Journal of Applied Artificial Intelligence
https://jaai.sabapub.com
ISSN: 2709-5908

2026 Volume 7, Issue 1: 36–52

DOI: https://doi.org/10.48185/jaai.v7i1.2012

Exploration of Human Design with Genetic Algorithms as Artistic Medium for Color Images

Aidan F. Schmelzle[1*] **and Arvin Agah**[1]

[1]Department of Electrical Engineering and Computer Science, University of Kansas, Lawrence, KS 66045 USA

Received: 28.02.2025 • Accepted: 15.05.2026 • Published: 04.06.2026 • Final Version: 30.06.2026

Abstract: Genetic Algorithms (GAs), a subclass of evolutionary algorithms, seek to apply the concept of natural selection to promote the optimization and furtherance of attributes/features designated by the user. GAs generate a population of chromosomes represented as value strings, score each chromosome with a fitness function on a defined set of criteria, and mutate future generations depending on the scores ascribed to each chromosome. In this paper, each chromosome is a bitstring representing one canvased artwork. Artworks are scored with a variety of design fundamentals and user preference. The artworks are then evolved through thousands of generations and the final art piece is computationally drawn for analysis. In cases that a user seeks to implement their own vision through careful algorithmic refinement, genetic algorithms find a place in generating color artwork. This paper shows that GAs can act as a medium for artistic expression. The key research finding is that genetic algorithms can be used to produce abstract, unique color images where the rules can be controlled through the definition of various fitness functions.

Keywords: genetic algorithms, evolutionary arts, applied AI, color images using AI.

1. Introduction

There have been numerous applications of genetic algorithms (GAs) in training machine learning models. GAs [1] can be applied to almost anything that a user can quantify and excel at finding global optima or near optima. This search can take place in large, potentially discontinuous search spaces. Though they have been found to possess sub-optimal image generation capabilities in comparison to advanced techniques, there is an argument for using to GAs as a mode to incorporate more human involvement. The intersection between art and technology has experienced much discussion in recent times. Many believe that the removal of humans from ”art” poses a dangerous question for our future. Others argue that machine-generated art opens doors and makes the generation of art accessible. The rapid progression of AI techniques requires us to take a look and consider what may have been left in the wake of this technological boom. The demand for careful construction of the GA elements reintroduces human touch to program output. This research explores the potential to quantify design elements in order to iterate generations of artworks.

* Corresponding Author: aidanfrose@gmail.com

2. Background

2.1. Genetic Algorithms

GAs draw upon the concept of natural selection to find optimal or near optimal solutions to a problem. The algorithm works by generating a population of candidate solutions (chromosomes), evaluating candidates (parents) on a defined fitness function, selecting the fittest individual chromosomes and using them to produce offspring. If a stopping criterion is met, the algorithm reaches completion, otherwise, evolution continues with the new generation of children. In this work, color artwork is generated using GAs.

Chromosome – Each candidate solution is represented by a value string. In this case, the string is a 736-bit binary string. The chromosome contains all the information needed to describe the artwork. The rationale for the chromosome structure is that it must contain information to uniquely describe the shape, size, base color, tint/shade, rotation, and position.

Gene – Genes are components of GA chromosomes. One or more bits provide information about the attributes of the chromosome. For example, in this work, the first 3 bits of every 23-bit sequence in a chromosome represent a shape. The number of bits assigned to each characteristic is calculated based on the number of options. For instance, if there are eight shapes, then three bits are allocated, as two to the power of three is eight.

Population – A population is a set of possible solutions. Initial GA populations can be either partially or fully constructed, or partially or fully set randomly. In this paper, the initial population is composed of 300+, random bit strings. The population was set at a value to seek a balance between diversifying the population, and not significantly increasing the computational complexity.

Selection – GAs utilize various selection methodologies to determine which chromosomes will produce offspring. Here, roulette wheel selection (RWS) is utilized. In this selection method, more highly fit chromosomes occupy a larger ”slice” of the roulette wheel, and thus, a larger chance of being selected to generate offsprings. RWS offers a realistic selection process as all chromosomes are still eligible to reproduce, as chromosomes receiving lower fitness scores are not eliminated from the gene pool, and they just have a lower chance of reproducing.

Crossover – Crossover constitutes the means by which chromosomes are combined to generate new offsprings. Single-point crossover is utilized in this experiment. In single-point crossover, a point between two bits is chosen at random, and the tails are swapped to create two new offsprings from the two parents. Single-point crossover is common in GA experiments, and was therefore chosen.

Mutation – Mutations are incorporated into GA evolutions the way one would expect to see in the natural world. Mutations introduce random changes to chromosomes to promote diverse populations and to prevent the algorithm from reaching premature convergence. Mutations occur at a set percentage rate, which can have a wide range. The mutation setting used the default values in GA.

2.2. Design Concepts

The question being asked by this research is not whether a computer can produce art but rather what kind of art a human can build with GAs. As the fitness function must be implemented by a human, there is demand for human design and discernment. The structure of both the chromosome and the fitness function were constructed with design concepts in mind. Thought was given to the number of elements and colors needed to generate interest, as well as how to preserve the standard color wheel. Some of the general design principles include composition, layout, color theory, focal point, hierarchy, and repetition. In order to evaluate/score artworks, design elements must be quantified

which is where the user must make decisions about how to reward or punish chromosome characteristics based on their own preference. For example, there is no absolute way to define what constitutes a perfect composition, but there are general rules of thumb. Further specifications are derived from individual preferences. The design elements selected for usage here are symmetry, color harmony, and shape diversity.

Symmetry – The design concept of symmetry is often thought of as providing a sense of balance to an art piece and aids in drawing a viewer in towards an image's focal point. Analyzing the characteristic of symmetry will involve examining the mirror-point of each pixel in the piece along one or more axes.

Color Harmony – Harmonious color palettes provide interest to pieces in a variety of ways. Monochromatic, or single-hue, pieces draw viewers in as the lack of color diversity encourages viewer to look more closely at what is hidden in the variety of values. Complementary palettes help to make opposing colors pop and can add an element of extremity due to the rich saturation. The algorithm will use chromosome attributes as well as the standard color wheel to analyze each art piece.

Shape Diversity – The range of shape diversity can add interest to a piece. Whether all shapes are the same or all types of shapes are included on the canvas, interest is built in the overlapping elements of the piece.

3. Related Work

Applications of GAs – GA [1] applications to different art forms have been explored in a number of projects. Examples of such applications include animated artwork [2][3]; three-dimensional art [4]; stylized images [5]; vector graphic images [6]; video art [7][8]; generating art and art design [9]; painting [10][11][12][13][14][15]; video generation [16]; fractal design [17]; and image enhancement, denoising, and reconstruction [18][19][20]. GAs and AI techniques have numerus other applications, with a few examples including scheduling [21]; optimization [22]; data classification [23]; disease detection [24]; social networks [25]; Student Attrition [26]; and network topologies [27].

GAs in Binary Image Evolution – Genetic algorithms have been applied to the evolution of digital images [28][29] with binary images represented by 16x16 matrices of pixels. These images are evaluated based on their fitness and reproduce to create the next generations of children. Fitness is calculated using various masks of 3x3 or 4x4 size grids which traverse the matrix looking for features such as solid squares, hollow squares, vertical lines, etc. The population size was set at 1,000 and each experiment ran for 120 generations. Final matrices were tiled to more clearly show the resulting patterns of the high-scoring images.

GAs in Art with Optimized Operators – The reproductive step of GA evolution is limited by the variety and methodology of its operators. An improved Genetic Algorithm has been proposed [30] to explore crossover and mutation optimization to promote a more diverse set of candidate solutions. Parameter-Free GAs (PfGA) automatically adapt their parameters over the duration of a run to more closely match the desired results.

GAs Incorporated with Artistic Decision-Making – When incorporating GAs in art image evolution and generation, some projects try to increase human involvement while others seek to automate the processes that mitigate human intervention. One example is based on a sitter image which the algorithm utilizes to develop derivative works based on some designated aspects of the human

creative process [31], acknowledging that the artistic choices of humans are based both on structural limitations and an understanding of form, as well as incorporation of breaking or manipulating those constructs with associative thought.

The Generative Artificial Intelligence (GAI) field is discussed in [32], which includes image generation and video production. The use of Convolutional Neural Networks (CNNs) to differentiate AI-generated art from human-created based on Error Level Analysis (ELA) is presented in [33]. The application of computer vision models to detect whether or not an artwork has been generated by a human or a computer is discussed in [34]. AI-generated paintings are studied in [35], investigating public interactions and perceptions. The copyright and ethical ownership of AI-generated art are covered in [36], and the effects of AI-generated images on education are discussed in [37]. Other recent works include the use of neural networks for paintings [38], review of AI-powered creativity [39].

4. Methodology

In order to achieve results that are complex enough to showcase the effectiveness of the GA, while avoiding excessive computational overload, 736 bits were designated to each chromosome. Each canvas would contain 32 shapes with 23 bits of attribute values divided into shape, size, color, the existence of a tint or shade and which, and rotation and position. Each art canvas is comprised of 512x512 pixels. The project is implemented in Python programming language [40], using the PyCairo module for graphics [41]. The methodology algorithm/pseudocode is included in Appendix B.

4.1. Chromosome Components

Shape – The first 3 bits of the chromosome indicate which of 8 shapes will be drawn onto the Canvas. The first shape in the chromosome will be drawn on the bottom layer and each shape in succession will be overlayed in a new layer on top of its predecessors. This is important to note as fitness function scoring rewards interesting shapes formed with this methodology. For instance, the top shape at maximum size covers all shapes. Shapes are represented as shown in Table 1 (Appendix A).

Size – There are 6 bits of the chromosome that are dedicated to shape size, as listed in Table 2 (Appendix A). Size applies to both the x and y axes, each of which receive 3 bits of information. This means that equivalent values will produce a symmetric shape, whereas differing bit values will produce an oblong or skewed shape. The maximum shape size is 512 pixels, meaning that a square of the largest size could cover the canvas entirely. A chromosome's final shape with this property would produce a blank or mono-colored canvas while the first shape with this property could provide an interesting background color for the rest of the shapes on the canvas.

Color – Overall, 5 bits are allotted to color values. The first 3 bits represent the primary (red, yellow, blue) and secondary (orange, green, purple) colors as well as black and white. Tints and shades are incorporated with 2 extra bits. This approach allows for the inclusion of a wider range of colors without filling extra bit representations randomly. The 8 base colors and their respective hex values are represented in Table 3 (Appendix A) and are shown in Figure 1.

Tint Shade (Binary Y/N) – Generating tints and shades by adding white or black, respectively, is a tool used by artists and allows the fitness function to more effectively reward chromosomes with interesting color palettes (such as complementary or monochromatic palettes). There is 1 bit allotted to give a binary yes or no, as shown in Table 4 (Appendix A). If the value here is a yes, or 1, the following bit, will indicate which value (black or white) will be mixed with the base color identified

earlier, as included in Table 5 (Appendix A). If the value is no, or 0, the base shade will not be made into a tint or shade and the subsequent bit value can be disregarded.

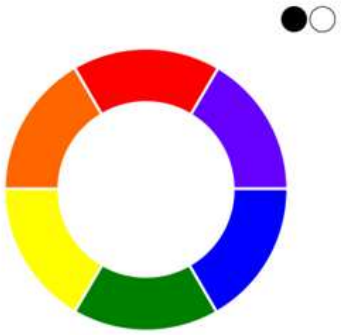

Figure 1. Color wheel representing 8 base colors.

Tint or Shade – There is 1 bit allotted to the black and white values which will be added to a base color given that the previous bit is a 1. This allows for the development of tints and shades of each base color, and their respective hex values, as represented in Table 6 (Appendix A).

It should be noted that the 8 base colors shown in Figure 1 have a statistically higher chance of being chosen due to the chromosome structure. If the 10th bit which decides if a base color will be mixed with a tint or shade is a 0, then the shape's color is limited to 1 of 8 colors, i.e., each has a 6.25% chance of being selected. If the 10th bit is a 1, then the base color will be mixed with a tint or shade (designated by the 11th bit) and will be one of 15 colors, i.e., each color on the extended palette has only a 3.125% chance of being selected. The number 15 is on the extended palette, illustrated in Figure 2, so that the grey shade can be selected two ways, either if the base color is white and a shade is added, or if the black color is the base and a tint is selected. While the other colors in the extended palette have a 3.125% chance of being chosen, gray has slightly higher odds at a 6.25% chance. Lastly, white and black are capable of being chosen in both palettes as black with the addition of a shade is still black, and white with the addition of a tint is still white. This means that black and white ultimately have a 9.375% chance of being chosen. Grey, as well as the primary and secondary colors have a 6.25% chance of being chosen, and all the other colors with either a tint or shade have a 3.125% chance of being chosen.

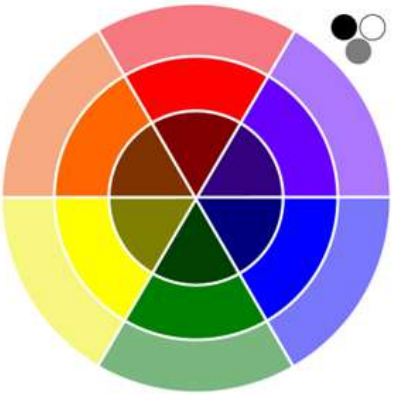

Figure 2. Color wheel representing extended palette with tints and shades.

Rotation – 3 bits are allotted to shape rotation, which works by rotating the entire user space, due to the software libraries used. This means that without recalibrating the center of each shape, the context

is rotated around the origin and shapes drawn near the canvas edges are rotated out of scope once the rotation is returned to 0 after drawing a shape. Prior to drawing each shape, its center point is calculated and utilized so that the rotation is only applied to that shape in that specific loop iteration. The center is reset to the origin after completion. Degrees of rotation are represented in Table 7 (Appendix A).

Position – There are 6 bits of the chromosome that identify the shape's position on the canvas with 3 bits identifying position on the x-axis and 3 bits identifying position on the y-axis. The canvas can be thought of as a grid, and the center of each shape can be snapped to the grid along the x and y axes, as shown in Table 8 (Appendix A).

Shapes are prevented from being lost in the canvas edges by padding the grid with 32 pixels on each side. This generates interest by allowing larger shapes to extend off the canvas but prevents smaller shapes from being lost on the edges.

Additionally, artworks attempt to pull the viewer inward towards a focal point which is not often on the canvas edges. It is of note that the surface origin is located in the top left corner as opposed to the bottom left corner. Many of the built-in shape-drawing functions utilize the positional parameters to represent the top corner of a shape rather than its center which had to be factored into the geometry calculations.

4.2. Initializing the Population

The initial population is set at random. Each chromosome is 736 bits and the population size is 1,000 chromosomes. After each generation, 2% of the parents are retained in the next generation, i.e., each generation after the initial population will contain 306 chromosomes. The fitness scores of the retained parents are recalculated in the next round of scoring.

4.3. Drawing the Artworks

Artwork construction happens primarily within PyCairo, which allows for the creation of a context surface and for ease of shape generation. It has some built-in shapes like circle and rectangle and supports the visualization of other polygons by using point-to-point calculations. Given the chromosome structure, attributes such as position and size are utilized to draw the shapes. After the canvas has been drawn, it is saved using PNG format and converted into a NumPy (Python open-source library for numerical calculations) array whose features are examined and scored by the fitness function [42]. This feature also allows the canvases to be saved periodically for examination throughout the fine-tuning process. In final runs, the most optimal chromosome is saved in PNG format.

4.4. Scoring the Art Utilizing GA Fitness Function

The fitness function is written to consider both the chromosome itself and the array representing the actual canvas. Fitness function scores on the design elements symmetry, color harmony, and shape diversity; and the weighted importance of each is toggled to examine the impact on GA results. The fitness rubrics are listed in Appendix B.

Symmetry – Symmetry is the element that is scored using an array. The function takes the array as input and calculates its symmetry across vertical, horizontal, and diagonal axes. The final symmetry score can be toggled to weight components more or less strongly; and is otherwise calculated as a mean of each score. The grey background color is not shared by the grey in the color wheel, and is disregarded when computing symmetry scores to prevent error in calculation.

Color Harmony – Interest can be generated in pieces by utilizing various color palettes. The color palette function takes a chromosome as input and calculates scores for usage of either a monochrome palette, a complementary palette, and a palette consisting of the most diversity. The weights can be adjusted across palettes to promote a specific type of harmony, but the final score is not an average, as an art piece that receives a highly-scoring monochrome palette will also have a low diversity. Averaging the scores would negate the purpose.

Shape Diversity – In order to promote complex compositions, a score identifies how homogeneous or diverse a set of shapes is by using Shannon Entropy. This score can be toggled up or down to encourage compositions to utilize more or less shape diversity.

5. Experimental Results

The results of this project can be seen through the progression from a completely random chromosome's display to a chromosome that has undergone genetic evolution. Figure 3 is a display of a randomly-generated chromosome from the initial population. There is no obvious color harmony, and any elements of symmetry or diversity are coincidental at this initial stage of the evolution.

High-symmetry weights produced somewhat weak results. This is because symmetry performs point-to-point calculations. At first, final evolutions often produced primarily solid canvases or extremely narrow shapes as either the largest shapes or the background color were unintentionally producing the most symmetry. This can be seen in Figures 4 and 5. Symmetry scoring was reprogrammed to disregard the background color when calculating symmetry. This change is reflected in the remaining figures.

The most interest was found in evolutions where the color harmony and shape diversity were heavily weighted while symmetry received minimal importance. Such instances can be seen in Figures 6 and 7. The color harmony generates interest as overlapping shapes of the same hue form complex forms. Shape diversity promotes interest by including a mixture of rounded and pointed corners.

As the images evolved over 300 generations, the program saved the high-scoring chromosome after every 30 generations until completion. This can be seen in Figures 8 to 12, where the generational number as well as the fitness score are denoted for each canvas in the figures. It is of note that scores could converge to 1, as fitness function scores are normalized in both their individual components and their final score. The early convergence is likely due to the simplicity of the fitness function. Even with a small population size, the likelihood of one of the chromosomes receiving a high score across all categories and weights is exceedingly high.

The initial population size was set to be 1,000. The time taken to evaluate one generation of this population size was approximately 68.79 seconds, i.e., it would have taken about 19.1 hours to iterate 1000 generations. Due to the fact that the high scoring chromosome of the first generation already displayed convergence led to scaling down the population size and number of generations. It is of note that to combat the high symmetry scores of much larger shapes, the scale of the shapes was scaled down for some experiments. In Figure 12, instead of shape sizes mapping to 4 to 512 pixels, they instead were mapped to 2 to 256 pixels. This arguably produced more interest in the resulting canvases.

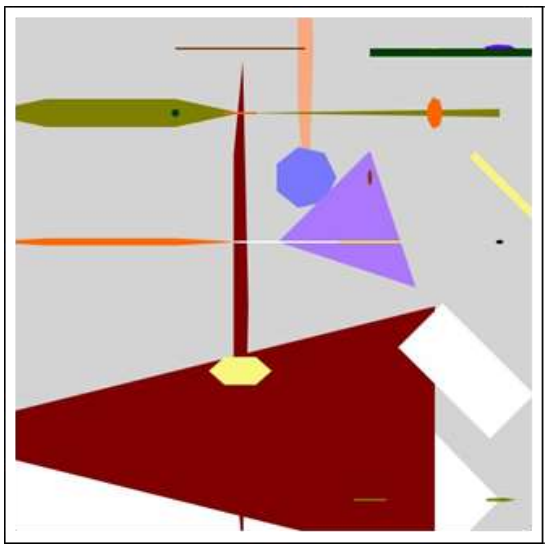

Figure 3. Randomly generated chromosome from initial population.

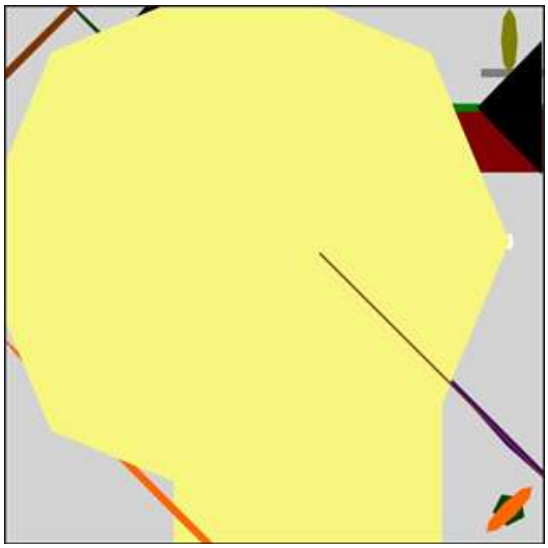

Figure 4. Symmetry weight 0.8.

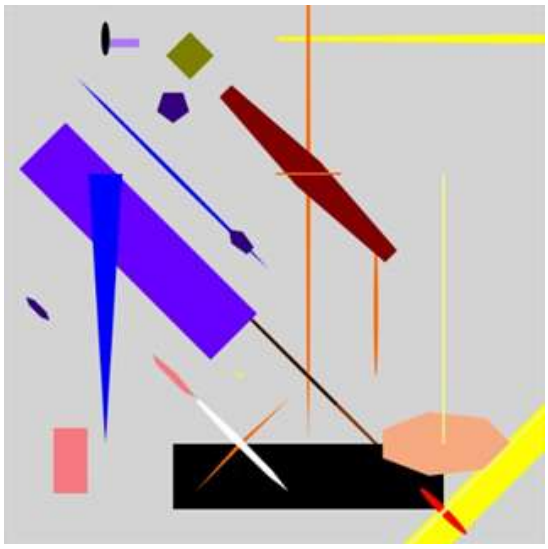

Figure 5. Symmetry weight 0.8.

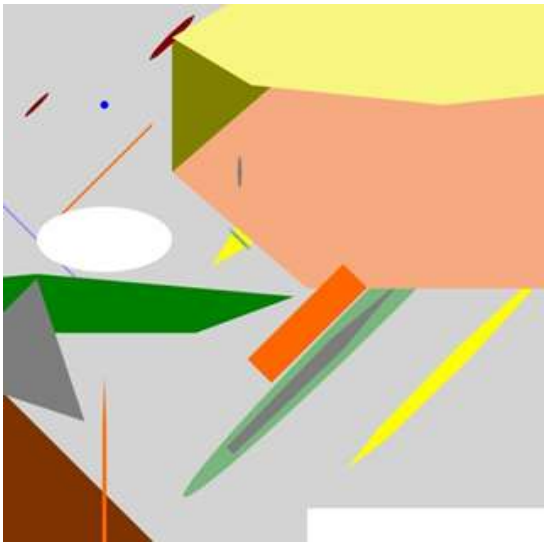

Figure 6. Color harmony weight 0.5 and shape diversity weight 0.5.

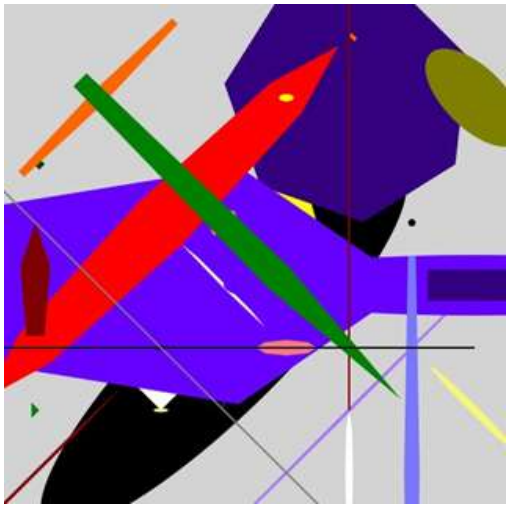

Figure 7. Color harmony weight 0.5 and shape diversity weight 0.5.

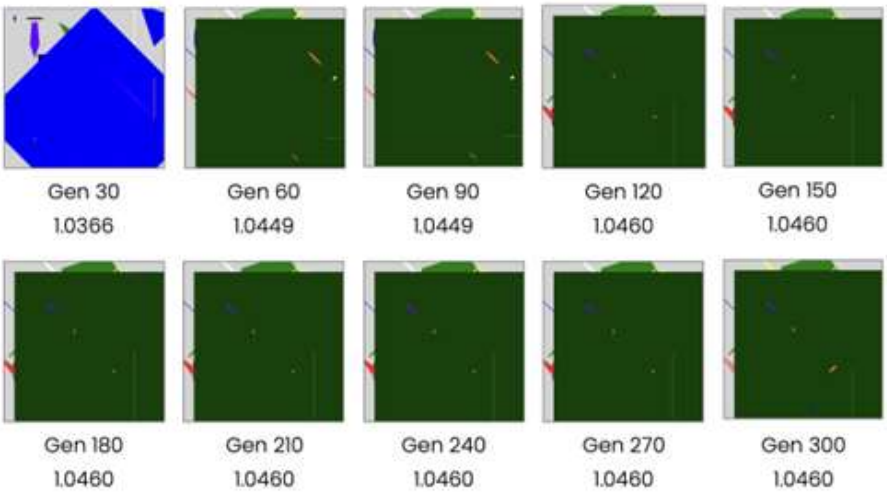

Figure 8. Symmetry (0.33) color harmony (0.34) shape diversity (0.33).

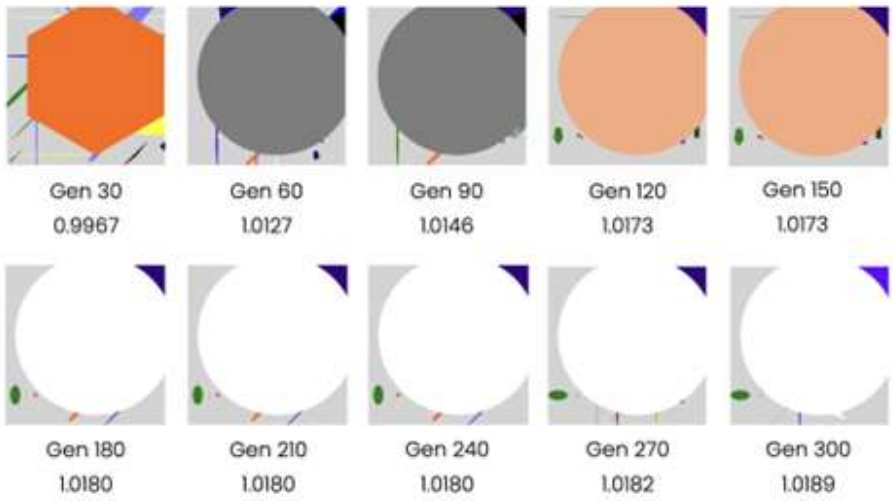

Figure 9. Symmetry (0.8) color harmony (0.1) shape diversity (0.1).

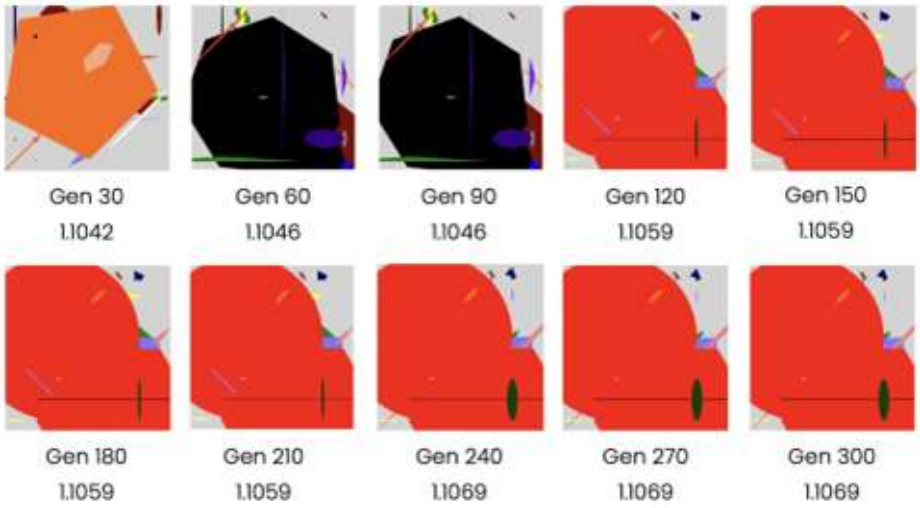

Figure 10. Symmetry (0.33) color harmony (0.34) shape diversity (0.33).

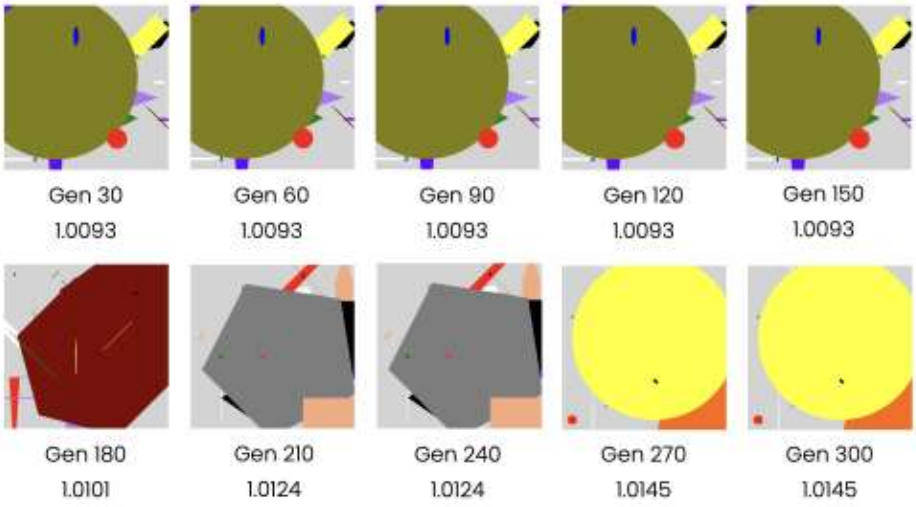

Figure 11. Symmetry (0.8) color harmony (0.1) shape diversity (0.1).

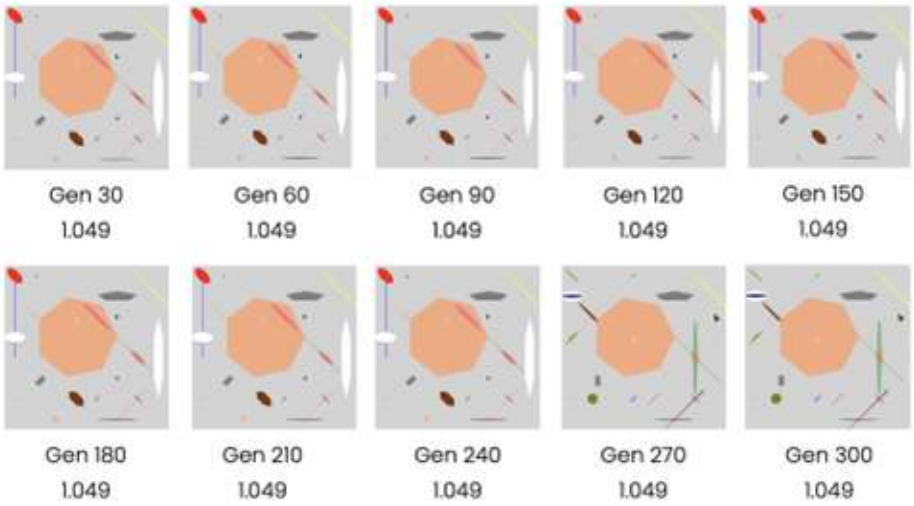

Figure 12. Symmetry (0.4) color harmony (0.4) shape diversity (0.2).

6. Conclusions

The work presented in this paper focuses on the use of GA to produce art. Another approach would be the use of generative AI approaches, such as [43]. The rationale for using GA is that the goal was to produce abstract, unique color images where the rules can be controlled (through the definition of the fitness function). The use of generative AI would be a dissimilar approach with the different goal of generating realistic, high-resolution color images based on a text prompt. There are also interesting discussions on ownership and other legal aspects of AI-generated artwork [43][44].

Additions to current work include fine-tuning the symmetry function to promote more subtle aspects of shape symmetry as opposed to rewarding canvases with extreme amounts of solid color. It is also possible to reward large shapes in the background (those which appear as one of the first shapes in a chromosome) while discouraging large shapes appearing in the foreground which lead to the coverage of any interesting elements lost underneath.

Future work consists of extending the scope of design elements considered by the GA fitness function. When more elements are considered, it becomes more difficult to discern how toggling different element weights affects the overall composition; however, too few fitness function components can lead to convergence that are not of interest. It is also possible to increase the amount of attributable information in each chromosome. Some potential implementable features include stroke, color gradients, and background fill colors. Increasing the variety of shapes, colors, positions, etc. is another way to advance the current project scope. Exploring parameter alterations is another avenue for future work. Utilizing other selection or crossover types as well as a much different mutation rate may produce different set of results.

Conflict of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

References

[1] Holland, J.H. (1992). Genetic algorithms. *Scientific American*, 267(1): 66–73.

[2] Barile, P., Ciesielski, V., Berry, M., and Trist. K. (2009). Animated drawings rendered by genetic programming. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*.

[3] Trist, K., Ciesielski, V., and Barile, P. (2010). Can't see the forest: using an evolutionary algorithm to produce an animated artwork. In Huang, F., and Wang, R.C. (Eds.) *Arts and Technology*. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, 30, Springer.

[4] Xuerui, C. (2020). Three-dimensional image art design based on dynamic image detection and genetic algorithm. *Journal of Intelligent and Fuzzy Systems: Applications in Engineering and Technology*, 40(4): 1–10.

[5] Bergen, S.R. (2009). Evolving stylized images using a user-interactive genetic algorithm. In *Proceedings of the 11th Annual Conference Companion on Genetic and Evolutionary Computation Conference*, 2745–2752.

[6] Bergen, S.R. and Ross, B.J. (2012). Automatic and interactive evolution of vector graphics images with genetic algorithms. *The Visual Computer*, 28: 35–45.

[7] Chambel, T., Correia, L., Manzolli, J., Miguel, G.D., Henriques, N.A.C., and Correia, N. (2007). Creating video art with evolutionary algorithms. *Computers & Graphics*, 31(6): 837–847.

[8] Li, H. (2024). Application of using improved genetic algorithm in art design. *Journal of Electrical Systems*, 20: 1603–1612.

[9] Cook, T.E. (2007). GAUGUIN: generating art using genetic algorithms and user input naturally. In *Proceedings of the 9th Annual Conference Companion on Genetic and Evolutionary Computation*, 2647–2650.

[10] Dijkzeul, D., Brouwer, N., Pijning, I., Koppenhol, L., and van den Berg, D. (2022). Painting with evolutionary algorithms. In Martins, T., Rodríguez-Fernández, N., and Rebelo, S.M. (Eds.) *Artificial Intelligence in Music, Sound, Art and Design*. Lecture Notes in Computer Science, 13221, Springer.

[11] Feng, S.Y., and Ting, C.K. (2014). Painting using genetic algorithm with aesthetic evaluation of visual quality. In Cheng, S.M., and Day, M.Y. (Eds.) *Technologies and Applications of Artificial Intelligence*. Lecture Notes in Computer Science, 8916, Springer.

[12] Guo, C., Dou, Y., Bai, T., Dai, X., Wang, C., and Wen, Y. (2023). ArtVerse: A paradigm for parallel human–machine collaborative painting creation in metaverses. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 53(4): 2200–2208.

[13] Jin, X. (2023). Application of machine vision based on genetic algorithm in image painting style method and image processing optimization. *Soft Computing*.

[14] Xie, J., Yu, M., and Liu, G. (2024). Fusing algorithms for intersection of computer science and art: innovations in generative art and interactive digital installations. *IEEE Access*, 12: 173255–173267.

[15] Witbrock, M., and Neil-Reilly, S. (1999). Evolving genetic art. In Bentley, P.J. (Ed.) *Evolutionary Design by Computers*, Morgan Kaufmann.

[16] Liu, D.-M., Li, S.-W., Zhou, R.-Y., Liang, L.-L., Hong, Y.-G., Zeng, Y.-Z. Chang, X., Li, L.-J., Xu, T.-S., Chao, F., Shang, C., and Shen, Q. (2025). NADM: Noise-aware diffusion model for landscape painting video generation. *IEEE Transactions on Cybernetics*, 55(8): 3686–3698.

[17] Hai, C. (2020). The fractal artistic design based on interactive genetic algorithm. *Computer-Aided Design and Applications*, 17(S2): 35–45.

[18] Shyu, M.-S., and Leou, J.-J. (1998). A genetic algorithm approach to color image enhancement. *Pattern Recognition*, 31(7): 871–880.

[19] Sizikova, E., and Funkhouser, T. (2017) Wall painting reconstruction using a genetic algorithm. *Journal on Computing and Cultural Heritage*, 11(1): 1–17.

[20] Luo, Z., Wang, X., Pellikka, P., Heiskanen, J., and Zhong, Y. (2024). Unsupervised adaptation learning for real multiplatform hyperspectral image denoising. *IEEE Transactions on Cybernetics*, 54(10): 5781–5794.

[21] Cao, Z., Lin, C., Zhou, M., and Wen, X. (2024). Learning-based genetic algorithm to schedule an extended flexible job shop. *IEEE Transactions on Cybernetics*, 54(11): 6909–6920.

[22] Grefenstette, J.J. (1986). Optimization of control parameters for genetic algorithms. *IEEE Transactions on Systems, Man, and Cybernetics*, 16(1): 122–128.

[23] Zhou, Y., Yang, N., Huang, X., Lee, J., and Kwong, S. (2024). A novel multiobjective genetic programming approach to high-dimensional data classification. *IEEE Transactions on Cybernetics*, 54(9): 5205–5216.

[24] Shuaib, I., Osaghae, E., and Frederick, B. (2022). Cancer detection using deep learning: a survey. *Journal of Applied Artificial Intelligence*, 3(1): 75–82.

[25] Hussain, S., Tahsin, A., and Sami, A. (2023). Security analysis of the data of social networks using AI techniques. *Journal of Applied Artificial Intelligence*, 4(2): 22–30.

[26] Sani, G., Oladipo, F., Ogbuju, E., and Agbo, F.J. (2022). Development of a predictive model of student attrition rate. *Journal of Applied Artificial Intelligence*, 3(2): 1–12.

[27] Pierre, S., and Legault, G. (1998). A genetic algorithm for designing distributed computer network topologies. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics*, 28(2): 249–258.

[28] Jones, M.E., and Agah, A. (2002). Evolution of digital images. *IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews*, 32(3): 261–271.

[29] House, A., and Agah, A. (2016). Autonomous evolution of digital art using genetic algorithms. *Journal of Intelligent Systems*, 25(3): 319–333.

[30] Chen, S. (2024). Genetic algorithm based on operator optimization in illustration art design. *Journal of Electrical Systems*, (20): 337–342.

[31] DiPaola, S.R., and Gabora, L. (2007). Incorporating characteristics of human creativity into an evolutionary art algorithm. In *Proceedings of the 9th Annual Conference Companion on Genetic and Evolutionary Computation*, 2450–2456.

[32] Bengesi, S., El-Sayed, H., Sarker, M. K., Houkpati, Y., Irungu, J., and Oladunni, T. (2024). Advancements in Generative AI: A comprehensive review of GANs, GPT, autoencoders, diffusion model, and transformers. *IEEE Access*, 12.

[33] Tinago, N., Verkijika, S. F., and Eva Mamabolo, K. (2025). Deepfakes in visual art: Differentiating AI-generated art from human art using Convolutional Neural Networks (CNN). *IEEE Access*, (13): 141484–141495.

[34] Bird, J.J., Barnes, C.M., Lotfi, A. (2024). AI generated art: Latent diffusion-based style and detection. In: Naik, N., Jenkins, P., Grace, P., Yang, L., Prajapat, S. (Eds.) *Advances in Computational Intelligence Systems*. Advances in Intelligent Systems and Computing, 1453, Springer.

[35] Wang, J., Yuan, X., Hu, S., and Lu, Z. (2026). AI vs. human paintings? Deciphering public interactions and perceptions towards AI-generated paintings on TikTok. *International Journal of Human–Computer Interaction*, 42(5): 3307–3330.

[36] Pal, L. (2026). AI-generated art: Copyright, creativity, and ethical ownership. *Journal of Responsible AI and Ethics*, (3)01.

[37] Bian, C., Wang, X., Huang, Y., Zhou, S., and Lu, W. (2025). Effects of AI-generated images in visual art education on students' classroom engagement, self-efficacy and cognitive load. *Humanities and Social Sciences Communications*, (12)1548.

[38] Sadikhova, S. (2026). Neural networks as artists: Exploring AI in contemporary painting. *Acta Globalis Humanitatis et Linguarum*, (3)1.

[39] Deckker, D., and Sumanasekara, S. (2025). A review of AI-powered creativity: The intersection of AI and the arts. *International Journal of Global Economic Light*, (11)4.

[40] Python. (2025). https://www.python.org/.

[41] Pycairo. (2025). https://cairographics.org/pycairo/.

[42] NumPy. (2025). https://numpy.org/.

[43] Leong, W.Y. (2025). AI-generated artwork as modern interpretation of historical paintings. *International Journal of Social Sciences and Artistic Innovations*. 5(1).

[44] Cunningham, J. (2025). Painting in gray: the legal and ethical ambiguities of AI-generated art. *Journal of Information, Communication and Ethics in Society*, (23)3.

[45] Hwang, Y., Shin, D., and Lee, J.H. (2025). Who owns AI-generated artwork? Revisiting the work of generative AI based on human-AI co-creation. *Telematics and Informatics*, 98.

Appendix A

Table 1. Shape representation.

Shape	Bit Representation
Line	000
Circle	001
Triangle	010
Rectangle	011
Pentagon	100
Hexagon	101
Septagon	110
Octagon	111

Table 2. Size representation.

Size (X or Y)	Bit Representation
4 px	000
8 px	001
16 px	010
32 px	011
64 px	100
128 px	101
256 px	110
512 px	111

Table 3. Color representation.

Color	Hex Value	Bit Representation
Black	#000000	000
Red	#ff0000	001
Orange	#ff6600	010
Yellow	#ffff00	011
Green	#008000	100
Blue	#0000ff	101
Violet	#6600ff	110
White	#ffffff	111

Table 4. Tint or shade representation.

Tint or Shade (Yes or No)	Bit Representation
NO	0
YES	1

Table 5. Tint and shade representation.

Black or White	Bit Representation
Black	0
White	1

Table 6. Shape representation.

Color	Shade Hex Value	Tint Hex Value
Black	#000000	#7d7d7d
Red	#7f0000	#f7787f
Orange	#7f3300	#f7aa7f
Yellow	#7f7f00	#f7f67f
Green	#004000	#79b77f
Blue	#00007f	#7978fd
Violet	#33007f	#ab78fd
White	#7d7d7d	#ffffff

Table 7. Rotation representation.

Rotation	Bit Representation
0	000
45	001
90	010
135	011
180	100
225	101
270	110
315	111

Table 8. Position representation.

Position (X or Y)	Bit Representation
32	000
96	001
160	010
224	011
288	100
352	101
416	110
480	111

Appendix B

Methodology Algorithm/Pseudocode.

Initialize the first population randomly
Draw each chromosome's canvas and save as a PNG
Use of "GetBits.py" functions to collect attributes of all shapes
Shape, size (x and y), base color, tint (yes or no), which tint/shade, rotation, position (x and y)
Use of "DrawShapes.py" to generate and color canvases with Cairo
Use of Cairo and PIL library to convert SVG surfaces into numpy array for fitness function calculations
The chromosomes are dissected to isolate bits for each attribute
Attributes are mapped to their numeric or dictionary representation
Convert the PNG to a numpy array for analysis
Utilize both the numpy array and the initial chromosome in calculating fitness
Fitness scores are calculated positionally mapped to population
Use RWS to identify which parents will reproduce, and create the next generation
Calculate fitness and repeat for each successive generation

Fitness Rubrics.

Are all shapes the same? Has all different shapes?
White shapes overlapping colored shapes (creates negative space)
Shape colors share a scheme
Complementary
Tints
Shades
Has elements of symmetry
One shape fills canvas
Fills canvas in the background
Shapes have "shadows" under them