuitive to use[a].
7. I think the pattern locking was intuitive to use[a].
8. I think the mutation settings were intuitive to use[a].
Category 3: Tool evaluation
9. I can see this idea being put into future games[a].
10. I would use at least one of the camouflages in a real game[a].
11. I could see this tool being used for other objects than camouflage skins[a].
12. I felt the target-replication task was successful[a].
13. I was satisfied with my result in the target-replication task[a].
14. I felt like I had control over the evolution[a].
Category 4: Other
15. Ideas for improvement?

[a]7-point Likert scale.

participants' replication task, three one-way ANOVAs were conducted to compare the mean completion time, generation, and camouflages chosen across the three targets: Target 1 (simple), Target 2 (moderate), and Target 3 (complex). Since the target patterns were assigned randomly, 9 participants received Target 1, 7 received Target 2, and 9 received Target 3. A figure of the collected data can be seen on Figure 6. The independent variables in this analysis were the three targets shown to the participants, varying in complexity. The dependent variables across the three tests were completion time, measured in seconds, generation of the result, and the total camouflages chosen throughout the task. The one-way ANOVA on completion time showed a significant difference among the different targets, $F(2,22)=3.77$, $p=0.039$, with a large effect size ($\eta^2=0.26$). Post hoc comparisons using Tukey's HSD indicated that the mean completion time for the simple target ($M=651.11, \mathrm{SD}=548.26$) was significantly higher than for the complex target ($M=216.67, SD=117.60$), $p=0.039$. No significant differences were found between the moderate target and the other groups. The ANOVAs on generation and camouflages chosen did not yield significant effects (ps > 0.05). The results from this analysis suggest that the complexity of the patterns significantly affects the participant's completion time of the replication task. Specifically, the study shows that the less complex the target pattern is, the longer it takes for the participant to complete the target replication. No findings indicate that the patterns had any significant effect on the generation of the result or the number of camouflages chosen throughout the task. A comparison between the results of Target 1 can be seen on Figures 7 and 8. compares two target-replication tasks that showcases the big visual difference in user completion time.

5.2. Survey Findings. The purpose of the survey was to assess the usability of the program, followed by questions regarding the personal satisfaction of the participants and the evaluation of the tool itself. The survey encompassed 26 participants (22 males, 3 females, and 1 person of another gender), with a mean age of 24.9 years (SD = 6.72). As introductory questions, the participants were asked about their experience with games and knowledge within texture and image crafting. Eighty-eight percent of the participants played video games at least several times a week and tended to be neutral ($M=3.8$, $\mathrm{SD}=1.1$) regarding knowledge within texture and image crafting. Most of the participants were randomly selected university students, studying game development. Questioning the participants about the intuitiveness of the program, mixed responses were received. The majority agreed ($M=4.8$, $\mathrm{SD}=1.1$) that the overall user interface of the program was intuitive; however, some parts of the program were more intuitive than others. Participants agreed that the color- ($M=4.6$, $\mathrm{SD}=1.6$) and pattern-locking ($M=5$, $SD=1.4$) mechanisms were slightly more intuitive than the mutation settings ($M=4.1$, $\mathrm{SD}=1.7$), which had many mixed answers. The participants also answered questions about the target replication task. The majority agreed ($M=4.9$, $\mathrm{SD}=1.5$) that the task was a success, agreeing upon a satisfying end result ($M=4.7$, $\mathrm{SD}=1.4$). This is also shown by the majority agreeing ($M=5.2$, $\mathrm{SD}=1.3$) that they would at least use one of their created camouflages in a real game. Participant were somewhat neutral ($M=4.4$, $\mathrm{SD}=1.6$) regarding their perceived control over the evolution. Lastly, most participants agreed that this tool could be used both in future games ($M=5.5$, $\mathrm{SD}=1.3$) and also as a tool to create patterns for objects other than camouflage skins ($M=5.9$, $\mathrm{SD}=1.1$). Many participants also had comments for improvements on the tool. Being able to pick colors yourself was a reoccurring theme. Other comments suggested more clarity on the effects of the mutation settings. Lastly, the participants were conflicted with the design of the user interface, seeking a better user experience, especially with the locking mechanisms.

6. Discussion

The project resulted in a program for creating camouflage patterns that proved successful with the method of using

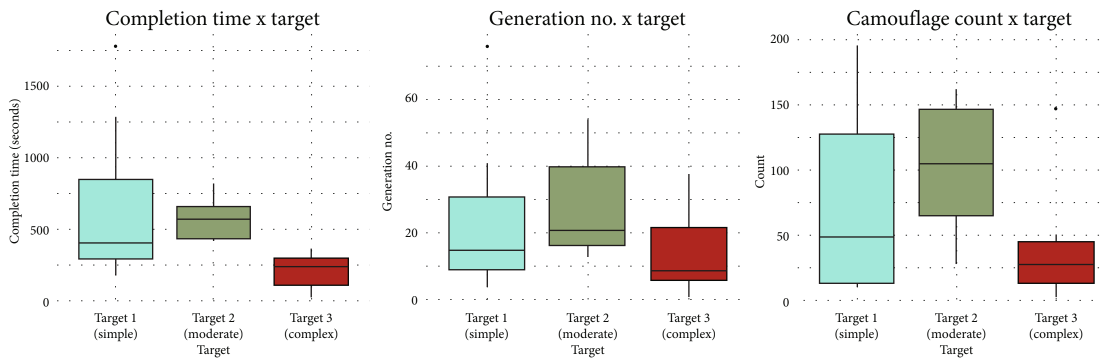

Figure 6: Box plots illustrating the completion time distribution, generation number, and count of camouflages chosen across different targets. The cyan-colored box plots represent Target 1, green represents Target 2, and red represents Target 3. Analyzing the models, it is clear that the complex red pattern stands out by having the lowest values in all three variables. The simple and moderate designs are more alike.

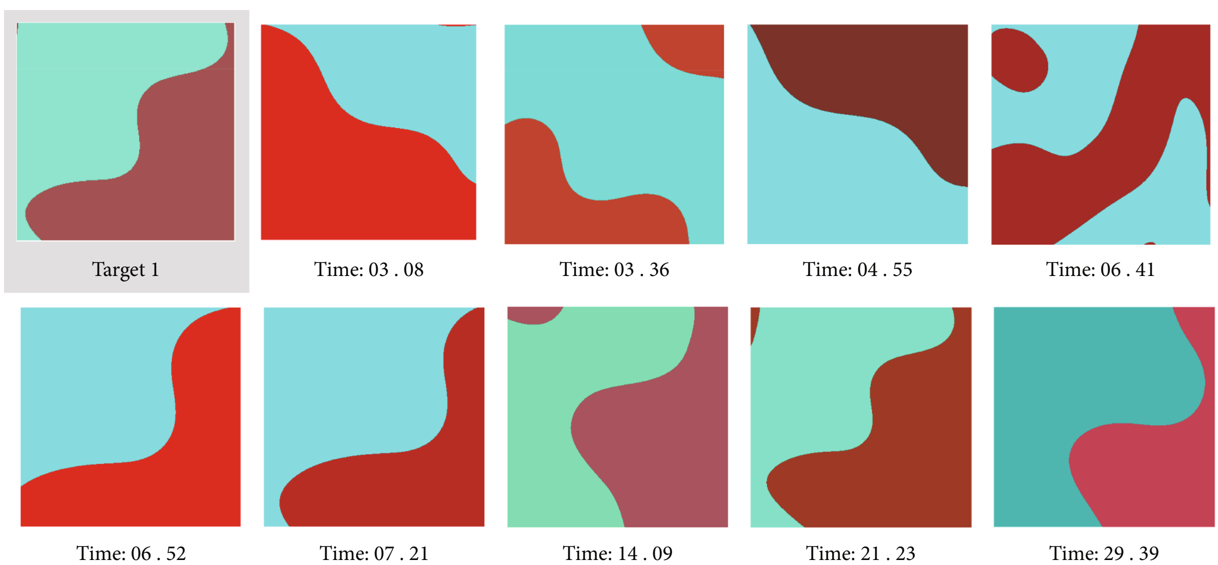

Figure 7: The results from the target-replication task that targeted the Target 1 (simple) pattern. The results are ordered after completion time. A clear difference can be seen between the results with a fast completion time and those with a longer completion time.

an EA. However, certain aspects of the results warrant discussion.

6.1. Experiment. The experiment was an overall success and yielded a relevant result. Valuable information was gathered through the target-replication task and the survey findings. It was found that the complex pattern was faster to replicate than the simple pattern. This led to the idea that the simple pattern was easier to mentally process by the participants. It was much easier for the participants to focus on a single curve or shape of the pattern with few shapes to worry about. This resulted in the participants using significantly more time on the simple pattern, trying to perfect it. In contrast, when presented with the complex pattern, participants did not feel like they could get every curve and shape in the pattern right, and opted therefore just for a similar pattern.

The survey findings suggest that the participants were satisfied with their results of the target-replication task. However, improvements are strongly advised for further testing. This includes adding a better user interface that enables the player to have more control of the color- and pattern-locking mechanisms. Suggestions were also made to add a color wheel to have full control of the color palette used in the design. This addition could make sense because of the fact that it was more complicated to achieve the correct colors compared to achieving the correct pattern. In the end, it would be a question of game design; making the users pick their own colors from a color wheel eliminates a big part of the current tool but would allow for more precision on the pattern design. In contrast, allowing participants to create their own colors through evolution adds a deeper layer of uniqueness to their patterns.

6.2. User Limitations and Understanding. From a user's perspective, one of the main limitations of the used method is the potential discord between how the program is expected

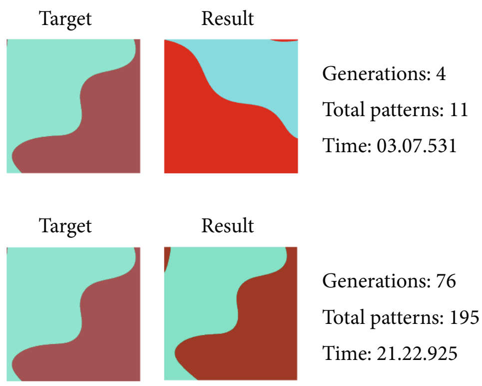

Figure 8: Two target-replication task results, both trying to replicate the simple pattern of Target 1. The top result, despite its faster completion time, shows a less accurate resemblance to the target, with fewer camouflages chosen and fewer generations of evolution. The bottom result closely resembles the target, with a slower completion time, more camouflages chosen, and more generations of evolution.

to work and how the user perceives and experiences the program. A user who wants to create a very specific skin might be limited by the program at a meta level. A reason could be that the user does not understand what the mutation settings are for or how the selection process affects the newly generated camouflages. Even with a goal from the start, a user will still have to control and understand the program to a certain level to achieve their desired pattern. The program invites people to explore the mechanics of creating textures, and having it as a feature in a game might not be a positive experience for everyone. This also correlates to the implications of an actual implementation in a video game. The program's user interface and camouflage design are very specific and might not benefit real-world games in its current state. The program would therefore have to be specifically tailored to the game where it is applied. The use of this specific program might, therefore, not prove optimal from a game developer's perspective unless it is modified. However, the proof of concept allows for future work to make the program easier to implement. This is because the current visuals do not necessarily restrict the meta-design of the tool. To make the tool fit real-world video games, other variables, such as new patterns and colors settings, could be added to the textures, allowing for a more a diverse outcome that would fit the specific games. Our survey findings also reinforce the notion that this tool could be used in other games. A game where this tool could see implementations could be The Sims [49] games. In these games, decorative textures could be applied to items like paintings, rugs, and clothing.

6.3. Future Development and Optimization. For future work, more statistical data on user engagement could benefit the project. The data would enable statistical analysis of what patterns and textures the users would like to create. A quite ambitious approach to this would be to use this data to create a neural network that can create examples of objectively good camouflages based on the user inputs.

Additionally, integrating quality diversity (QD) algorithms could expand the range of camouflage patterns by ensuring a broader variety of patterns while maintaining high quality. This approach could address limitations in current crossover methods and pattern diversity.

Another point of interest for future work would be to adjust the crossover methods used when breeding the camouflages. In the project's current iteration, the crossover introduces some limitations in color and pattern mixing. Through user testing, a more refined crossover method could be developed. In addition to this, tweaking the weights in the fitness evaluation might also prove beneficial. This could potentially result in a better evolution experience for the user.

CPPNs, known for their capacity to generate complex and varied patterns, could enhance the user's expressiveness. Incorporating CPPNs could introduce new pattern generation techniques and improve the overall design flexibility.

7. Conclusions

This paper presents an evolutionary system for dynamically generating personalized camouflage patterns. The system supports personalization and agency by actively enabling the user to participate in the selection process. This broadens the scope of available camouflage options. The successful implementation further showcases how this type of algorithm can be used for other patterns beyond camouflage. Therefore, it can be implemented in many other areas within video games.

The experiment revealed a significant difference in completion time between the simple and complex patterns during the target-replication task. Responses from the participants suggested that, due to the simple pattern being less detailed, they were more inclined to replicate the pattern to perfection, thereby using more time on the task. In contrast, the increased details of the complex pattern resulted in the participants realizing that a perfect replication would be challenging. This resulted in a lower completion time and a less accurate replication.

The survey conducted during the experiment indicated an interest in implementing the tool in the games industry. Overall, the participants expressed satisfaction with their created camouflages, further emphasizing the potential for future implementations of this tool. Additionally, the survey conveyed mixed results in regard to participant agency during the target-replication task.

Technical and design aspects could be improved to provide a more optimized and refined user experience. Additional user testing would be required to further improve the tool and better accommodate user inputs. The algorithm currently introduces some limitations on the color and pattern mixing, which should be improved upon for further development of the tool.

In conclusion, the tool successfully implemented an EA for creating personalized camouflage patterns in a generic video game setting. The personalized pattern generation system of this tool could potentially be expanded to multiple areas within the games industry, and further research within this problem domain is suggested.

Data Availability Statement

The data that support the findings of this study are available from the corresponding author upon reasonable request.

Conflicts of Interest

The authors declare no conflicts of interest.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Endnotes

[1]https://unity.com/
[2]https://stability.ai/stable-image
[3]A recreation of the original Picbreeder can be seen via this link: https://nbenko1.github.io/.

References

[1] M. Scirea, J. Togelius, P. Eklund, and S. Risi, "Metacompose: A Compositional Evolutionary Music Composer," in *Evolutionary and Biologically Inspired Music, Sound, Art and Design: 5th International Conference, EvoMUSART 2016* (Springer, 2016), 202–217, https://doi.org/10.1007/978-3-319-31008-4_14.

[2] M. Treanor, A. Zook, M. P. Eladhari, et al., *AI-Based Game Design Patterns* (Society for the Advancement of the Science of Digital Games, 2015).

[3] R. Ploug, E. Rimer, A. K. S. Petersen, M. Scirea, and J. A. Brown, "Towards Interactive Evolutionary Camouflage Design," in *Proceedings of the 2024 IEEE Conference on Games (CoG)* (IEEE, 2024), 1–4, https://doi.org/10.1109/CoG60054.2024.10645592.

[4] Respawn Entertainment, *Titanfall 2* (Electronic Arts, 2016).

[5] A. McVittie, *The Art of Titanfall 2* (Titan Books, 2016).

[6] Treyarch, *Call of Duty: Black Ops* (Activision, 2010).

[7] Treyarch, *Call of Duty: Black Ops II* (Activision, 2012).

[8] EA DICE, *Battlefield [Video Game Series]* (Electronic Arts, 2021).

[9] A. A. Efros and W. T. Freeman, "Image Quilting for Texture Synthesis and Transfer," in *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (Association for Computing Machinery (ACM), 2001), https://doi.org/10.1145/383259.383296.

[10] A. A. Efros and T. K. Leung, "Texture Synthesis by Non-Parametric Sampling," in *Proceedings of the Seventh IEEE International Conference on Computer Vision* 2, IEEE, 1999), 1033–1038, https://doi.org/10.1109/ICCV.1999.790383.

[11] L. Gatys, A. S. Ecker, and M. Bethge, "Texture Synthesis Using Convolutional Neural Networks," *Advances in Neural Information Processing Systems* 28 (2015).

[12] A. Radford, L. Metz, and S. Chintala, "Unsupervised Representation Learning With Deep Convolutional Generative Adversarial Networks" 2015, https://arxiv.org/abs/1511.06434.

[13] J. Dong, L. Li, and S. Ma, "3D Surface Texture Synthesis Using Improved Graph Cuts," in *Proceedings of the MIPPR 2007: Pattern Recognition and Computer Vision* 6788, SPIE, 2007), 671–677, https://doi.org/10.1117/12.752735.

[14] Z. Guo, L. Zhang, D. Zhang, and S. Zhang, "Rotation Invariant Texture Classification Using Adaptive LBP With Directional Statistical Features," in *Proceedings of the 2010 IEEE International Conference on Image Processing* (IEEE, 2010), https://doi.org/10.1109/icip.2010.5652209.

[15] M. Jakab, L. Hudec, and W. Benesova, "Partial Disentanglement of Hierarchical Variational Auto-Encoder for Texture Synthesis," *IET Computer Vision* 14, no. 8 (2020): 564–574, https://doi.org/10.1049/iet-cvi.2019.0416.

[16] I. Karth and A. M. Smith, "Wavefunctioncollapse: Content Generation via Constraint Solving and Machine Learning," *IEEE Transactions on Games* 14, no. 3 (2021): 364–376, https://doi.org/10.1109/TG.2021.3076368.

[17] T. Langendam, *Designer Empowerment Through Mixed-Initiative Wave Function Collapse* (Association for Computing Machinery (ACM), 2022).

[18] D. Gravina, A. Khalifa, A. Liapis, J. Togelius, and G. N. Yannakakis, "Procedural Content Generation Through Quality Diversity," in *Proceedings of the 2019 IEEE Conference on Games (CoG)* (IEEE, 2019), 1–8, https://doi.org/10.1109/CIG.2019.8848053.

[19] J. K. Pugh, L. B. Soros, and K. O. Stanley, "Quality Diversity: A New Frontier for Evolutionary Computation," *Frontiers in Robotics and AI* 3 (2016): 202845, https://doi.org/10.3389/frobt.2016.00040.

[20] A. Alvarez, S. Dahlskog, J. Font, and J. Togelius, "Interactive Constrained Map-Elites: Analysis and Evaluation of the Expressiveness of the Feature Dimensions," *IEEE Transactions on Games* 14, no. 2 (2020): 202–211, https://doi.org/10.1109/TG.2020.3046133.

[21] M. Charity, A. Khalifa, and J. Togelius, "Baba Is Y'all: Collaborative Mixed-Initiative Level Design," in *2020 IEEE Conference on Games (CoG)* (IEEE, 2020), 542–549, https://doi.org/10.1109/CoG47356.2020.9231807.

[22] A. Alvarez, S. Dahlskog, J. Font, J. Holmberg, and S. Johansson, "Assessing Aesthetic Criteria in the Evolutionary Dungeon Designer," in *Proceedings of the 13th International Conference on the Foundations of Digital Games* (Association for Computing Machinery (ACM), 2018), 1–4.

[23] A. Liapis, G. N. Yannakakis, and J. Togelius, "Adapting Models of Visual Aesthetics for Personalized Content Creation," *IEEE Transactions on Computational Intelligence and AI in Games* 4, no. 3 (2012): 213–228, https://doi.org/10.1109/TCIAIG.2012.2192438.

[24] L. A. Gatys, A. S. Ecker, and M. Bethge, "Image Style Transfer Using Convolutional Neural Networks," in *Proceedings of the 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (IEEE, 2016), https://doi.org/10.1109/cvpr.2016.265.

[25] D. M. Reddy, S. M. Basha, M. C. Hari, and N. Penchalaiah, "Dall-e: Creating Images From Text," *UGC Care Group I Journal* 8, no. 14 (2021): 71–75.

[26] J. H. Holland, *Adaptation in Natural and Artificial Systems: An Introductory Analysis With Applications to Biology, Control, and Artificial Intelligence* (MIT Press, 1975).

[27] S. Baluja, D. Pomerleau, and T. Jochem, "Towards Automated Artificial Evolution for Computer-Generated Images,"

Connection Science 6, no. 2-3 (1994): 325–354, https://doi.org/10.1080/09540099408915729.

[28] J. Dong, J. Liu, K. Yao, et al., "Survey of Procedural Methods for Two-Dimensional Texture Generation," *Sensors* 20, no. 4 (2020): 1135, https://doi.org/10.3390/s20041135.

[29] K. Sims, "Artificial Evolution for Computer Graphics," in *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques* (Association for Computing Machinery (ACM), 1991), 319–328.

[30] A. Gonzalez, M. Guzdial, and F. Ramos, "A Tool for Generating Monster Silhouettes with a Word-Conditioned Variational Autoencoder," *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 17, no. 1 (2021): 237–239, https://doi.org/10.1609/aiide.v17i1.18915.

[31] F. Coutinho and L. Chaimowicz, "On the Challenges of Generating Pixel Art Character Sprites Using GANs," *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 18, no. 1 (2022): 87–94, https://doi.org/10.1609/aiide.v18i1.21951.

[32] D. M. Yoon and K.-J. Kim, "3D Game Model and Texture Generation Using Interactive Genetic Algorithm," in *Proceedings of the Workshop at SIGGRAPH Asia* (Association for Computing Machinery (ACM), 2012), 53–58.

[33] A. Alvarez, S. Dahlskog, J. Font, J. Holmberg, C. Nolasco, and A. Österman, "Fostering Creativity in the Mixed-Initiative Evolutionary Dungeon Designer," in *Proceedings of the 13th International Conference on the Foundations of Digital Games* (Association for Computing Machinery (ACM), 2018), 1–8.

[34] J. Secretan, N. Beato, D. B. D. Ambrosio, A. Rodriguez, A. Campbell, and K. O. Stanley, "Picbreeder: Evolving Pictures Collaboratively Online," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Association for Computing Machinery (ACM), 2008), 1759–1768, https://doi.org/10.1145/1357054.1357328.

[35] T. Kowaliw, A. Dorin, and J. McCormack, "Promoting Creative Design in Interactive Evolutionary Computation," *IEEE Transactions on Evolutionary Computation* 16, no. 4 (2012): 523–536, https://doi.org/10.1109/TEVC.2011.2166764.

[36] K. O. Stanley and R. Miikkulainen, "Evolving Neural Networks Through Augmenting Topologies," *Evolutionary Computation* 10, no. 2 (2002): 99–127, https://doi.org/10.1162/106365602320169811.

[37] J. A. Brown and M. Scirea, "Evolving Woodland Camouflage," *IEEE Transactions on Games* 15, no. 3 (2023): 411–419, https://doi.org/10.1109/TG.2022.3196284.

[38] J. Qadir, M. Q. Islam, and A. Al-Fuqaha, "Toward Accountable Human-Centered AI: Rationale and Promising Directions," *Journal of Information, Communication and Ethics in Society* 20, no. 2 (2022): 329–342, https://doi.org/10.1108/JICES-06-2021-0059.

[39] A. Liapis, G. N. Yannakakis, and J. Togelius, *Sentient Sketchbook: Computer-Assisted Game Level Authoring* (Association for Computing Machinery (ACM), 2013).

[40] G. Lai, F. F. Leymarie, and W. Latham, "On Mixed-Initiative Content Creation for Video Games," *IEEE Transactions on Games* 14, no. 4 (2022): 543–557, https://doi.org/10.1109/TG.2022.3176215.

[41] S. Lee, A. Isaksen, C. Holmgård, and J. Togelius, "Predicting Resource Locations in Game Maps Using Deep Convolutional Neural Networks," *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 12, no. 2 (2016): 46–52, https://doi.org/10.1609/aiide.v12i2.12893.

[42] G. Smith, J. Whitehead, and M. Mateas, "Tanagra: A Mixed-Initiative Level Design Tool," in *Proceedings of the Fifth International Conference on the Foundations of Digital Games* (Association for Computing Machinery (ACM), 2010), 209–216.

[43] N. Shaker, M. Shaker, and J. Togelius, "Ropossum: An Authoring Tool for Designing, Optimizing and Solving Cut the Rope Levels," in *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 9, no. 1 (2013): 215–216, https://doi.org/10.1609/aiide.v9i1.12611.

[44] M. Cook, "A Vision for Continuous Automated Game Design," *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 13, no. 2 (2017): 54–60, https://doi.org/10.1609/aiide.v13i2.12967.

[45] M. Cook, "Puck: A Slow and Personal Automated Game Designer," *Proceedings of the AAAI Conference on Artificial Intelligence and Interactive Digital Entertainment* 18, no. 1 (2022): 232–239, https://doi.org/10.1609/aiide.v18i1.21968.

[46] R. Bernhaupt, *Game User Experience Evaluation* (Springer, 2015), https://doi.org/10.1007/978-3-319-15985-0.

[47] European Parliament, and Council of the European Union, "Regulation (EU) 2016/679 of the European Parliament and of the Council. Of 27 April 2016 on the Protection of Natural Persons with Regard to the Processing of Personal Data and on the Free Movement of Such Data, and Repealing Directive 95/46/EC (General Data Protection Regulation)" 2016, OJ L 119, 4.5.2016, p. 1–88. https://data.europa.eu/eli/reg/2016/679/oj.

[48] R. Likert, "A Technique for the Measurement of Attitudes," *Archives of Psychology* 22 (1932): https://doi.org/10.4135/9781071812082.n626.

[49] Electronic Arts, *The Sims [Video Game Series]* (Electronic Arts, 2000).