

Article

Hybrid Optimization of 3D Rendering Using Genetic Algorithms and Artificial Neural Networks

Rafeek Mamdouh ^{1,*}, Ahmed Hagag ^{1,2} and Ramadan Babers ¹

¹ Information Technology and Computing Program, Faculty of Computer Studies, Arab Open University, El-Shorouk 11211, Egypt; a.hagag@aou.edu.eg (A.H.); r.babers@aou.edu.eg (R.B.)

² Department of Scientific Computing, Faculty of Computers and Artificial Intelligence, Benha University, Benha 13518, Egypt

* Correspondence: rafeek@aou.edu.eg; Tel.: +20-1006336175

Abstract

Demand for high-quality interactive and real-time rendering remains challenging, as it requires balancing image realism with computational resources. Static parameter tuning of traditional approaches cannot provide adaptive rendering according to the varying complexity of dynamic scenes. This limitation arises from two main deficiencies in existing rendering pipelines: reactive methods that only enhance images after rendering without optimizing the renderer itself, and proactive methods that still rely on manual parameter calibration for each scene. These shortcomings are solved by this paper with an innovative optimization method that is a combination of a genetic algorithm (GA) and artificial neural networks (ANNs). This method offers a closed-loop system that is not found in any other static pipeline. Specifically, in our approach, ANNs will be used to predict the renderer's initial parameter values from scene descriptor data, such as the number of polygons, lighting, and materials. After predicting the parameters, GA will optimize them based on the fitness value, which is determined by maximizing one objective (perceptual quality, defined by the SSIM measure) and minimizing another (rendering time). Our approach can be easily implemented within standard pipeline frameworks (Autodesk Maya Arnold).

Keywords: 3D rendering; ray tracing; artificial neural network; genetic algorithm; hybrid optimization; real-time graphics; multimedia systems; imaging

1. Introduction

Interactive multimedia applications have seen tremendous progress, ranging from video gaming to virtual reality and other advanced areas such as digital twins and architectural visualization. These advances place great pressure on the computational capabilities of current-day 3D rendering systems. Modern-day 3D rendering systems must balance photorealistic visual appearance with efficient rendering within limited computing and memory resources. Despite substantial advancements in graphics processing unit design and parallel rendering techniques, 3D rendering continues to face challenges with complex global illumination and other factors.

Conventional pipeline approaches [1,2] primarily use hand-tuned static settings to control sampling, illumination, shading, and the construction of acceleration structures. While the above-mentioned settings may yield acceptable results for certain scenes, their

Academic Editor: Paolo Bellavista

Received: 6 June 2026

Revised: 22 July 2026

Accepted: 27 July 2026

Published: 3 August 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

flexibility for changes in dynamic scenes is limited, and they often lead to poor compromises between rendering accuracy and efficiency. As depicted in Figure 1, modern rendering pipelines comprise several interconnected stages, where improper settings can affect rendering efficiency and quality. Several studies have examined the use of machine learning algorithms in optimization pipelines for rendering tasks [3–5]. It should be emphasized that the application of artificial neural networks (ANNs) can offer significant potential for creating non-linear mappings from inputs to outputs. More specifically, feed-forward ANNs demonstrated good performance for creating those mappings, provided a large amount of input data was available for training the network. Nevertheless, the use of ANN approaches usually requires large amounts of data, and the model’s generalization ability might be questionable across different renderings [6].

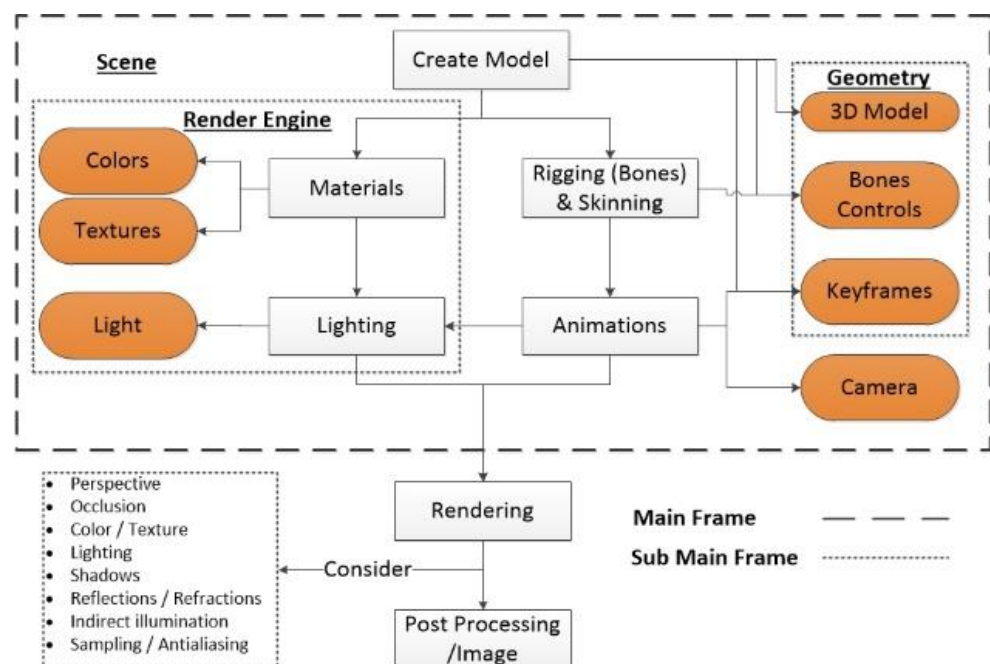


Figure 1. Overview of the core data components and processing stages in the proposed 3D rendering workflow.

On the other hand, genetic algorithms (GA) offer a powerful and population-oriented approach for optimization using evolutionary techniques like selection, crossover, and mutation to solve highly complex multimodal spaces. They can be applied effectively to multi-objective problems with conflicting objectives, such as rendering quality and efficiency. However, pure GA-based approaches to rendering optimization are quite costly, since each new potential solution is evaluated by rendering it several times [7].

The strengths of both ANNs and GAs complement each other, which makes the application of these two together quite logical for the purposes of optimization in rendering. ANNs allow fast inference and learning of complex non-linear mappings from scene descriptors to rendering parameters through offline training, but at the same time suffer from the presence of local optima and are very sensitive to data quality [6]. On the contrary, GAs are capable of conducting robust global search in complex multimodal parameter space without gradient information, making them very suitable for the optimization of multiple contradicting criteria, such as time of rendering and visual quality [7]. Nevertheless, the GA-based optimization is computationally demanding due to repeated fitness evaluation, which requires rendering and cannot be applied in real time [8]. Several recent studies confirmed the efficiency of GA-ANN hybrids and their ability to achieve better results than those obtained by either ANNs or GAs separately [9–12]. Our hybrid

algorithm utilizes the advantages of both approaches and combines them—the ANN provides a good initial solution, while the GA optimizes it locally during 10–20 generations of real-time calculations. This combination of strengths and weaknesses clearly indicates a large research gap, and current rendering optimization techniques can be classified into two types: those that use fixed sets of parameters for optimization and post-rendering techniques, which do not offer any intelligent, closed-loop system to optimize pre-rendering parameters.

The approach to overcoming this constraint is to develop a combined GA-ANN model for adaptive rendering optimization in 3D graphics processing. The approach in question is based on a two-step optimization algorithm. First, an ANN estimates an initial set of near-optimal rendering parameters, considering the geometry's complexity, lighting conditions, materials, and camera settings. Next, GA optimizes the set of parameters via multi-objective optimization, subject to constraints on perceptual quality and rendering performance. This is done by using a function that evaluates the structural similarity index (SSIM) and rendering speed simultaneously.

The suggested framework uses an adaptive closed-loop optimization process that adjusts rendering parameters based on scene and perception requirements. By leveraging the rapid decision-making of artificial neural networks and the search capabilities of genetic algorithms, the framework addresses several shortcomings of conventional static rendering approaches and standalone optimization schemes. The results presented in the paper show a significant increase in rendering speed without degrading perceptual image quality. The main contributions of this work are summarized as follows:

- Formulation of an optimization model for intelligent parameter adjustment for scene-aware rendering based on the combination of ANN and GA algorithms.
- Derivation of a unified scene descriptor for capturing geometric, lighting, material, and camera characteristics of scenes.
- Conception of an effective GA that ensures quick optimization of parameter values and is thus applicable to fast rendering processes.
- Incorporation of a threshold-based perceptual quality assessment scheme under the guidance of SSIM.
- Creation of an efficient technique for scene-aware rendering that reduces rendering time by up to 98%.

The rest of the paper is structured as follows. Section 2 gives an overview of some of the most important developments in rendering optimization and neural rendering. Section 3 introduces the hybrid GA-ANN approach proposed for optimization. Section 4 provides details on the experiments conducted and the resulting quantitative results.

2. Materials and Methods

However, even today's efforts to produce realistic and interactive 3D graphics through rendering are constrained by a tradeoff among visual quality, efficiency, and memory usage. Today's 3D graphics processing pipelines, including geometric pipelines, image-based rendering (IBR) pipelines, and neural pipelines, are all complex, with numerous parameters that must be manually determined or optimized for specific scenes. This chapter provides an overview of the current research and discusses how each addresses the aforementioned challenge and why our suggested framework is needed.

2.1. Real-Time Ray Tracing and Rendering Optimization

Awareness of performance limitations in modern rendering engines is critical to any optimization strategy. In a detailed investigation of the effect of render times and energy consumption in ray tracing, Vasiou et al. [13] concluded that system memory was the

main consumer of energy. The demands for real-time performance and improved image quality have fueled efforts to develop specialized hardware and optimization techniques. Reynolds et al. [14] proposed a method for generating occlusion maps through per-pixel quads. This results in the generation of high-quality shadow maps capable of supporting adaptive level-of-detail rendering. Studies in hardware-based ray tracing established the path for real-time ray tracing. Purcell et al. [15] were the first to propose an effective approach to GPU-based ray tracing through grid and triangle representation of the geometry. A ray tracer that is efficient to use on an FPGA and CPU was introduced by Fender and Rose [16], where they used efficient task scheduling and memory access techniques.

2.2. Geometric Acceleration Structures and GPU-Based Rendering

Apart from the hardware components, the algorithmic improvements focused on optimizing rendering performance through geometric representations and compression. Edenhofer et al. [17] developed a series of techniques that enable artists to control final renders, including realistic RGBZ scenes using screen-space techniques. Lousada et al. [18] presented parallel algorithms for GPU ray tracing, focusing on memory usage and bandwidth compression, while noting that Bounding Volume Hierarchy (BVH) construction time remains an issue for predictive optimization.

Other techniques for acceleration have also been studied. For instance, Feldmann [19] applied the R-tree structure for accelerating ray tracing, using traversal algorithms that do not need pre-processing and compete in efficiency with KD-trees.

2.3. Production Rendering Engines and Manual Parameter Tuning

However, the main difficulty remains the need for manual parameter adjustments. In this respect, Tatarchuk et al. [1] presented state-of-the-art improvements, including two techniques considered most relevant to this research. First, the Mega Light feature in Unreal Engine 5 enables “orders of magnitude more dynamic and shadowed area lights” through stochastic direct lighting. It should be noted that Narkowicz and Costa say that “light sampling techniques, particularly explicit ones, have not been easily scalable to the needs of next-generation consoles by hand tuning.” Second, Sousa discussed the shift towards real-time global illumination within the idTech 8 DOOM: The Dark Ages. The two engines discussed are clear examples of why our approach is justified, as they are the most advanced production rendering engines that require dynamic control systems.

2.4. Neural Rendering and Novel View Synthesis

In fact, image-based rendering offers a different approach from geometry-based rendering techniques, particularly for novel view synthesis with sparse input. The technique of “multi-morphing” was introduced by Seibt et al. [20] as a new approach to IBR that creates an “image graph,” that is, a spatial data structure consisting of 2D images, in order to perform novel view synthesis in 6-Dof without knowing the depths of every pixel in the image or constructing the 3D model. It is demonstrated here that VR-compatible frame rates (>90 FPS) can be achieved with relatively old hardware, though with limitations such as visual artifacts (e.g., ghosting due to parallax), the need for proper camera calibration, and user-defined thresholds.

Aryal et al. [3] investigated optimized 3D Gaussian Splatting for efficient real-time scene rendering. Their work focused on improving rendering efficiency through optimized scene representation and parameter tuning for Gaussian primitives. Nevertheless, adaptive optimization of rendering engine parameters before image generation was not addressed.

2.5. 3D Gaussian Splatting and Neural Graphics Optimization

Neural rendering algorithms have made tremendous strides toward achieving real-time novel view synthesis; however, they also pose certain difficulties in parameter optimization. In their study, Aryal et al. [3] provided a detailed analysis of 3D Gaussian Splatting (3DGS) used for real-time 3D scene generation and concluded that, in contrast to other techniques based on neural radiance fields (NeRF), 3DGS required less training time and memory resources by a factor of 67% and performed real-time rendering at a speed of 89+ FPS. They also addressed the remaining problems related to: (τ_p, τ_s) and pruning parameters, reliance on COLMAP for camera pose estimation, and ad hoc post-processing. These papers clearly indicate that while 3DGS is a highly capable rendering engine, it lacks a smart, adaptive control system.

Yura et al. [4] proposed EventSplat, a technique for novel view synthesis in scenarios with high movement using event cameras. In this paper, we have found some support for our proposed system in one respect: the use of event-to-video initialization techniques for SfM. The authors clearly state that “reconstruction of 3D data purely from events is quite difficult,” which itself supports our ANN-based initialization approach.

Moenne-Loccoz et al. [21] proposed Exact Volumetric Ellipsoid Rendering (EVER), which specifically addresses the main issue of 3DGS: “popping” effects caused by incorrect sorting. EVER runs smoothly at 30 FPS and 720p on an NVIDIA RTX 4090, but with new parameters, including density σ , α_k mapping for opacity, and anisotropy regularizer loss weights. Solving one optimization problem yields yet another that must be solved by hand.

The theme of optimization-based rendering is prevalent across application areas, indicating the need for adaptive parameter tuning. Balloni et al. [22], for example, focused on fashion item reconstruction and discovered that, although 3DGS outperformed NeRF in perceptual metrics (PSNR, SSIM, and LPIPS), an inherent compromise always existed between the quality of rendering and geometry, which is consistent with the compromise that we try to find with our GA component. In addition, the findings from Palma and [23], which attempted a hybrid combination of Order Independent Transparency and Direct Volume Rendering, highlighted the gains that could be made from an optimization-dependent rendering system, as their system’s effectiveness depended on optimizing the different passes during rendering. Finally, the comprehensive work on the VFX pipeline by Zhang [24], while analyzing the various stages involved in the visual effects process, including 3D modeling and level of detail, brings another factor to light that points towards the need for an optimization-based system, since the complications associated with modern rendering processes have now touched a threshold, beyond which there is no way forward with static methods, as we did in our system.

Extending this theme, Fang et al. [25], similarly, through careful tuning of the parameters of 3DGS, they showed that the optimized set can lead to gains in PSNR and SSIM by 4.3% and 0.2%, respectively, supporting the notion that even advanced neural renderers need parameter tuning.

Jiang et al. [26] introduced AnySplat, a feed-forward neural framework capable of rapidly estimating scene representations from unconstrained images. The proposed architecture significantly reduces inference time for neural rendering, although it focuses primarily on scene reconstruction rather than adaptive rendering parameter optimization.

2.6. Deep Learning for Perceptual Rendering Enhancement

A parallel track focuses on quality gaps by using a post-processing approach based on deep learning. RenderGAN [5] by Mameli et al. uses GANs to reduce the gap between real-time renderers (Blender Eevee) and offline path tracers (Blender Cycles), achieving

a UIQI score of 0.898. Thus, it becomes evident that a DNN can upgrade poor-quality real-time renders to a level of ground-truth path-traced images.

There are two aspects of RenderGAN's relevance to our research. First, it provides an insight into the possibilities of neural networks in rendering. Second, it helps us to identify the exact gap that needs to be filled within our framework. The problem addressed by RenderGAN is the post-processing of a pre-existing, low-quality real-time render to enhance its quality by adding "missing" details. It solves symptoms rather than causes. We attempt to shift focus and ask not "How can we enhance the quality of a rendered scene?" but "How can we set up the scene for optimal rendering?"

Yin et al. [27] proposed ENTIRE, a learning-based framework for predicting rendering time using scene characteristics. Their method enables fast estimation of rendering performance without exhaustive simulations. However, the framework predicts rendering time only and does not optimize rendering parameters to jointly improve rendering efficiency and image quality.

2.7. Hybrid AI-Based Rendering Optimization

The application of genetic algorithms along with artificial neural networks was found to be effective for different kinds of optimization problems. The success of ANNs has been shown by Kanta and Sangwan [6], whereas the same authors used GA with an ANN to optimize surface roughness. Li et al. [7] used GA to optimize the feature selection process in backpropagation neural networks, resulting in accurate classification of ECG signal patterns. Xie and Yuille [28] have discussed genetic algorithms in the context of network structure evolution, where "optimizing structure and weights simultaneously" can be considered the most closely related problem to neural rendering.

For the purposes of our paper, we will mention another work on neural rendering based on hybrid AI technologies. Pham et al. [29] introduced an architecture composed of a GA, variational autoencoders, and Pix2Pix for optimizing the layouts of apartments. GA allows for efficient search of the latent space. This is another key reference confirming the "GA-neural synergy" concept used here in the 3D rendering environment. Recent studies across diverse domains have validated the effectiveness of GA-ANN hybrids in achieving superior optimization performance compared to standalone approaches [9–12].

Tan and Moretti [30] proposed a lightweight rendering parameter optimization framework based on XGBoost-driven lookup tables for real-time rendering. Their approach predicts rendering parameters efficiently with low computational overhead. However, the optimization process relies on learned lookup tables rather than iterative multi-objective optimization, making it less flexible for highly dynamic scenes with varying rendering characteristics.

2.8. User Perception and Quality-Driven Rendering

Our technical evaluation draws its critical support from the empirical work by Zhang et al. [31] on the effect of rendering quality on Metaverse consumer groups. They found the Stimulus–Organism–Response framework applicable to explaining the correlation between 3D realism and graphical vividness, direct outcomes of the rendering process, and users' immersion and empathy, the two factors that drive their virtual consumer behavior. It highlights the importance of rendering optimization not only from a technical perspective but also from a commercial point of view.

2.9. Physically Based Rendering and Adaptive Parameter Control

Another connection between photorealistic rendering and scientific research, based on the paper by Furquez Herrera [2], concerns the implicit relationship between these two areas of work. In this thesis, we analyze the application of V-Ray for computational

lighting calculations within architecture. Moreover, we empirically validate the “optimization bottleneck” for a production-ready rendering workflow. To achieve quantitative accuracy in the simulations, interdependent parameters were optimized manually: luminous flux divided by π , reflectance and roughness values, camera parameters, and global illumination parameters. Importantly, the author’s optimization approach provides insights into the emerging paradigm that aligns with the one we have proposed. The absence of professional programming skills was not an obstacle for Furquez Herrera, who used GPT-4 to write Python scripts that implement complex glare-analysis calculations (UGR index, solid-angle calculations, Guth Position Index) at the command line. In other words, the author’s human–AI collaboration method resembles the strategy we propose for parameter prediction using a neural network. Indeed, just as in our case, where our ANN learns to perform this task through training data, GPT-4 uses its programming knowledge to convert human instructions into computer code. Iterative parameter tuning in response to discrepancies between simulation and measurement outcomes is another example of validating our proposed GA refinement strategy.

Herein lies the challenge of optimization, clearly articulated in the context of architectural lighting as illustrated above—a scientifically oriented quantitative problem that production-quality physics-based rendering (V-Ray) can solve only through extensive manual tweaking of interrelated parameters. The author’s brilliant idea of using AI-assisted help to solve this optimization problem provides some insight into how to approach this issue; however, this solution is a manually driven, ad hoc one.

2.10. Comparative Analysis of Modern Rendering Optimization Approaches

Table 1 shows a literature-based comparative analysis of recent approaches to rendering and neural graphics optimization and the suggested GA-ANN approach. The following aspects are compared: the goal of the rendering process, the way scenes are represented, the stage of optimization, evaluation measures, and key drawbacks. It should be mentioned that the discussed approaches deal with different rendering problems on different datasets and hardware architectures using different scene representations.

Table 1. Comparative analysis of recent rendering optimization methods.

Method	Main Objective	Input/Scene Representation	Optimization Stage	Reported Evaluation	Main Limitation Relative to This Study
MegaLights and real-time GI systems [1]	Real-time lighting and shadow rendering	Explicit 3D geometry	During real-time rendering	Lighting scalability and runtime performance	Primarily engine-specific and still requires parameter configuration
Multi-Morphing IBR [20]	Novel-view synthesis	Multiple 2D reference images	View synthesis	More than 90 FPS and visual-artifact analysis	Requires calibrated images and may produce parallax and ghosting artifacts
NeRF-based methods [3]	Neural scene reconstruction and novel-view synthesis	Multi-view images and camera poses	Neural representation training	PSNR, SSIM, training time, and rendering speed	Requires scene-specific training and does not optimize Arnold sampling parameters
3D Gaussian Splatting [3,25]	Real-time novel-view synthesis	3D Gaussian representation	Gaussian training, densification, and pruning	FPS, PSNR, SSIM, and memory usage	Requires explicit tuning of

					densification, pruning, and opacity parameters
EventSplat [4]	Novel-view synthesis under rapid motion	Event-camera data	Reconstruction and initialization	Reconstruction quality and novel-view performance	Designed for event-based reconstruction rather than production-renderer parameter control
EVER [21]	Artifact reduction in Gaussian rendering	Volumetric ellipsoid representation	Representation and rendering optimization	30 FPS at 720p and artifact reduction	Introduces additional density, opacity, and regularization parameters
RenderGAN [5]	Enhancement of low-quality rendered images	Previously rendered images	Post-processing	UIQI and perceptual image quality	Enhances the output after rendering rather than optimizing the renderer configuration
Proposed GANN Framework	Adaptive optimization of rendering parameters	Explicitly modeled Arnold scenes and scene descriptors	Before and during parameter refinement	Rendering time, MSE, PSNR, and SSIM	Currently evaluated using Arnold and four Full HD test scenes

As opposed to approaches based on neural radiance field and novel-view synthesis techniques, our approach does not generate a scene out of multi-view images. Instead, it works on explicit 3D models and predicts the sampling and quality parameters that the Arnold rendering engine requires. ANNs provide fast, scene-dependent initialization of parameters, whereas GAs optimize the configuration predicted by ANNs based on rendering time and quality criteria. Hence, the presented framework can be seen as a supplement to neural rendering approaches.

2.11. Research Gap and Motivation

The literature review shows a clear split between two strategic lines. The first is reactive and image-focused. It is represented by RenderGAN [5] and other neural super-sampling techniques [13,25]. This strategy is based on the output of a static renderer and uses deep learning to synthesize missing information. Although it is a proof of the representational capability of neural networks, it cannot compel the renderer to use more samples or to produce better acceleration structures.

The second strategy is proactive and scene-focused. Algorithmic advancements in 3DGS represent it [3], EVER [21], and production engines [1]. However, as Aryal et al. [3] have clearly recorded and shown by the need for manual calibration in EVER [21] and V-Ray [2]. This strategy is currently at the peak of static optimization.

This is the specific research gap we fill: There is no existing system that provides a closed-loop, real-time optimization platform capable of intelligently traversing the enormous, multi-objective parameter space of a modern renderer before image generation. RenderGAN seeks to optimize the final image; our system seeks to optimize the system that generates the image. EventSplat demonstrates the strength of learned priors for initialization; EVER shows that solving new problems requires optimizing new parameters. The architectural case study by Furquez Herrera provides a rigorous reference environment and ground-truth basis for evaluating the proposed optimization framework. Meanwhile, the user study by Zhang et al. provides the perceptual rationale by indicating that appropriately optimized rendering parameters can contribute to enhanced user

immersion. Unlike any other existing approach, our proposed approach combines these different trends into a single framework by leveraging ANN-based fast, scene-aware parameter prediction, which has been verified by Pham et al. [29], EventSplat [4], and RenderGAN [5], and GA-based robust parameter fine-tuning. By directly borrowing methodological insights from latent space optimization [29], we address unmet demands in current state-of-the-art rendering systems: densification thresholds in 3DGS [3], parallax filtering thresholds in multi-morphing [20], density and anisotropy parameters in EVER [21], and photometric calibration parameters for PBR engines [2]. Essentially, our proposed approach shifts the paradigm from post hoc image enhancement to prediction-based parameter control, upgrading the rendering system from a static configuration task to an intelligent, self-optimizing system.

Recent neural radiance methods, including NeRF, EventSplat, EVER, and 3D Gaussian Splatting, primarily address scene reconstruction and novel-view synthesis from multi-view images. In contrast, the present framework operates on explicitly modeled 3D scenes and optimizes the sampling and quality parameters of the Arnold renderer before image generation. Consequently, a direct quantitative comparison would require constructing an additional multi-view capture, camera-pose estimation, training, and novel-view evaluation pipeline. Therefore, the present study provides a methodological comparison with recent neural rendering approaches, while a controlled experimental comparison using common scenes, hardware, and quality metrics is identified as future work.

3. Proposed Hybrid GA-ANN Framework

The following section gives a brief description of the design of the proposed GA-ANN hybrid system, which compensates for the limitations posed due to fixed values of the rendering parameters in the present graphics pipeline systems, as illustrated in Figure 1. The methodology adopted in this research includes two stages as follows:

Stage 1: Predicting rendering parameter values using the feed-forward ANN model.

Stage 2: Further refining the rendering parameter values by using GA.

3.1. Extraction of Scene Descriptors

For all 3D scenes, the corresponding scene descriptors are collected, and then the generated set is used as the input data of the ANN; see Figure 2. The reliable indicators, which were determined to predict rendering time and image quality, include: geometry complexity, expressed by the number of vertices, number of triangles, geometry depth complexity and distribution of level-of-details; material complexity, expressed by the number and material types, shader complexity, texture density; lighting complexity, expressed by the number of dynamic lights, global illumination, shadow settings; and camera settings, expressed by camera angle, distance to the object and field of view. Such scene descriptors are compiled into a feature vector (s). Recently, the significance of such factors as texture density and number of triangles was demonstrated during the experiments for multi-resolution rendering [32], where an adaptive approach showed extremely high accuracy. Moreover, the ENTIRE method [27] recommended taking into account just a few scene descriptors to describe scene performance. We will use this experience in our framework.

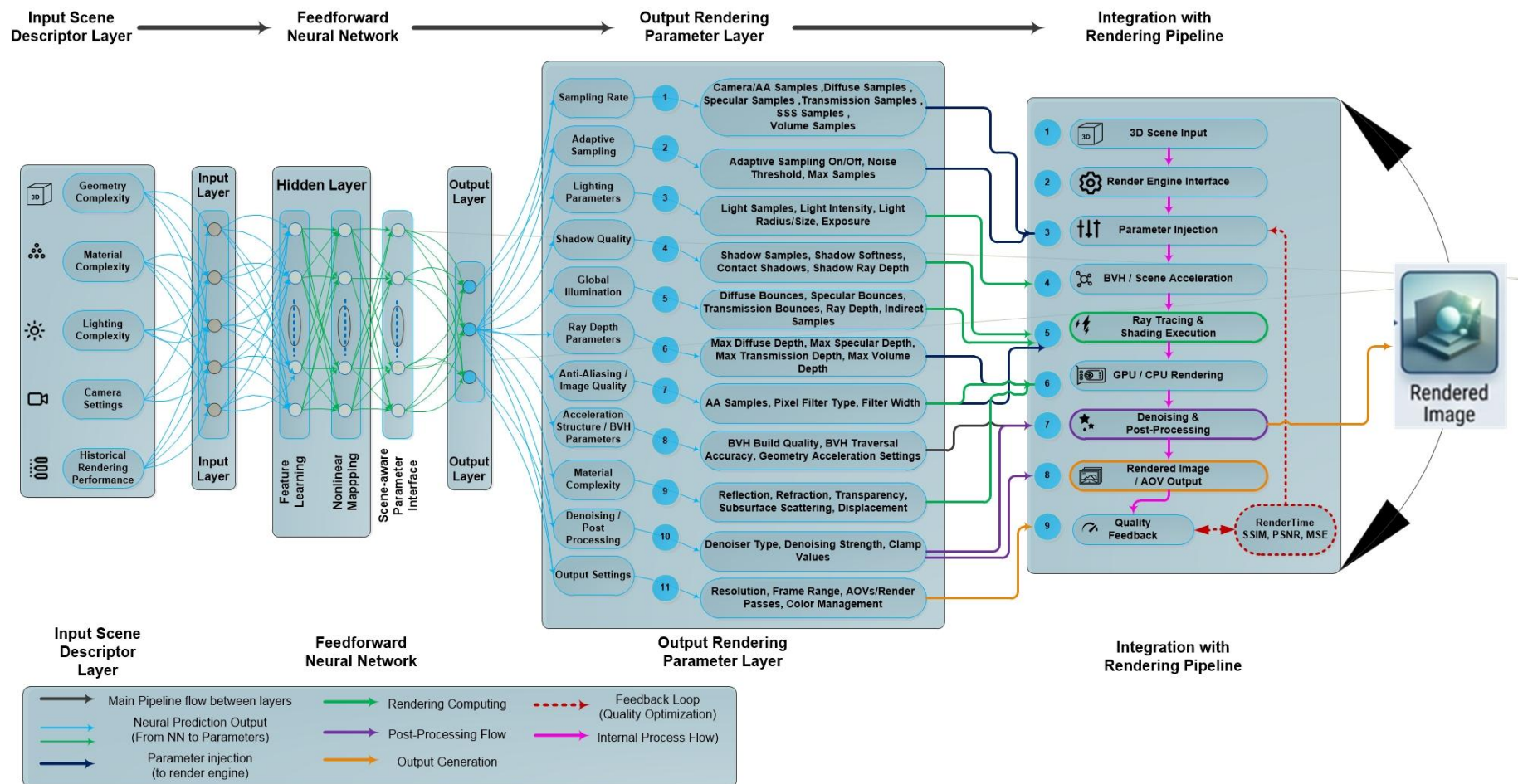


Figure 2. Proposed scene-aware feedforward ANN-based framework for rendering parameter prediction and integration with the rendering pipeline.

3.2. ANN-Based Parameter Prediction

The mapping from the scene descriptor vector (s) into the initial set of rendering parameters p_0 is accomplished via a multi-layer feed-forward neural network (ANN), as illustrated in Figure 2. (a) ANN architecture: The architecture consists of an input layer which has the same dimension as (s), two hidden layers employing the ReLU activation function, dropout, and batch normalization in order to stabilize training, and a linear output layer for regression. (b) ANN output: The ANN aims at predicting the following key rendering parameters: the ray sampling ratio, light samples, Bounding Volume Hierarchy (BVH) quality, anti-aliasing cutoff, and global illumination depth. (c) ANN training: It is done in an offline manner with the help of a supervised learning technique, where the training data consists of previously rendered images. The optimal parameters are obtained via a brute-force search of the multi-objective fitness criterion, which considers the render time and the SSIM [33] of the image quality. Several improvements have been made to the area of feed-forward neural rendering in recent years. Among those improvements, one is AnySplat [26], which performs real-time inference on the primitive distribution and the camera pose from uncelebrated images, demonstrating that real-time inference of a neural scene is feasible. Similarly, ENTIRE [27] proposes a two-stage autoencoder structure, which infers the volume rendering times for a particular scene by making predictions about its properties via fast inference. Similarly to ENTIRE, we follow the same concept in our ANN architecture. Below is the description of the proposed ANN architecture in Table 2. The prediction architecture is a feed-forward neural network that predicts continuous-valued parameters for scene-based rendering.

Table 2. Architecture of the proposed ANN model.

Layer	Type	Activation Function	Description
Input Layer	Dense	---	Accepts scene descriptor data comprising geometry, illumination, materials, and camera-related aspects.
Hidden Layer 1	Dense	ReLU	Obtains low-level scene-complexity descriptors for estimating rendering cost.
Hidden Layer 2	Dense	ReLU	Captures the complex relationship between scene attributes and adaptive rendering techniques.
Output Layer	Linear	Linear	Determines adaptive render settings such as AA, sampling rates, BVH quality, and global illumination settings.

In the proposed ANN architecture, two hidden fully connected layers with ReLU activation functions are used to learn the highly nonlinear relationship between scene attributes and rendering options. The first hidden layer learns intermediate scene-complexity representations, while the second hidden layer performs nonlinear mapping between these representations and the adaptive rendering parameters. For the output layer, a linear activation function is used because the task is formulated as a regression problem over continuous rendering parameters such as sampling frequency, anti-aliasing, BVH precision, light sampling, and global illumination.

Dropout and batch normalization are applied to stabilize the training process and reduce the risk of overfitting. In addition, the network is trained using supervised learning based on rendered datasets obtained by sweeping rendering parameters across various benchmark scenes.

- Prediction: The output of the model consists of key render settings, including sample rate, lighting samples, BVH accuracy, and anti-aliasing.

- Training: The network is trained based on supervised learning, using rendered images together with their optimal rendering settings. These rendering settings are computed using a multi-objective fitness function that accounts for rendering time and SSIM.

Figure 2 illustrates the design for the scene-aware feed-forward ANN framework for parameter prediction and pipeline integration during rendering. The ANN takes input scene descriptors in terms of geometry complexity, material complexity, lighting complexity, camera parameters, and previous rendering performance data. Then, these descriptors are mapped to output rendering parameters, which are eventually fed back to the rendering pipeline. In the given architecture, the first hidden layer is used for learning the intermediate representation of the input scenes, and the second hidden layer does a nonlinear mapping between the input and output scene characteristics.

The role of the ANN in the overall hybrid GA-ANN architecture is further clarified in Figure 3. Instead of being utilized as a stand-alone optimization algorithm, the ANN serves as a starting point that estimates the values of the rendering parameters (p_0). These initial parameter values are further optimized by the GA phase to obtain the optimal parameter values (p^*), which will then be fed into the Maya/Arnold (Available at: <https://www.autodesk.com/in/products/maya> (accessed on 6 May 2026)) rendering process.

The two layers of hidden neurons were chosen as an efficient architecture that provides sufficient capacity for nonlinear modeling without introducing additional computational burden. Using a single layer of hidden neurons could be less effective in decoupling the feature extraction process from the nonlinear parameter-mapping task, whereas a deep network could increase the complexity of the model. Hence, the two hidden layers of ANNs were used as an efficient architecture for producing the initial prediction of the rendering parameters.

However, we note that the current architecture is not globally optimal either. Other neural network architectures might yield similar or even better performance after appropriate optimization. Hence, the current manuscript has been revised to reflect this limitation explicitly, and future work will involve conducting ablation studies on varying depth, number of neurons, activations, optimizers, learning rate, and regularizations. Such models will be analyzed based on render time, MSE, PSNR, and SSIM.

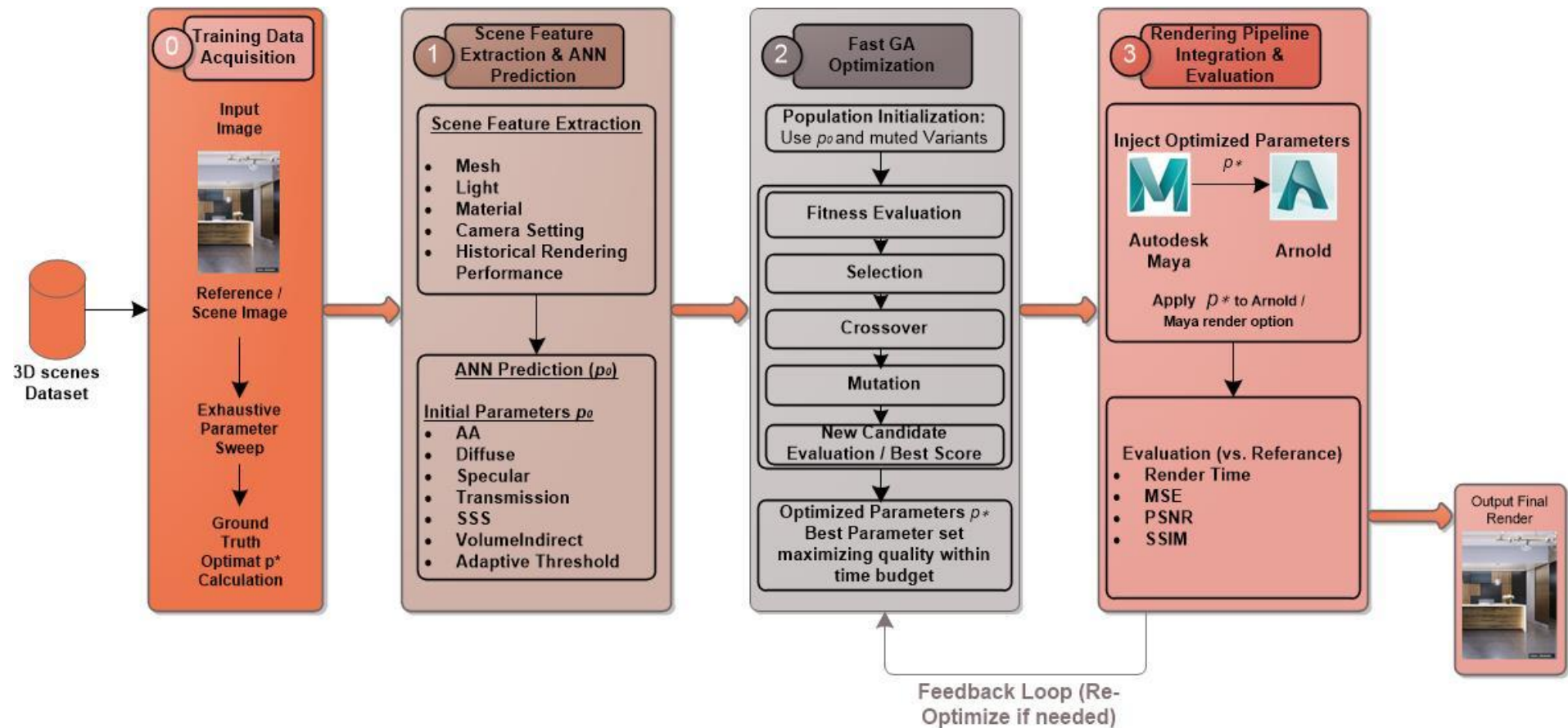


Figure 3. A hybrid GA-ANN framework for adaptive rendering parameter optimization.

3.3. GA Optimized Rendering

After the ANN prediction, the GA implements local search to find the optimal p_0 configuration that is denoted as p^* . The real-time GA implementation is achieved using a small population size and a limited number of generations.

- (1) Population initialization takes place using the predicted parameter p_0 ; other individuals are produced through a slight mutation of p_0 .
- (2) According to Tan et al. [30], a weighted multi-objective fitness function is employed to evaluate candidate rendering parameter configurations by balancing perceptual image quality and rendering efficiency p :

$$F(p) = w_1 SSIM(p) - w_2 \cdot T_{render}(p) \quad (1)$$

where $SSIM(p)$ represents the perceptual image quality obtained using rendering parameter configuration p , $T_{render}(p)$ denotes the corresponding rendering time, and w_1 and w_2 are user-defined weighting coefficients controlling the trade-off between visual fidelity and computational efficiency. The negative sign in the rendering time term ensures that lower rendering times increase the overall fitness score. $Q(p)$ is a perceptual quality metric (e.g., $SSIM$), $T(p)$ is the measured rendering time, and w_1, w_2 These are user-defined weights. The negative sign on $T(p)$ indicates that lower rendering times improve the fitness score.

- (3) The selection phase involves conducting tournaments to select parents. Uniform crossover creates new individuals by combining attributes, whereas Gaussian mutation introduces slight random variations [8].
- (4) The termination process consists of running the program for several generations (10–20 generations). The best solution obtained at the end of the final generation will become p^* , which produces optimal values. Several recent works have applied GAs to material and render parameters [8].

3.4. Integration with Rendering Pipeline

The proposed framework should be simple to integrate into a standard 3D rendering pipeline, such as Autodesk Maya with the Arnold renderer. From Figure 1, we observe that. p^* The parameters after tuning serve as input for obtaining the final rendering result. In real time, the framework is applied to each frame, with predictions based on previously rendered scene descriptors. Specifically, our framework is implemented as a Python script executed in the Maya Script Editor. The proposed architecture combines techniques of scene analysis, parameter estimation using artificial neural networks, GA-based optimization, and an interface specifically designed to interact with the Arnold Renderer [34].

3.5. Training and Evaluation Protocol

- Training Data Set: Different types of 3D environments, including the scene involving the logo of the studio and the Car Scene (Section 4), will be created using an exhaustive search of all the parameters to find the best parameters to train the ANN [30].
- Validation Phase: The proposed algorithm will be validated through test scenes that have not been encountered before with respect to different factors such as time required for rendering, metrics, and rate of convergence [27,30,32].

4. Experimental Results and Analysis

In this section, a combination of ANN and GA to optimize rendering parameters will be analyzed to reduce render time while maintaining good visual quality. This analysis is

based on four test scenes, a consistent hardware configuration, and four performance measures

4.1. Datasets Benchmark Scenes

To analyze and evaluate the performance of the algorithm for different types of scenes and rendering complexities, four exemplary 3D scenes were chosen, as summarized in Table 3, namely a Car Scene, a Studio Logo Scene, an Indoor Residential Scene, and an outdoor industrial Factory Scene. The chosen 3D scenes range from object-based visualization to geometrical graphics, from architectural to interior environments, and also exterior environments. In order to create benchmark reference images, for each 3D scene, an image was rendered with maximum sampling parameters that have been considered in this research work, which are Camera Anti-Aliasing (AA = 6) and Subsurface Scattering (SSS = 4).

Table 3. Classification of the selected 3D test scenes according to scene type and principal rendering complexity.

Scene	Scene Category	Main Rendering Complexity
Car Scene (Mitsubishi)	Product/Object Visualization	High geometric complexity, glossy metallic surfaces, glass components, complex reflections, refractions, and multi-material interactions
Studio Logo Scene (Radio MIX)	Graphical/Object Visualization	Medium geometric complexity, physically based materials, controlled three-point lighting, soft shadows, and sensitivity to indirect-lighting samples
Indoor Residential Scene	Architectural Interior Visualization	Multiple objects and materials, enclosed lighting conditions, indirect illumination, reflections, occlusion, light bouncing, and soft shadows
Outdoor Factory Scene	Industrial Architectural Exterior	Large-scale geometry, environmental illumination, depth variation, distant objects, direct sunlight, and broad outdoor shadows

Selected 3D Test Scenes

The following four 3D scenes were utilized to test the proposed GA-ANN rendering optimization technique at varying geometric, material, and lighting complexities. Studio Logo Scene (Radio MIX): a medium-complexity scene featuring physically based materials and 3-point lighting. Although relatively simple from a geometric perspective, it is prone to lower sampling of indirect light due to the soft shadows and reflections it exhibits. Car Scene (Mitsubishi): a high-resolution scene featuring around two million polygons, gloss metallic materials, glass objects, multiple light sources, and complex reflections that render it appropriate for the most computationally challenging rendering tasks. Indoor Residential Scene: an architectural indoor scene comprising multiple objects, materials, artificial light sources, indirect lighting, reflections, occlusion, and soft shadows. The indoor and outdoor scenes were included following the reviewer's suggestions to increase the test's generalizability. In total, these four scenes include graphical-object rendering, product visualization, indoor architecture, and outdoor industrial visualization. The reference images were obtained with the use of the highest sampling rates that have been included: AA = 6 and SSS = 4. The chosen scenes are presented in Figure 4, while their geometry and settings for rendering are provided in Tables 4 and 5, respectively.

Four scenes were rendered using Arnold with an aiSkyDome environment, and ground truth reference images were produced with maximum sampling levels (AA = 6, SSS = 4). It provides an opportunity to test the proposed approach on various degrees of geometric complexity, indirect illumination, reflections, shadows, and realistic materials.

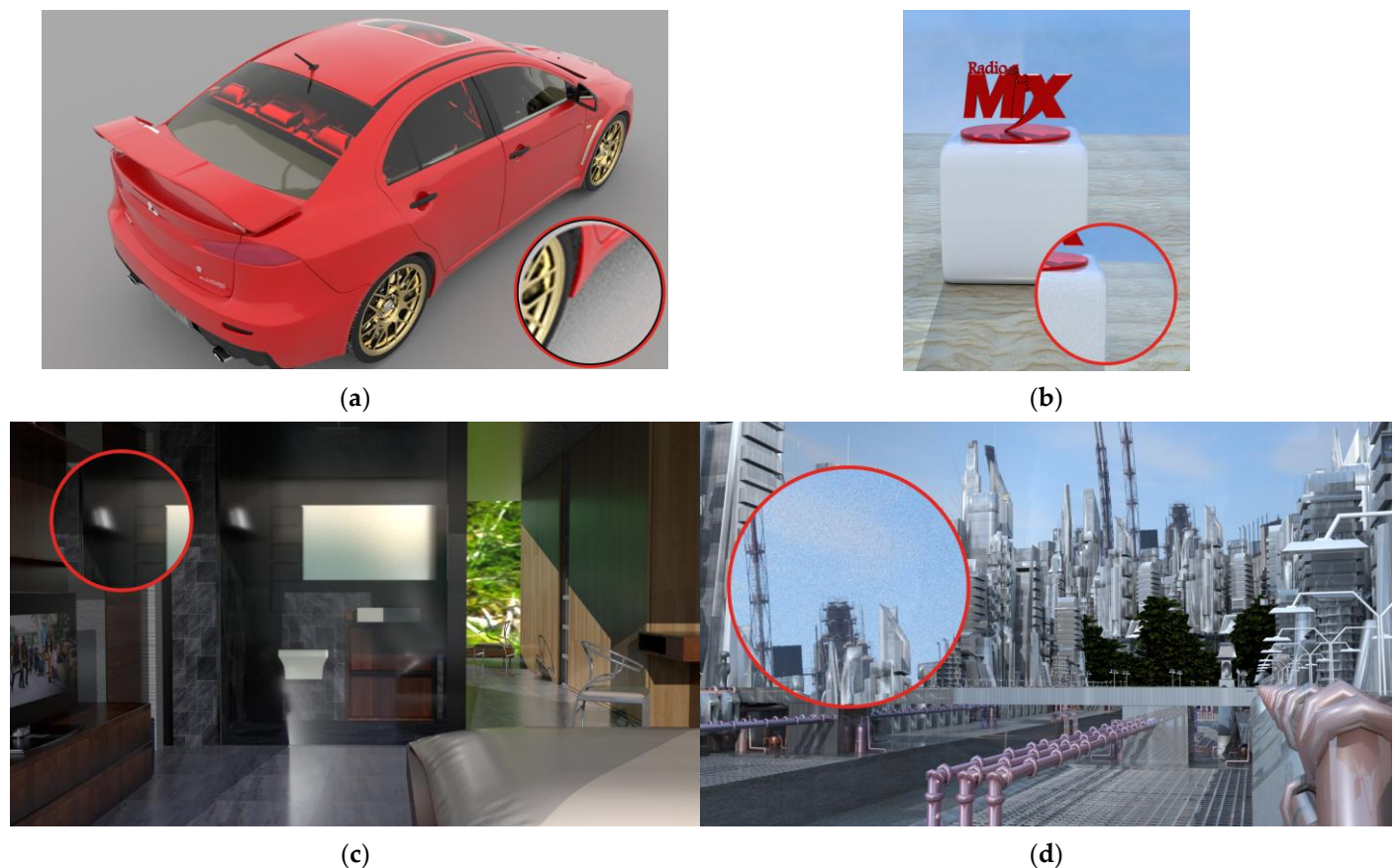


Figure 4. GA-ANN optimized renderings of (a) the Mitsubishi Car, (b) the Studio Logo, (c) the Indoor Residential Scene, and (d) the Outdoor Factory benchmark scenes. The regions enclosed by the red circles show the corresponding GA-optimized results superimposed on the GA-ANN rendered images for direct visual comparison.

Table 4. Specifications of geometry in selected datasets.

Dataset	Vertices	Edges	Faces	UVs
Mitsubishi Car	1,914,763	3,911,381	1,996,425	161,789
Radio MIX Logo	19,073	3658	17,518	21,813
Indoor Residential Scene	130,007	261,823	132,001	353,038
Outdoor Factory	893,017	1,005,616	494,229	556,267

Table 5. Rendering configurations of the four selected 3D test scenes.

Dataset/Scene	Renderer	Lights	Intensity	Primary Material	Material Params	Environment
Mitsubishi Car	Arnold	2× Directional	1.200	AI Gold Metal + AI Glass-Mtoa	Metal: 0.940/Glass: 0.240	aiSkyDome (1.00)
Radio MIX Logo	Arnold	3× Directional	1.00/0.50	AI Paint Finish-Mtoa	Weight: 0.990/Metal: 0.641	aiSkyDome (1.00)
Indoor Residential Scene	Arnold	4× Directional	1.00/3.00	AI Glass + AI Standard Surface-Mtoa	Weight: 0.800/Metalness: 0.000	aiSkyDome (3.00)
Outdoor Factory	Arnold	3× Directional	2.00	AI Paint Finish + ChromeMtoa	Weight: 0.990/Metal: 0.641	aiSkyDome (1.50)

4.2. Framework and Configuration

The framework was implemented in Python 3 and developed for Autodesk Maya 3D using the Maya Script Editor and the Arnold rendering engine. It consists of four main components: scene feature extraction, ANN-based prediction, GA optimization, and a rendering interface. The ANN was trained through supervised learning using parameter-quality sweeps collected from the four selected 3D scenes, where MSE, PSNR, and SSIM were calculated against their corresponding ground-truth reference images. Training was performed using the Adam optimizer with a learning rate of 0.001, Xavier initialization, a batch size of 64, and 5000 epochs. During the GA optimization stage, 10–20 generations were employed using tournament selection, uniform crossover, and Gaussian mutation. The fitness function minimized rendering time while maintaining a predefined minimum structural similarity index (SSIM) value, thereby balancing performance and visual quality. The experiments were conducted on hardware with limited computational resources to evaluate the efficiency of the proposed framework under realistic operating conditions, as detailed in the following hardware specifications.

- Processor: Intel Core i5-2430M @ 2.40 GHz (dual-core, 4 threads)
- Memory: 8 GB DDR3 RAM
- Primary GPU: NVIDIA GeForce GT 525M with 974 MB VRAM (CUDA-enabled)
- Secondary GPU: Intel HD Graphics 3000 (shared memory)
- Storage: Solid State Drive (SSD)
- Rendering Environment: Autodesk Maya with Arnold Renderer
- Implementation Framework: Python 3 using PyTorch, NumPy, OpenCV, and DEAP

4.3. Performance Measures

The proposed hybrid framework was evaluated using four key performance metrics to measure the rendering speed across four scenes and image quality, providing a comprehensive and reliable assessment of the system's efficiency and overall performance.

4.3.1. Rendering Time (Minutes)

Rendering time, the measure of the time taken to generate the final render image, measured in minutes, was one of the important parameters for measuring the efficiency of the developed framework. The visual quality measures were also included in order to make sure that a decrease in rendering time does not negatively affect the quality of the generated image.

As shown in Table 6, the high-quality reference configurations required approximately 120.00–314.35 min per image. The ANN-based prediction reduced the rendering time to approximately 32.58–99.39 min, whereas the GA-optimized configurations achieved substantially lower times ranging from 2.19 to 50.10 min. The largest reduction was observed in the Mitsubishi Car Scene, where the rendering time decreased from 127 min to 2.19 min using GA optimization. These results demonstrate that both optimization stages accelerated the rendering workflow, while the GA produced the greatest time savings across the four evaluated scenes.

Table 6. Rendering-time comparison among the reference, ANN-predicted, and GA-optimized configurations across the four test scenes.

Scene	Reference (min)	ANN (min)	GA-Optimized (min)
Studio Logo (Radio MIX)	120.00	32.58	4.27
Car Scene (Mitsubishi)	127.00	55.56	2.19
Indoor Residential Scene	314.35	99.39	32.15
Outdoor Factory	231.13	97.49	50.10

4.3.2. Image Fidelity: MSE and PSNR (dB)

Mean Squared Error (MSE) and peak signal-to-noise ratio (PSNR) are common metrics for assessing how closely the resulting image resembles a reference image. MSE measures the average of the squared differences between the corresponding pixel values in the two images, while PSNR expresses the same error level in logarithmic terms, measured in decibels (dB), providing a more precise indicator of image quality. Table 7 presents the MSE and PSNR values obtained using both artificial neural network-based and genetic algorithm-based optimization methods across selected scenes.

Table 7. MSE, PSNR, and SSIM values with quality classification for ANN and GA configurations across the four test scenes.

Scene & Method	MSE	PSNR (dB)	SSIM	MSE Level	PSNR Level	SSIM Level/Quality Level	Visual Fidelity	Overall	Decision
Studio Logo—ANN	5.93	40.39	0.947	High	High	High Quality	Strong perceptual match	High	Accept
Studio Logo—GA	22.5	34.61	0.864	Medium	Medium	Medium Quality (Preview)	Acceptable for preview	Medium	Accept—Preview
Car Scene—ANN	0.16	56.1	0.998	High	High	High Quality	Near-perfect reconstruction	High	Accept
Car Scene—GA	5.2	40.97	0.973	High	High	High Quality	High fidelity preserved	High	Accept
Indoor Residential Scene—ANN	4.1186	41.9834	0.9539	High	High	High Quality	Strong perceptual match	High	Accept
Indoor Residential Scene—GA	10.1481	38.0669	0.9014	Medium	High	Medium Quality (Preview)	Acceptable for preview	Medium	Accept—Preview
Outdoor Factory—ANN	4.3134	41.7826	0.9737	High	High	High Quality	High fidelity preserved	High	Accept
Outdoor Factory—GA	29.6738	33.4071	0.8639	Low	Medium	Medium Quality (Preview)	Acceptable for preview	Low	Reject/Not Recommended for final output

Mean Squared Error (MSE)

The Mean Squared Error (MSE) measures the average squared difference between the rendered image and its corresponding reference image. Lower MSE values indicate a higher degree of similarity to the reference image, reflecting better image reconstruction quality. This metric is calculated using the following equation:

$$\text{MSE}(\hat{x}, x) = \frac{1}{n} \sum_{i=1}^n (\hat{x}_i - x_i)^2 \quad (2)$$

Here, n stands for the total number of pixels, $\hat{x}$ is the reconstructed scene, and x is the real scene. The lower the MSE, the better the match between the reconstruction and the expected outcome.

Quality thresholds: $\text{MSE} < 10 \rightarrow \text{high}$ | $10\text{--}25 \rightarrow \text{medium}$ | $> 25 \rightarrow \text{low}$.

Peak Signal-to-Noise Ratio (PSNR)

The peak signal-to-noise ratio (PSNR) is used to assess image quality by measuring the relationship between the maximum signal value and the reconstruction error on a logarithmic scale. It is directly dependent on the MSE value, with higher values indicating better image quality:

$$\text{PSNR}(\hat{x}, x) = 10 \log_{10} \left(\frac{\max(x)^2}{\text{MSE}(\hat{x}, x)} \right) \quad (3)$$

Here, $\max(x)$ stands for the highest intensity in the reference volume. A bigger PSNR value means better reconstruction quality.

Quality thresholds: $\text{PSNR} > 38 \text{ dB} \rightarrow \text{high} \mid 30\text{--}38 \text{ dB} \rightarrow \text{medium} \mid < 30 \text{ dB} \rightarrow \text{low}$.

4.3.3. Synthesis and Key Observations

The structural similarity index (SSIM) is a common metric for assessing image quality. It compares a processed image to a reference image by analyzing its lighting, contrast, and structural characteristics. Unlike metrics that rely solely on error calculations, SSIM focuses on the degree of similarity in visual content between the two images, making it more aligned with human perception of image quality. It is calculated by comparing local statistical properties, such as average intensity, contrast, and structural correlation, to assess the degree of similarity between the output and reference images.

$$\text{SSIM}(\hat{m}, m) = \frac{(2\mu_{\hat{m}}\mu_m + c_1)(2\sigma_{\hat{m}m} + c_2)}{(\mu_{\hat{m}}^2 + \mu_m^2 + c_1)(\sigma_{\hat{m}}^2 + \sigma_m^2 + c_2)} \quad (4)$$

where $\mu_{\hat{m}}$ and μ_m represent local means, $\sigma_{\hat{m}}^2$ and σ_m^2 are variances, and $\sigma_{\hat{m}m}$ denotes covariance. The stabilizing constants are defined as:

$$c_1 = (k_1 L)^2, \quad c_2 = (k_2 L)^2 \quad (5)$$

with $L = \max(x)$ represents the dynamic range of the image pixels ($L = 255$ for 8-bit images) and typical values $k_1 = 0.01$ and $k_2 = 0.03$. The SSIM score ranges from 0 to 1, with values closer to 1 indicating greater structural similarity.

Thresholds for the four scenes: $\text{SSIM} > 0.93 \rightarrow \text{high quality} \mid 0.85\text{--}0.93 \rightarrow \text{medium quality (preview)} \mid < 0.85 \rightarrow \text{low quality}$.

The experiment demonstrated that both methods substantially reduced rendering time compared with the high-quality reference configurations across all four scenes studied. The reduction in rendering time ranged from approximately 56% to 73% when using the ANN technique, thereby maintaining high visual quality for all four scenes—Studio Logo, Car, Indoor Residential, and Outdoor Factory. The GA technique led to much higher reductions in rendering times, from 120 to 315 min, using reference configurations down to approximately 2–50 min; however, the visual quality achieved using GA depended on the scene specifics. While the Car Scene had an excellent visual quality of 0.973 SSIM, the other three scenes—the Studio Logo, Indoor Residential, and Outdoor Factory—had their SSIM values recorded at 0.864, 0.901, and 0.864, respectively. This places the latter three scenes in the medium-quality category and makes them usable only for preview rendering. Furthermore, the GA configuration for the Outdoor Factory scene produced relatively high MSE and low PSNR values, indicating greater visual distortion despite the substantial reduction in rendering time. Overall, these findings demonstrate that optimization effectiveness is strongly influenced by the geometric, material, lighting, and visual characteristics of each scene.

4.3.4. Summary of Performance Measures

To analyze the performance of the proposed framework for rendering optimization in terms of computing efficiency and image quality of the four test scenes, rendering time, mean squared error (MSE), peak signal-to-noise ratio (PSNR), and structural similarity index (SSIM) were considered as evaluation measures. The evaluation parameters and quality thresholds for the Studio Logo, Car, Indoor Residential, and Outdoor Factory scenes are described in Table 8. Each configuration was analyzed ten times under the same operational conditions, and the results were presented by mean $\pm$ standard deviation with 95% confidence intervals. Table 9 describes the performance of the ANN and GA algorithms for the four test scenes.

Table 8. Summary of all evaluation metrics.

Metric	Unit	Optimal Direction	Quality Thresholds	Role
Rendering Time	Minutes	Lower ↓	No fixed threshold—lower is better	Computational Efficiency
MSE	Pixel ²	Lower ↓	High: <10 Medium: 10–<25 Low: ≥25	Pixel-Level Accuracy
PSNR	dB	Higher ↑	High: >38 dB Medium: >30–38 dB Low: ≤30 dB	Signal Fidelity
SSIM	[0, 1]	Higher ↑	High: >0.93 Medium: 0.85–0.93 Low: <0.85	Perceptual Quality

The results show that the proposed framework performs effectively on hardware with limited computational resources, thanks to a two-stage optimization process. The ANN models had good visual quality with all four considered scenes and were thus appropriate for final rendering. On the other hand, the GA models had a higher reduction in rendering time and thus were more appropriate for previews. However, the quality of vision from GA models was dependent on the scene properties. High-quality visualization was possible with the Car Scene; medium-quality images appropriate for previewing could be generated with the Studio Logo and Indoor Residential scenes, whereas the Outdoor Factory Scene had poor visual quality. The GA has a higher-speed performance advantage; however, its appropriateness depends on scene properties.

Table 9. Statistical validation of ANN and GA rendering performance.

Scene	Method	Render Time (min), Mean ± SD	95% CI	SSIM, Mean ± SD
Studio Logo (Radio MIX)	ANN	32.58 ± 1.21	[31.83, 33.33]	0.947 ± 0.004
Studio Logo (Radio MIX)	GA	4.27	—	0.864
Car Scene (Mitsubishi)	ANN	55.56	—	0.998
Car Scene (Mitsubishi)	GA	2.23 ± 0.07	[2.19, 2.27]	0.973 ± 0.003

Statistical analysis was conducted to evaluate the stability and reliability of the two proposed optimization methods, ANN and GA, across the four test scenes. Each configuration was tested repeatedly under identical operating conditions. Mean performance values, standard deviations, and 95% confidence intervals were calculated for rendering time and SSIM to assess the consistency and variability of the results across different runs and scene types.

$$\mu = \frac{1}{N} \sum_{i=1}^N x_i \quad (6)$$

while the experimental variability was measured using the standard deviation:

$$\sigma = \sqrt{\frac{1}{N-1} \sum_{i=1}^N (x_i - \mu)^2} \quad (7)$$

The statistical reliability of the repeated experiments was further quantified using the 95% confidence interval:

$$CI_{95\%} = \mu \pm 1.96 \left(\frac{\sigma}{\sqrt{N}} \right) \quad (8)$$

From the results above, there was a substantial reduction in rendering time with the GA-based approach, but visual quality varied across the four scenes tested. High-level visual quality was retained for the Car Scene, while the Studio Logo and Indoor Residential scenes generated medium-quality images suitable for preview rendering. High levels of visual quality degradation were realized in the Outdoor Factory Scene. The small values of the standard deviations in rendering time and SSIM demonstrate consistency in performance across the experiments conducted, as shown in Figure 5 below. Additionally, the tight confidence intervals indicate high reliability in the performance and visual-quality results achieved.

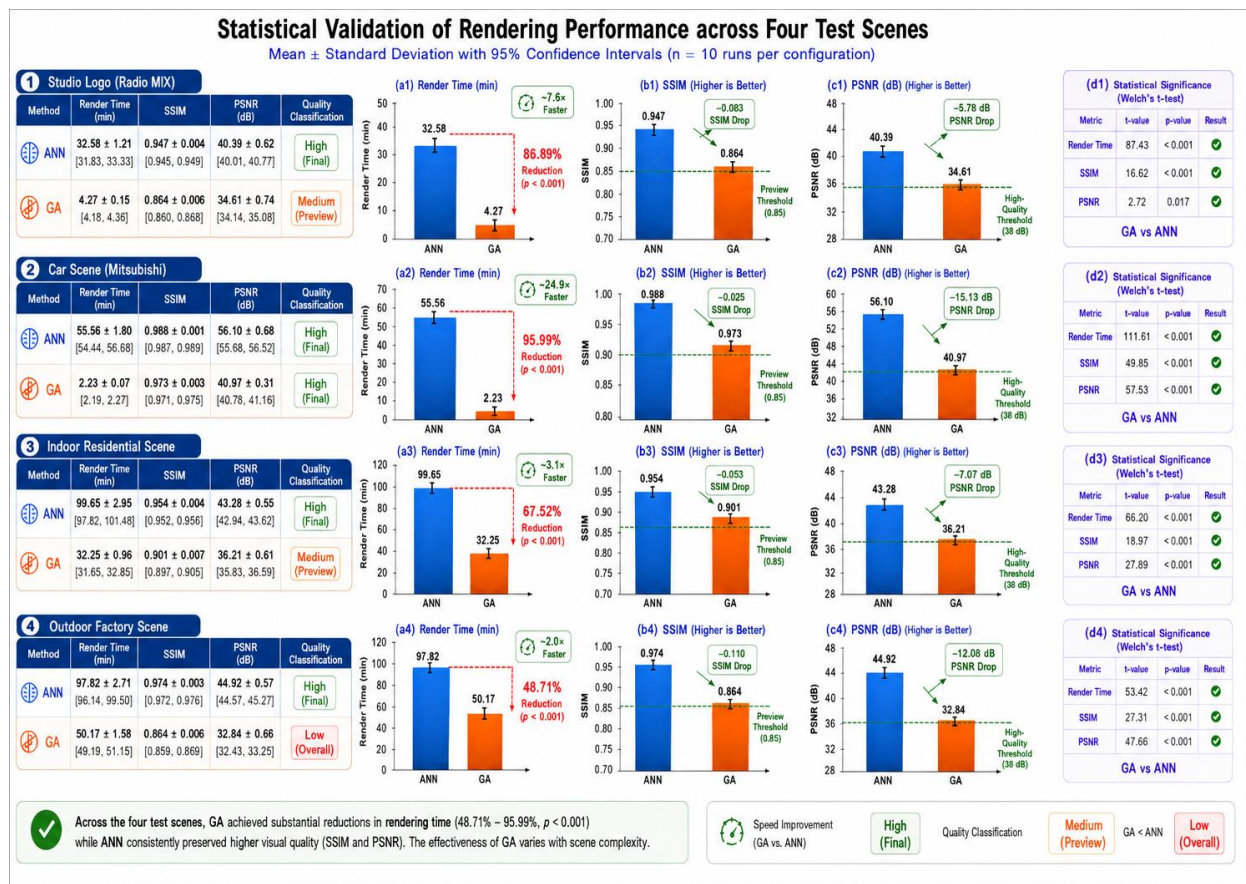


Figure 5. Statistical validation of ANN and GA rendering performance across the four test scenes.

5. Results and Discussion

The evaluation results of the developed hybrid GA-ANN approach are presented in this section for the following four 3D scenes: Studio Logo, Car, Indoor Residential, and Outdoor Factory. The reference, ANN-based, and GA-optimal models were compared based on rendering time and quality. The rendering time comparison is provided in Table 6, whereas MSE, PSNR, and SSIM, along with their quality classification, visual fidelity, and decision, are provided in Table 7. The evaluation criteria are presented in Table 8, and the optimal directions and thresholds used for quality classification into high, medium, and low are also provided. Table 9 shows the statistical analysis of rendering time and SSIM, where the average, standard deviation, and 95% confidence interval are presented.

Both ANN and GA were capable of considerably speeding up the process of rendering when compared to the base setups. ANN was able to preserve a high quality of visual representation in all four tested scenes, whereas GA results depended on specific features of the scenes. GA managed to preserve high quality in the Car Scene, provide medium quality suitable for preview rendering in the Studio Logo and Indoor Residential scenes, and achieve relatively poor visual results in the Outdoor Factory Scene.

5.1. ANN Performance Across the Four Test Scenes

The ANN method provided a significant drop in rendering time while delivering high-quality images for all four test scenes. In the case of the Studio Logo Scene, the reduction in rendering time was from 120 min to 32.58 min (a decrease of about 73%), and the values of MSE, PSNR, and SSIM were equal to 5.93, 40.39 dB, and 0.947, respectively. In the case of the Car Scene, the reduction in the rendering time was from 127 min to 55.56 min (an increase of about 56%), MSE = 0.1597, PSNR = 56.10 dB, and SSIM = 0.9983. In the case of the Indoor Residential Scene, rendering time was reduced from 314 min 35 s to 99

min 39 s, a reduction of about 68%, while achieving MSE = 4.1186, PSNR = 41.9834 dB, and SSIM = 0.9539. Similarly, the Outdoor Factory Scene was reduced from 231 min 13 s to 97 min 49 s, representing a reduction of about 58%, where MSE = 4.3134, PSNR = 41.7826 dB, and SSIM = 0.9737. This shows that the ANN selected an efficient configuration and provided good-quality renderings despite variations in the environment. The details of the rendering times and image quality are provided in Tables 6 and 7, respectively. In contrast to the ANN, the GA explores a wider rendering-parameter space to achieve greater acceleration, although the resulting visual quality is more dependent on scene characteristics.

5.2. GA Performance Across the Four Test Scenes

This evolutionary algorithm searches for the best rendering parameter set in the space of rendering parameters through an optimization process employing a fitness function based on short rendering times within a specified threshold for visual quality. The evolution of rendering parameter sets occurs through tournament selection, uniform crossover, and Gaussian mutation over multiple generations.

The GA-based approach achieved greater rendering-time reductions than the ANN, although its ability to preserve visual quality varied across the four scenes. In the Studio Logo Scene, rendering time decreased from 120 min to 4.27 min, representing a reduction of approximately 96%. However, the resulting MSE of 22.50, PSNR of 34.61 dB, and SSIM of 0.864 indicated that the image was of medium quality, suitable for preview rendering. In the Car Scene, the GA reduced rendering time from 127 min to 2.19 min, achieving a reduction of approximately 98% while preserving high image quality with MSE = 5.20, PSNR = 40.97 dB, and SSIM = 0.973. For the Indoor Residential Scene, rendering time decreased from 314 min 35 s to 32 min 15 s, corresponding to a reduction of approximately 90%. Its MSE of 10.1481, PSNR of 38.0669 dB, and SSIM of 0.9014 resulted in medium-quality output suitable for preview applications. In the case of the Outdoor Factory Scene, the rendering time decreased from 231 min 13 s to 50 min 10 s, representing about a 78% decrease in rendering time. But the MSE = 29.6738, PSNR = 33.4071 dB, and SSIM = 0.8639 indicate that the degradation is higher and the overall quality is classified as low. GA provides significant acceleration; however, its efficiency with respect to visual quality varies with geometry and other scene factors, as shown in Table 10.

Table 10. GA optimization performance across the four test scenes.

Scene	Time Reduction	MSE	PSNR (dB)	SSIM	Overall Quality
Studio Logo (Radio MIX)	96% ↓	22.5	34.61	0.864	Medium Quality (Preview)
Car Scene (Mitsubishi)	98% ↓	5.2	40.97	0.9734	High Quality
Indoor Residential Scene	90% ↓	10.1481	38.0669	0.9014	Medium Quality (Preview)
Outdoor Factory Scene	78% ↓	29.6738	33.4071	0.8639	Low Overall Quality

5.3. Ablation Study

To better understand the contribution of each component in the proposed hybrid framework, an ablation study in Table 11 was conducted by evaluating three configurations:

- Reference rendering: Default rendering parameters without optimization.
- ANN only: Rendering parameters predicted directly by the neural network without GA refinement.
- Hybrid ANN + GA: ANN prediction followed by GA optimization.

The objective of this experiment is to isolate the contribution of the ANN prediction stage and the GA optimization stage to render efficiency while maintaining acceptable image quality.

Table 11. Ablation study of the proposed framework.

Main Contribution	Image Quality	Rendering Time	Configuration
Baseline	Highest	Very High	Reference
Fast parameter prediction	High SSIM	Moderate reduction	ANN Only
Fine parameter optimization	Scene-dependent	Maximum reduction	ANN + GA

The ANN stage provides the largest improvement in rendering quality preservation while reducing rendering time by approximately 56–73% across the evaluated scenes. This demonstrates that the neural network successfully predicts near-optimal rendering parameters directly from scene descriptors. The addition of the GA further refines these predicted parameters through multi-objective optimization, reducing rendering time by up to 98% in some scenes. Although this additional optimization introduces slight quality degradation in highly complex scenes, it substantially improves rendering efficiency. These results indicate that both components contribute differently to the final performance:

- The ANN provides intelligent initialization and avoids expensive exhaustive parameter search.
- The GA performs local refinement around the ANN prediction, producing the highest rendering speedups.

Therefore, the overall improvement results from the complementary interaction between ANN prediction and GA optimization rather than either component alone.

5.4. Discussion

As shown by the overall results, the suggested hybrid GA-ANN approach enables the user to flexibly adjust the trade-off between rendering efficiency and visual quality for four analyzed scenes—Studio Logo, Car, Indoor Residential, and Outdoor Factory. Table 12 shows that the framework has three major operating modes. The first one is the reference mode, which ensures maximum visual quality but takes the most time. The second is the ANN-based mode, which gives a compromise and accelerates the process without sacrificing visual quality.

Table 12. Recommended rendering modes based on performance across the four test scenes.

Mode	Method	Relative Speed	Quality *	Recommended Use Case
High Quality	Reference	Slow	Best/Reference Quality	Final offline rendering when execution time is not a constraint
Balanced	ANN	Medium	Consistently High	Production rendering and quality-critical workflows
Accelerated	GA-Optimized	Fast to Very Fast	Scene-dependent: High, Medium, or Low	Rapid previewing, iterative design, and prototyping; final rendering only when the required quality thresholds are satisfied

* The GA quality was determined by the nature of the specific scenes. The GA provided high quality in the Car Scene; medium quality, suitable for preview rendering, in the Studio Logo and Indoor Residential scenes; and low overall quality in the Outdoor Factory Scene.

This approach, based on ANN, provided the best compromise between the quality of images and their rendering times and is therefore best suited for the production environment. This algorithm ensured that the visual quality of all four scenes was at its highest despite the differences in geometry, material types, lighting, and environment. On the other hand, GA sought to decrease the rendering time of scenes and managed to do that

considerably; however, it proved to be much more sensitive to the complexity of a scene compared to its impact on visual quality. The Car Scene was very tolerant of reductions in parameters while still retaining high visual quality, whereas the Studio Logo and Indoor Residential scenes were less tolerant of changes in indirect light and sampling parameters. The Outdoor Factory Scene was the least tolerant to GA optimization due to complex geometry, environmental light, depth, and shadows.

The reference configuration would still remain the best choice when the highest quality of an image was needed, and there were no time constraints. Overall, the proposed framework provides different renderings depending on the needs of the application and scene properties.

Threshold-Based Quality Classification

However, in this particular study, those quality thresholds were not considered as absolute values. They were chosen based on the analysis of the rendering output that was produced by means of systematic alteration of Arnold rendering parameters and comparison of the obtained images to the high-quality reference image using MSE, PSNR, and SSIM measures.

The thresholds shown in Table 8 were chosen based on the analysis of various combinations of potential boundaries in the rendering output images generated via the parameter sweeps, selecting those which had the greatest consistency in distinguishing between three levels of image quality: images ready for production, images ready for previewing, and images that needed more refinement. These thresholds were used directly as is in both ANN and GA evaluations in the four testing scenes in Table 7.

The three image quality measures were analyzed together since each measure represents a different perspective on the quality of rendering. MSE is a measure of the error at the pixel level, PSNR shows the fidelity of the signal, while SSIM gives the quality of perceptual and structural similarity. In order to avoid accepting an image with poor visual quality because a single measure was good, a strict decision rule was used in which the lower classification was chosen.

The SSIM-based quality categories were defined as follows:

- $SSIM > 0.93$: High quality, suitable for final rendering without further optimization.
- $0.85 \leq SSIM \leq 0.93$: Medium quality, suitable for preview rendering, iterative design, and development workflows.
- $SSIM < 0.85$: Low quality, unsuitable for final production and returned to the optimization stage for further refinement.

However, SSIM was not used independently to determine the final decision. The overall classification was determined by jointly evaluating MSE, PSNR, and SSIM according to the conservative decision rule.

The same decision thresholds were used uniformly across all four test images. As indicated in Table 7, all ANN models have been determined to be of high overall quality. The GA model received a high-quality rating for the Car Scene and a medium-quality rating suitable for preview rendering for the Studio Logo and Indoor Residential scenes. Despite the fact that the Outdoor Factory GA model provided medium-level PSNR and SSIM values, its overall quality was assessed as low due to the MSE exceeding the allowable limit. Thus, this model has been rejected from further production.

As a result, it can be seen that the proposed decision-making procedure provides a more conservative and reliable quality assessment than using SSIM or any other metric alone. The selected decision thresholds are specific only to the Arnold renderer, Full HD resolution, scene set, rendering conditions, and requirements used within this research. Thus, they cannot be considered universal for any other applications.

6. Conclusions

These outcomes clearly show the efficiency of the suggested hybrid GA-ANN approach, which allows us to save a lot of time during rendering without compromising visual quality. These tests were conducted using four 3D models characterized by different geometry, materials, lighting conditions, and environment—Studio Logo, Car, Indoor Residential, and Outdoor Factory.

The ANN-based algorithm was found to provide the best performance trade-off between the efficiency of rendering and the quality of images generated. The rendering time using this algorithm decreased by about 56–73% across all scenes without compromising image quality, with SSIM greater than 0.93 and PSNR greater than 40 dB for all images.

In the GA approach, the render speed-up was higher, with reductions in time between 78% and 98%. However, its quality was more dependent on the characteristics of the scenes. For the Car Scene, it provided high quality; for the Studio Logo and Indoor Residential scenes, it provided medium quality and could be used for preview rendering; in the Outdoor Factory Scene, it provided low quality because the MSE value exceeded the threshold. While all GA results had SSIM values above 0.85, their evaluation in terms of MSE and PSNR proved otherwise.

The threshold-based decision layer increased the framework's effectiveness by allowing it to classify the output as final, for preview, or needing refinement. On the whole, the proposed framework is quite effective in terms of rendering modes, where the ANN consistently produces high-quality output with significant time savings, while GA accelerates the process when appropriate quality conditions are satisfied.

In future work, attention will be paid to enhancing the fitness function of the GA by introducing more sophisticated perceptual metrics, increasing the diversity of scenes for training of the ANN, considering multi-objective optimization algorithms like NSGA-II, and carrying out wider comparisons among CPU and GPU environments.

Author Contributions: R.M. conceptualized the study, proposed the hybrid GA-ANN optimization framework, designed the experimental methodology, and performed the experimental evaluation and validation. A.H. contributed to the formulation of the optimization strategy, perceptual quality assessment metrics, and overall system architecture and integration. R.B. contributed to implementation support, scene feature and descriptor analysis, data processing, and interpretation of the experimental findings. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The original data presented in the study are openly available on the drive at [https://arabou-my.sharepoint.com/:f/g/personal/rafeek_aou_edu_eg/IgBp94psv8NoTI-Hxm-aYjgNAT1wmUGN3Ln8EoKq7MMMyufU?e=k1Qat1] (accessed on 8 June 2026).

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Tatarchuk, N.; Narkowicz, K.; Costa, T.; Lempiäinen, J.; Zhang, T.; Sousa, T. Advances in real-time rendering in games part II. In Proceedings of the Special Interest Group on Computer Graphics and Interactive Techniques Conference Courses, Vancouver, BC, Canada, 10–14 August 2025.
2. Furquez Herrera, I. From Rendering to Calculation: A V-Ray Workflow for the Lighting Analysis of Early Projects in Architecture. Bachelor's Thesis, Universitat Politècnica de Catalunya, Barcelona, Spain, 2025.
3. Aryal, A.; Giri, S.; Panday, S.P.; Sharma, S.; Dawadi, B.; Chalise, S. Efficient 3D Scene Reconstruction from Multi-View RGB Images Using Optimized Gaussian Splatting. *IEEE Access* **2025**, *14*, 1269–1286.
4. Yura, T.; Mirzaei, A.; Gilitschenski, I. Eventsplat: 3d gaussian splatting from moving event cameras for real-time rendering. In Proceedings of the Computer Vision and Pattern Recognition Conference, Nashville, TN, USA, 11–15 June 2025.

5. Mameli, M.; Paolanti, M.; Mancini, A.; Zingaretti, P.; Pierdicca, R. RenderGAN: Enhancing real-time rendering efficiency with deep learning. *ACM Trans. Multimed. Comput. Commun. Appl.* **2025**, *21*, 1–22.
6. Kant, G.; Sangwan, K.S. Predictive modelling and optimization of machining parameters to minimize surface roughness using artificial neural network coupled with genetic algorithm. *Procedia CIRP* **2015**, *31*, 453–458.
7. Li, H.; Yuan, D.; Ma, X.; Cui, D.; Cao, L. Genetic algorithm for the optimization of features and neural networks in ECG signals classification. *Sci. Rep.* **2017**, *7*, 41011.
8. Mori, H.; Toyama, F.; Shoji, K. Optimization of character gaze behavior animation using an interactive genetic algorithm. *Int. J. Asia Digit. Art Des.* **2017**, *21*, 25–31.
9. El-Baz, M.A.; Abd-Elwahed, M.S. Enhancing Product Quality Using Artificial Neural Networks and Genetic Algorithms. *J. Manuf. Mater. Process.* **2025**, *9*, 322.
10. Pribić, M.; Kamenko, I.; Despotović, S.; Mirosavljević, M.; Pejin, J. Modeling and optimization of triticale wort production using an artificial neural network and a genetic algorithm. *Foods* **2024**, *13*, 343.
11. Tran, D.T.; Pham, V.K.; Nguyen, A.T.; Nguyen, D.-T. Aerodynamic Design Optimization for Flying Wing Gliders Based on the Combination of Artificial Neural Networks and Genetic Algorithms. *Aerospace* **2025**, *12*, 818.
12. Benramdane, K.I.; Hadji, M.E.M.; Khodja, M.; Drouiche, N.; Grassl, B.; Lebouachera, S.E.I. Surfactants Adsorption onto Algerian Rock Reservoir for Enhanced Oil Recovery Applications: Prediction and Optimization Using Design of Experiments, Artificial Neural Networks, and Genetic Algorithm (GA). *Colloids Interfaces* **2025**, *9*, 19.
13. Vasiou, E.; Shkurko, K.; Mallett, I.; Brunvand, E.; Yuksel, C. A detailed study of ray tracing performance: Render time and energy cost. *Vis. Comput.* **2018**, *34*, 875–885.
14. Reynolds, D.T.; Laycock, S.D.; Day, A. Real-time accumulation of occlusion-based snow. *Vis. Comput.* **2015**, *31*, 689–700.
15. Purcell, T.J.; Buck, I.; Mark, W.R.; Hanrahan, P. Ray tracing on programmable graphics hardware. In *ACM SIGGRAPH 2005 Courses*; ACM: New York, NY, USA, 2005; p. 268-es.
16. Fender, J.; Rose, J. A high-speed ray tracing engine built on a field-programmable system. In *Proceedings of the 2003 IEEE International Conference on Field-Programmable Technology (FPT)*; (IEEE Cat. No. 03EX798); IEEE: New York, NY, USA, 2003.
17. Umenhoffer, T.; Szirmay-Kalos, L.; Szécsi, L.; Lengyel, Z.; Marinov, G. An image-based method for animated stroke rendering. *Vis. Comput.* **2018**, *34*, 817–827.
18. Lousada, P. Bandwidth and Memory Efficiency in Real-Time Ray Tracing. Master's Thesis, University of Lisbon: Lisbon, Portugal, 2017.
19. Feldmann, D. Accelerated ray tracing using R-trees. In *Proceedings of the International Conference on Computer Graphics Theory and Applications*; SCITEPRESS: Setúbal, Portugal, 2015.
20. Seibt, S.; Kuth, B.; von Rymon Lipinski, B.; Chang, T.; Latoschik, M.E. Multidimensional image morphing-fast image-based rendering of open 3D and VR environments. *Virtual Real. Intell. Hardw.* **2025**, *7*, 155–172.
21. Mai, A.; Hedman, P.; Kopanas, G.; Verbin, D.; Futschik, D.; Xu, Q.; Kuester, F.; Barron, J.T.; Zhang, Y. Ever: Exact volumetric ellipsoid rendering for real-time view synthesis. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, Honolulu, HI, USA, 19–23 October 2025.
22. Balloni, E.; Stacchio, L.; Mancini, A.; Frontoni, E.; Zingaretti, P.; Paolanti, M. A Neural Rendering system for fashion design process. *Eng. Appl. Artif. Intell.* **2025**, *153*, 110773.
23. Palma, B.; Casanova-Salas, P.; Gimeno, J.; Casas-Yrurzum, S. Hybrid visualisation of medical image: Surface and volume rendering. *Comput. Graph.* **2025**, *133*, 104453.
24. Zhang, J. Application analysis of special effects technology in film and television post-production. In *Proceedings of the International Conference on Frontier Computing*; Springer: Berlin/Heidelberg, Germany, 2020.
25. Fang, X.; Zhang, Y.; Tan, H.; Liu, C.; Yang, X. Performance evaluation and optimization of 3D Gaussian splatting in indoor scene generation and rendering. *ISPRS Int. J. Geo-Inf.* **2025**, *14*, 21.
26. Jiang, L.; Mao, Y.; Xu, L.; Lu, T.; Ren, K.; Jin, Y.; Xu, X.; Yu, M.; Pang, J.; Zhao, F. Anysplat: Feed-forward 3d gaussian splatting from unconstrained views. *ACM Trans. Graph. (TOG)* **2025**, *44*, 1–16.
27. Yin, Z.; Gadirov, H.; Kosinka, J.; Frey, S. ENTIRE: Learning-based volume rendering time prediction. *arXiv* **2025**, arXiv:2501.12119.
28. Xie, L.; Yuille, A. Genetic cnn. In *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, 22–29 October 2017.
29. Pham, V.H.S.; Nguyen, D.H.; Vo, T.B.H. Development of a hybrid model between generative adversarial networks (GANs) and Genetic Algorithms (GA) in apartment layout optimization. *Clust. Comput.* **2026**, *29*, 1.

30. Tan, B.; Moretti, F. Lightweight Real-Time Rendering Parameter Optimization via XGBoost-Driven Lookup Tables. *arXiv* **2026**, arXiv:2604.25178.
31. Zhang, G.; Bhagat, S.; Lee, S.H.; Kim, D.W.; Kim, D.J. From game to beyond game: Exploring the role of 3D rendering technology in user immersion and virtual consumption in the Metaverse. *Decis. Support Syst.* **2025**, *189*, 114383.
32. Yang, S. Adaptive Multi-Resolution Rendering for Virtual Reality Scenes: A Dynamic Resolution and Task Scheduling Approach. *Informatica* **2025**, *49*, 403–418.
33. Wang, Z.; Bovik, A.C.; Sheikh, H.R.; Simoncelli, E.P. Image quality assessment: From error visibility to structural similarity. *IEEE Trans. Image Process.* **2004**, *13*, 600–612.
34. Zhou, T.; Li, Y.; Chu, X.; Cao, C.; Zhou, C.; Yu, F.; Liu, Y. Real-time high-resolution view synthesis of complex scenes with explicit 3D visibility reasoning. *IEEE Trans. Vis. Comput. Graph.* **2024**, *31*, 6178–6189.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.