Proceedings of the 8th International Conference on Computing and Data Science
DOI: 10.54254/2755-2721/2026.36116

Review of Key Technologies and Application Scenarios of Neural Rendering

Zhuotong Liu

School of Management Science and Engineering, Southwestern University of Finance and Economics, Chengdu, China
1415664267@qq.com

Abstract. At prohibitive labor costs and low flexibility, physically based rendering is far from effective for the high-fidelity 3D content creation, thus in this paper, we attempt to give a survey of the area of neural rendering. We explain how deep neural networks learn implicit representation of scenes from visual data for efficient and photorealistic novel view synthesis. We review three common types of neural rendering methods: implicit representation (such as neural radiance fields (NeRF) and its subsequent architectures such as Instant-NGP), explicit representation (such as neural texture), and hybrids (such as NeuS and 3D Gaussian splatting). We explain their essence, linkages, advantages and disadvantages. For example, 3D Gaussian splatting achieves state-of-the-art results on the celebrated Blender dataset (33.55 dB PSNR, 0.967 SSIM) with real-time rendering 134 FPS, solving the efficiency limit of vanilla NeRF.Looking ahead, the combination of neural rendering with generative AI, large-scale 3D foundation models and lightweight architectures will lay a foundation for the emerging immersive internet, and digital twin landscape.

Keywords: neural rendering, NeRF, 3D Gaussian Splatting, implicit representation, view synthesis

1. Introduction

Conventional physically based rendering (PBR) pipelines can produce photorealistic images; but they require a complex amount of manual interventions (such as specifying a mesh geometry, texture and lighting) and intensive physics simulations, which is cost-ineffective and inflexible in production, often requiring substantial recalculation for different viewpoints or light setups. Also, PBR cannot model complex light transport phenomena accurately, such as subsurface scattering of skin or smoke-like effects in volumetric phenomena [1].

Neural rendering represents a paradigm shift from explicit physical modelling to data-driven inference. By learning deep neural networks to implicitly represent scene properties (geometry, materials and lighting) directly from image data, neural rendering renders new views implicitly and supports also a scene-based editing (e.g., changing camera pose or illumination) without manual parameter tuning [2]. Thus, it reduces the technical barrier towards high fidelity content generation and provides a scalable path to virtual reality and digital twin applications.

© 2026 The Authors. This is an open access article distributed under the terms of the Creative Commons Attribution License 4.0 (https://creativecommons.org/licenses/by/4.0/).

DOI: 10.54254/2755-2721/2026.36116

Efficiency first. Early work focused on improving efficiency. DeepVoxels [3] came out with a learned appearance representation based on voxel grids, solving novel view synthesis for simple objects. At the same time, the Scene Representation Network (SRNs) [4] proposed a continuous, class-conditional scene representation, which generalizes from sparse inputs, proving the feasibility of replacing parts of the traditional rendering with neural networks. The field hit its key milestone in 2020 with the advent of Neural Radiance Fields (NeRF) by Mildenhall et al. [5]. Using an MLP to form a 5D radiance field, NeRF synthesizes realistic novel views from sparse inputs through differentiable volume rendering, forming the"implicit representation + differentiable rendering"concept.

Subsequent works split into horizontal developments and vertical optimizations. Vertical advances traded off efficiency and robustness: Instant-NGP [6] trained from hours to seconds with multi-resolution hash encoding; Mip-NeRF [7] eliminated aliasing issues with a built-in positional encoding and conical frustum sampling. Horizontal extensions explored wider scenarios: D-NeRF [8] models non-rigid dynamics with a spatio-temporal deformation field, NeuS [9] utilizes a Signed Distance Function (SDF) based framework to recover a noise-free surface geometries. Most recently, 3D Gaussian Splatting [10] has appeared to be the state-of-the-art hybrid method. By representing the scene as a set of learnable anisotropic Gaussians and perform differentiable rasterization, 3D Gaussian Splatting is able to obtain a well-balanced visual quality/efficiency trade-off, rendering cinematic quality images in real time.

Although the story of static scene reconstruction, photorealism, and computational speed has made impressive advances, current neural rendering still suffers from persistent problems, such as the heavy computational overhead to modelling complex dynamics, the poor 3D consistency and generalization of generative methods, as well as the challenging problems of learning disentangled and controllable scene representations. Moreover, there is still a systematic analysis of the evolutionary development and inherent linkage among the three most popular technical paradigms: implicit, explicit and hybrid representations, missing. To fill this gap, in this paper, we offer a review of the latest advancements of neural rendering. We carry out a comprehensive discussion of the key technical pipelines, compare representative methods against different application scenarios, analyse some common bottlenecks. Lastly, we discuss several promising topics for future studies to encourage research innovations in the field.

2. Main contributions of this paper

Unlike existing surveys that lack a systematic taxonomy, we have established a unified taxonomy covering the three major"paradigms": implicit, explicit and hybrid representations. We decouple systematically, in great detail, the main mechanisms used by the most important landmark methods —NeRF, Instant-NGP, NeuS and 3D Gaussian Splatting. Most importantly, we provide the evolution narrative of these methods and disclose the intrinsic reasons of their different performance trade-offs. We also provide a multi-factor, comparative study of the visual aspects (subjective visual quality) and quantitative aspects (PSNR, SSIM, LPIPS, training speed, rendering speed), following a comprehensive protocol that allows for a strong qualitative and quantitative characterization of the merits, and of the intrinsic advantages and disadvantages in their favor between the three technical paradigms.

By reviewing four exemplary application scenes: virtual human, virtual production, AR/VR, and text-to-3D generation, we distilled four representative technical bottlenecks of technologies today: dynamic scene reconstruction, cross-application migration, compatibility of edge equipment,

DOI: 10.54254/2755-2721/2026.36116

disentangled controllability of attributes, etc.; and also proposed three emerging future trends: generative AI integration, 3D foundation models deployment and lean models.

3. Related work

3.1. Neural representations of core elements

Implicit representations of geometry: Unlike discrete meshes or point clouds, neural rendering often uses continuous implicit functions to describe geometry. For example, NeRF uses an MLP to learn the volume density σ at any point X=(x,y,z) in space. The density field indirectly defines the isosurface of the object [5]. Another idea is to learn the signed distance function (SDF) f(x), whose zero level set f(x)=0 is the surface, which can produce a clearer and noise-free geometry [9]. The SDF is defined as follows:

$$\mathscr{S} = \{\mathbf{x} \in \mathbb{R}^3 \mid f(\mathbf{x}) = 0\} \quad (1)$$

Where f(x)>0 means that the point is outside the surface, f(x)<0 means that the point is inside the surface, and |f(x)| is the distance to the surface.

Neural Encoding of Materials: Neural networks implicitly encode complex material properties (e.g., BRDF) by learning how image appearance changes with view d and illumination l. For example, neural texture stores material information in a high-dimensional feature map associated with an explicit mesh UV and decodes the final color through a small coloring network, achieving decoupling of appearance details from underlying geometry [2].

Reverse Illumination Inference: Neural rendering has the ability to reverse estimate and control illumination from an image. Common approaches include learning an illumination encoding that is decoupled from the scene description, or directly predicting spherical harmonic coefficients of ambient illumination, thus allowing scene illumination to be individually adjusted after reconstruction [2].

Necessity of camera model: Accurate camera parameters (intrinsic and extrinsic) are a prerequisite for neural rendering to be able to correctly establish 2D-3D correspondences. Most methods rely on preprocessing steps such as Motion Recovery Structure (SfM) to obtain the camera pose of the input image [5].

Positional encoding: In order to overcome the bias of MLP in fitting high-frequency details (spectral bias), NeRF et al. introduced high-frequency positional encoding to map low-dimensional input to high-dimensional space, so that MLP can easily learn the details of the scene [5].

3.2. Fundamentals of deep learning

Multilayer Perceptron (MLP) : As a universal function approximator for learning the mapping from coordinates to scene properties, its structure is simple and theoretically able to fit any continuous function, which is the cornerstone of implicit representation (such as NeRF) [5].

Convolutional Neural Networks (CNNS) : Good at extracting local features and structural priors in images. In the generalization approach, Convolutional Neural Network are used as image encoders to provide contextual information about geometry and appearance to subsequent MLPS, enabling them to generalize to new scenes from a small number of inputs [11].

Transformer: Its powerful global attention and long-range dependence modeling ability make it perform well when dealing with dynamic sequences (such as temporal information) [8] or large-

DOI: 10.54254/2755-2721/2026.36116

scale complex scenes, which provides the possibility to build general 3D large models.

Differentiable rendering: This is the core technique that enables neural rendering to achieve end-to-end optimization. It is mathematically refactored to make key steps in traditional rendering pipelines (e.g., ray-geometry intersection, shading calculation, rasterization) differentiable.

4. In-depth analysis of key technical paths of neural rendering

4.1. Implicit representation

4.1.1. NeRF method

NeRF represents a static scene as a continuous 5D neural radiance field function：

$$F_\Theta : (\mathbf{x}, \mathbf{d}) \rightarrow (\mathbf{c}, \sigma) \quad (2)$$

Where X=(x,y,z) is the spatial position, d=(θ,ϕ) is the viewing direction, and the output is the view-dependent RGB color c and volume density σ.

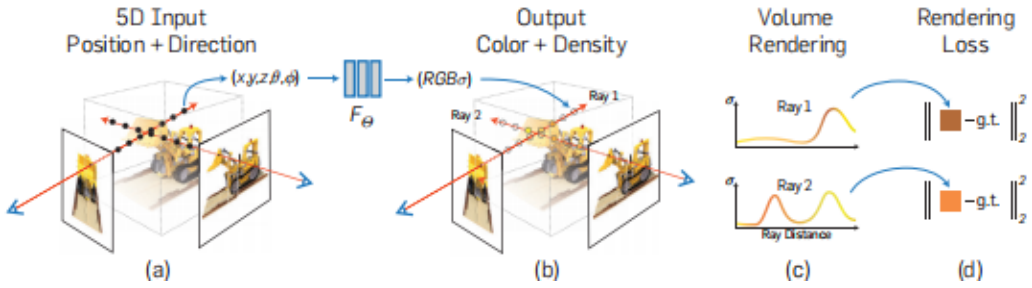


Figure 1. Flowchart of NeRF

The complete workflow is shown in Figure 1. First, a known camera pose and multi-view image are input, and for each pixel in the target view, the system shoots a ray r(t) from its corresponding camera center. Subsequently, a sequence of discrete 3D points {xi} is obtained by stratified sampling along this ray within a set depth range $[t_n,t_f]$. Then, the high-frequency position encoding function γ(·) was used to map the spatial coordinates xi and light direction d of these sampling points in high dimension, so as to improve the network's ability to perceive high-frequency details. The encoded feature vectors are fed into a shared MLP network, which predicts the volume density σi based on the position information and predicts the view-dependent RGB color ci in combination with the orientation information. After that, the system uses the discrete volume rendering formula to calculate the final predicted color C^(r) for that pixel based on the properties of sampled points along the ray (σi,ci), where the weight Tiαi is designed to ensure that only points located near the visible surface contribute the most to the final color. Finally, the Mean Square Error (MSE) loss between the rendered color C^(r) and the corresponding pixel color Cgt in the real image was calculated, and the network parameter Θ of the MLP was optimized by the back propagation algorithm, so as to obtain a neural radiance field model capable of generating photo-realistic images from any new viewpoint.

Input and encoding: For a given camera pose, a ray is emitted to each pixel. A sequence of 3D points {xi} is obtained by stratified sampling in the depth range $[t_n,t_f]$ along the ray. The coordinates xi and light direction d of these points are transformed by the above high-frequency

DOI: 10.54254/2755-2721/2026.36116

position encoding $\gamma(.)$, respectively, to improve the learning ability of MLP for high-frequency details.

$$\gamma(p) = (\sin(2^0\pi p), \cos(2^0\pi p), \ldots, \sin(2^{L-1}\pi p), \cos(2^{L-1}\pi p)) \tag{3}$$

Where p represents a one-dimensional scalar input (usually a coordinate or angular component); L denotes the highest frequency level of positional encoding (controlling the number of high-frequency components). $2^l\pi$ (l=0,... ,L−1) denotes the angular frequency of the l-th level sine and cosine function; Let γ(p) denote the output vector (of dimension 2L) formed by concatenating alternating sines and cosines.

Neural Field Query: The encoded position γ(xi) and orientation γ(d) are fed into a shared MLP network. The network first predicts the density σi with γ(xi), and then combines σi and the encoded orientation γ(d) to predict the color ci.

$$\gamma(x)=[\gamma(x),\gamma(y),\gamma(z)], \quad \gamma(d)=[\gamma(d_x),\gamma(d_y),\gamma(d_z)] \tag{4}$$

Where, x=(x,y,z) represents the three-dimensional space position vector, whose components x,y,z respectively represent the coordinates of the space point on the three orthogonal axes. d=(dx,dy,dz) denotes the observation direction vector, whose components dx,dy,dz represent the components of the direction vector along the three axes; Let $\gamma(\cdot)$ denote the position-encoding function defined in Equation (3); γ(x) and γ(d) denote the feature vectors formed by concatenating the components of the spatial position vector and the viewing direction vector independently encoded, respectively.

Differentiable split volume rendering: The discrete volume rendering formula is used to synthesize (σ_i,c_i) of all sampled points on this ray into the final color $\widehat{C}(r)$ of this pixel. The weight $T_i\alpha_i$ naturally makes samples close to the surface (high density) with little front occlusion (high transmittance) contribute the most to the final color.

$$C(r)=\int_{t_n}^{t_f} T(t)\sigma(r(t))c(r(t),d)dt, \quad \text{where } T(t)=\exp(-\int_{t_n}^{t}\sigma(r(s))ds) \tag{5}$$

Where r(t)=o+td represents a camera ray, where o is the camera optical center (origin) and t is the depth parameter along the ray direction. $[t_n,t_f]$ denotes the proximal and distal depth range of ray sampling; $\widehat{C}(r)$ is the predicted RGB color of the pixel for that ray; Let σ(r(t) denote the volume density of the ray at depth t; c(r(t),d) denotes the observed color of the light at depth t along direction d; T(t) is the cumulative transmittance of a ray from the near plane tn to depth t (that is, the probability that a ray reaches t without colliting with a particle); $\exp(\cdot)$ is the exponential function.

4.1.2. Instant-NGP method

Unlike NeRF methods, the revolutionary aspect of Instant-NGP is the introduction of multi-resolution hash coding.

DOI: 10.54254/2755-2721/2026.36116

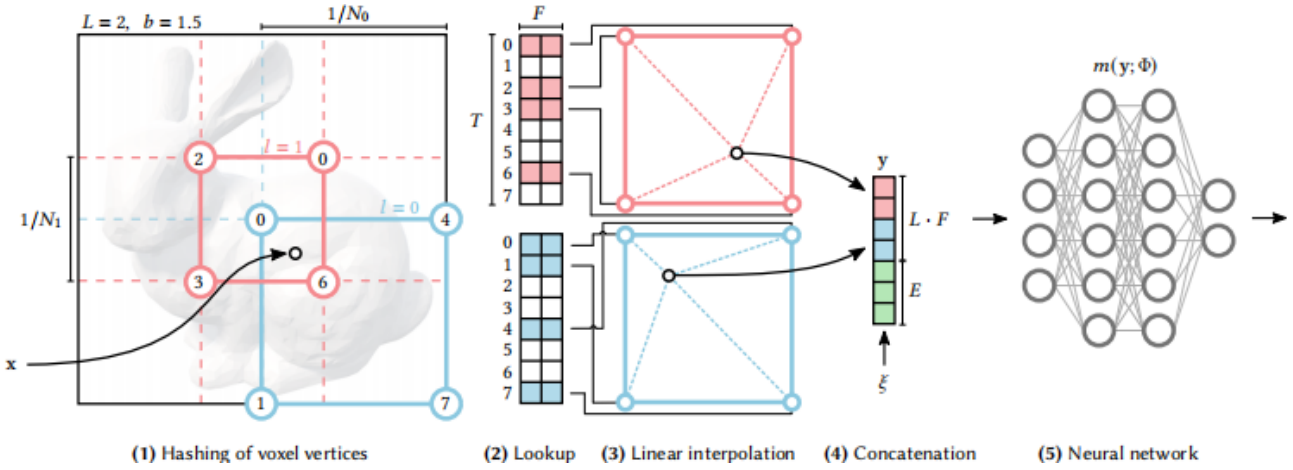


Figure 2. Hash encoding module diagram

Its complete workflow is shown in FIG. 2, first the system maps continuous input vertex coordinates into discrete hash bucket indices using a hash function according to the input hash table size (l,b) and scaling factor (1/Na,1/Nb). Next, a Lockup operation is performed to extract the corresponding feature vector T from the pre-allocated hash table based on the generated index. Subsequently, due to the low resolution of the features in the hash table, a Linear interpolation operation is used to smoothly transition the extracted discrete features based on the exact continuous positions of the vertices in the grid to obtain the continuous interpolated features F. After that, the interpolated feature F is concatenated with other attributes of the input (such as normal, color, etc.) to form a joint feature vector of higher dimensions. Finally, this joint feature vector is fed into a multi-layer perceptron (Neural network) for nonlinear regression to finally output the desired prediction result m(y;θ), such as displacement, color, or other geometric properties of the vertices.

The spatial domain is partitioned into Ldistinct resolution levels. For a given 3D query point, the algorithm identifies the enclosing voxel at each resolution level. The eight vertices of this voxel are mapped via a hash function to a compact, learnable feature vector table for efficient indexing. Features retrieved through this hash-based indexing undergo trilinear interpolation to compute the feature embedding specific to that resolution. Subsequently, the interpolated features across all Llevels are concatenated and fed into a shallow MLP (typically comprising only 1–2 layers). By offloading the primary representational burden from the MLP to these efficient hash-table lookups and interpolation operations, this architecture achieves a three-order-of-magnitude acceleration in training throughput compared to vanilla NeRF.

4.2. Explicit representation

4.2.1. Neural texture rendering

The core idea of Neural texturing is to extend the traditional 2D RGB texture Map into a high-dimensional Neural Feature Map, associated with an explicit (often less accurate) 3D mesh model [2].

The rendering flow is as follows:

DOI: 10.54254/2755-2721/2026.36116

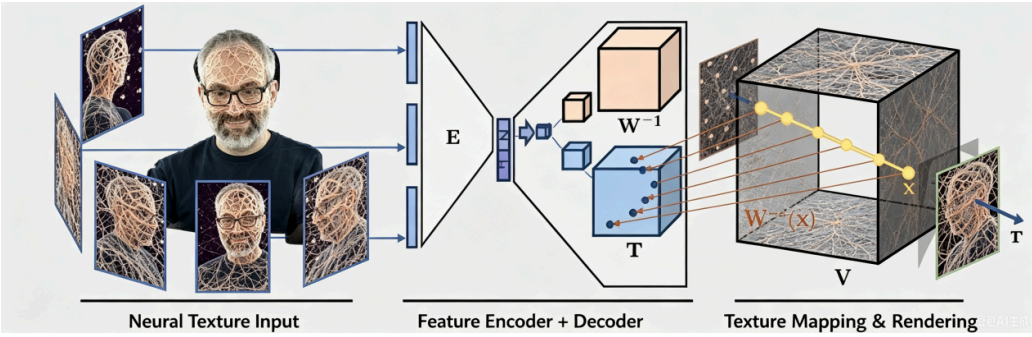


Figure 3. Flow chart of neural texture rendering

This figure shows the flow of a neural texture-based view synthesis method: first, a neural texture image from multiple views is input, then its 3D implicit representation is learned by a feature encoder-decoder network, and finally a new image from the target view is generated by ray sampling and texture mapping rendering.

Feature Mapping: Given a triangular mesh model and its UV parameterization, create a neural feature map corresponding to it. Instead of storing RGB values, each pixel (or grain) of this feature map stores a k-dimensional feature vector (K is typically 16, 32, and so on).

Rasterization and Feature query: For each camera ray, its intersection with the grid is determined by differentiable rasterization. According to the UV coordinates (u,v) of the intersection points, the corresponding feature vector $f \in R^K$ is obtained by bilinear interpolation from the neural feature map and query.

Neural coloring: The queried feature vector f, the surface normal at the intersection n, the viewing direction d, and possibly the illumination information l are fed into a lightweight MLP (called a coloring network).

Color decoding: The coloring network outputs the final RGB color c of this intersection. The whole "feature query-coloring" process is differentiable [12].

4.3. The hybrid representation technology path

While implicit methods dominated early research, their computational overhead spurred interest in explicit data structures. A significant deviation from the MLP-centric paradigm came with Plenoxels [13]. By discretizing the scene into a sparse voxel grid and optimizing spherical harmonic coefficients directly via gradient descent, Plenoxels demonstrated that high-quality view synthesis was feasible without deep neural networks, albeit at the cost of increased memory consumption. This insight laid groundwork for subsequent innovations balancing speed and fidelity. NeuS refines this balance for surface reconstruction using Signed Distance Functions, whereas 3D Gaussian Splatting [10] further disrupts the status quo by replacing voxels with anisotropic Gaussians.

4.3.1. NeuS: high-quality surface reconstruction with SDFS

NeuS aims to perform high-quality geometric surface reconstruction from multi-view images, using an MLP that learns the signed distance function (SDF) $f(x)$ and color function $c(x,d)$ of the scene, rather than just novel view synthesis.

NeuS designs a novel, SDF-derived weighting function $\phi_s(f(x))$ where s is a learnable parameter. The function is constructed to take a peak at the zero level set of the SDF, that is, the surface。

Specifically, NeuS defines a logical density distribution based on SDF

DOI: 10.54254/2755-2721/2026.36116

$$\rho(t) = \phi_s(f(\mathbf{P}(t))) \cdot |\frac{d}{dt} f(\mathbf{P}(t))| \quad (6)$$

Where $\phi_s(f(x))=\frac{s\cdot e^{-sf(x)}}{(1+e^{-sf(x)})^2}$ is the derivative of the Sigmoid function. This design ensures that the color contributions mainly come from a narrow region near the surface during the volume rendering integration, so that the rendered colors accurately reflect the surface properties, and the accurate surface can be extracted by the cumulative distribution function of the weights of the volume rendering formula.

4.3.2. 3D Gaussian Splatting (3DGS) : rendering radiance fields in real-time

3DGS enables neural rendering with both movie-quality and real-time rendering speed, where scenes are explicitly represented as a collection of millions of learnable 3D Gaussian ellipsoids. Each Gaussian is defined by the following properties: the center position (mean $\mu\in R^3$), the 3D covariance matrix $\Sigma\in R^{3\times3}$ that determines its shape, orientation, and scale, the opacity $\alpha\in[0,1]$, and the set of spherical harmonic (SH) coefficients used to express view-dependent colors. The covariance matrix is constructed by a scaling matrix S and a rotation matrix R: $\Sigma=RSS^TR^T$.

Rendering flow - Differentiable Splatting：



Figure 4. Flow chart of real-time radiance field rendering

This figure shows a complete real-time rendering pipeline of 3D Gaussian Splatting. The core process begins with scene initialization, which uses Structure from Motion (SfM) technology to generate an initial sparse point cloud from multi-view images. Then, in the 3D Gaussian sputtering stage, these points are expanded and refined into a dense set of 3D Gaussian ellipsoids by a differentiable optimization process, which can finely characterize the geometry and appearance of the scene. The next most critical step is differentiable rendering, which abandons the traditional time-consuming ray-by-ray volume rendering mode and uses tile-based parallel rasterization technology to directly project 3D Gaussian and perform fast Alpha blending. Finally, the real-time rendering results with high quality are output efficiently

Initialization: Starting from the sparse point cloud generated by the Motion Recovery Structure (SfM), a 3D Gaussian is initialized for each point.

Projection: At rendering time, each 3D Gaussian is projected to the 2D image plane by a differentiable operation according to the camera projection model to obtain a 2D Gaussian. Its 2D covariance matrix Σ' is derived from the 3D covariance matrix and the Jacobian matrix J of the view transformation: $\Sigma'=JW\Sigma W^TJ^T$, where W is the view transformation matrix.

Tiled rasterization: Dividing the screen into multiple tiles (e.g. 16x16 pixels). Quickly assign each 2D Gaussian to the tile it covers.

Hybrid rendering: Within each tile, all Gausses covering the current pixel are sorted by depth, and then alpha blending is performed from front to back to calculate the pixel color

DOI: 10.54254/2755-2721/2026.36116

$C=\sum_{i\in N} c_i\alpha_i \prod_{j=1}^{i-1}(1-\alpha_j)$, where c_i is calculated from the spherical harmonic coefficient of the Gaussian and the viewing direction. This process is similar to conventional rasterization of transparent objects, but fully differentiable.

Adaptive Optimization: During training, 3DGS periodically clones (for under-covered areas) or splits (for over-stretched Gaussians) Gaussians and culls those with low opacity, thus adaptively controlling the number and density of Gaussians to better fit the scene geometry.

5. Results analysis and application scenarios of neural rendering

5.1. Evaluation index

SNR (Peak signal-to-Noise Ratio) : From a pixel-level perspective, the reconstruction accuracy is measured by calculating the mean square error (MSE) between the corresponding pixels of the generated image and the real image. A higher PSNR value indicates that the model has a smaller fitting error in pixel intensities, which reflects the basic ability to restore the geometric structure and texture details of the scene.

SSIM (Structural similarity) : Focusing on structure-level perception, SSIM comprehensively evaluates the similarity of two images from three dimensions: brightness, contrast and structure. Compared with PSNR, SSIM is more consistent with the sensitivity of human visual system to structural information, and can effectively reflect the authenticity of rendering results in structural features such as edge sharpness and texture coherence.

LPIPS (Learning Perceptual Image Patch Similarity) : stands for perception-level evaluation, which utilizes pre-trained deep learning models (such as AlexNet or VGG) to extract image features and calculate distances in the feature space. This metric is highly correlated with human subjective visual judgment, and can acutely capture perceptual defects (such as blur and artifacts) that are easily ignored by PSNR and SSIM, which is a key indicator to evaluate the visual realism of generated images.

Training time: refers to the total time it takes for the model to go from input data (e.g., multi-view images) to convergence and finish learning the scene representation. This metric is directly related to the feasibility of the method in data iteration, prototype verification, and large-scale production environment.

Rendering speed (FPS) : The rate at which a model can generate a single new view of an image given the camera pose. For real-time interactive applications such as virtual reality (VR), augmented reality (AR), digital twins, and online games, it is critical to maintain a high frame rate (typically ≥30 FPS) to enable smooth, delay-free, and immersive experiences for users.

DOI: 10.54254/2755-2721/2026.36116

5.2. Comparative analysis of results

5.2.1. Subjective analysis

Table 1. Depth comparison of main technical routes for neural rendering

Technical route	Representative method	Core representation	Rendering mode	Advantages	Limitations
Implicit representation	NeRF [5]	5D Radiance Field (MLP)	Differentiable volume rendering	The rendering quality is extremely high with rich details	Rendering is slow and limited to static scenes
	Instant-NGP [6]	Radiation field with hash encoding	Differentiable volume rendering	Training/rendering is extremely fast	Hash collisions may result in loss of detail
Explicit representation	Neural texture [12]	Grid + Neural feature map	Differentiable Surface rendering	Rendering speed is the fastest, suitable for real-time applications	Dependent on initial geometry
Hybrid representation	3DGS [10]	A collection of 3D Gaussian spheres	Differentiable rasterization	High quality + real-time rendering, fast training	High storage overhead

As shown in Table 1, this paper qualitatively compares the representative methods of the three technology paths from the dimensions of core representation, rendering mode, advantages and limitations. In general, methods based on implicit representations (exemplified by the NeRF family) are benchmark in rendering quality but often limited by speed; The methods based on explicit representation (such as neural texture) have the highest rendering efficiency, but they depend on the initial geometry and the appearance flexibility is limited by the texture resolution. For example, 3DGS achieves the best balance between efficiency and quality, but the explicit storage brings a large memory overhead.

In order to compare the performance of each method more objectively and quantitatively, this paper collates the performance of each method on the Blender dataset as shown in Table 2, which contains eight object scenes with complex geometry and non-Lambertian appearance.

5.2.2. Objective analysis

Table 2. Performance comparison of representative methods on Blender synthetic datasets

Methods	Path it belongs to	PSNR(dB)	SSIM(Scalar)	LPIPS(Scalar)	Training time (h)	Rendering speed (FPS)
NeRF [5]	Implicit representation	31.01	0.947	0.081	30	0.1

DOI: 10.54254/2755-2721/2026.36116

Table 2. (continued)

Mip-NeRF [7]	Implicit representation	33.09	0.961	0.043	30	0.11
Instant-NGP [6]	Implicit representation	33.18	0.963	0.047	0.1	10-15
Plenoxels [13]	Explicit/mixed	31.71	0.958	0.049	0.2	15.1
3D Gaussian Splatting (3DGS) [10]	Hybrid representation	33.55	0.967	0.041	0.6	134

Table II shows that 3DGS performs the best in terms of rendering quality. In terms of efficiency, Instant-NGP reduces the training time from "days" to "minutes", and 3DGS opens the door to interactive applications with real-time rendering speed of 134 FPS (over 1000 times higher than NeRF). From the perspective of evolution path, 3DGS with hybrid representation has become the most attractive direction with the best balance of quality, training and rendering speed, while the implicit method based on hash coding (Instant-NGP) maintains unique advantages with extreme training efficiency and good quality, which indicates that neural rendering is accelerating towards the direction of both high fidelity and efficient and practical.

5.3. Application scenarios

5.3.1. Virtual digital humans: from reconstruction to actuation

High-fidelity reconstruction: Reconstruct high precision 3D model and dynamic texture of specific people from multi-cam arrays or monocular video. For example, technology behind Meta Human project is based on the idea of neural rendering and can quickly and accurately generate very realistic digital faces [14]. Real-time animation & actuation: Latency constraints for low-latency interaction. Typical technical solutions are neural texture or lightweight NeRF. For example, combining a parametric face model (such as 3DMM) with neural textures to render high fidelity facial animation in real time by driving a limited number of expression parameters [15]. These applications are convenient for virtual anchors, virtual avatar for video conferencing and so on.

5.3.2. Video and game production: virtual production and special effects

Virtual production: such as the LED Volume technology used in the Mandalorian. The challenge is in producing good quality backgrounds on-the-fly, in synchronization with the viewpoint and position of the actor. Neural rendering, specifically the very optimized real-time NeRF variant or 3DGS can pre-learn a scene appearance and render the correct parallax and lighting effects in real time while shooting, substituting a green screen and post-synthesis, getting "what you see is what you get" [14]. Effects Replacement & Scene Expansion: Neural rendering can reconstruct scenes from old footage or short footage for a shot, enabling easy prop replacements, add effect and expand a shot without having to reconstruct an expensive set.

DOI: 10.54254/2755-2721/2026.36116

5.3.3. AR/VR: virtual-real fusion and immersive interaction

Central to augmented reality (AR) is the natural fusion of virtual and real world content. Neural rendering tackles this problem by inferring ambient lighting from monocular images in real-time (typically predicting the Spherical Harmonics (SH) coefficients). The estimated ambient lighting is then applied to render synthetic content and the cast shadows and specular highlights of the virtual objects are rendered with perfect photometric accuracy relative to the physical world [12, 16].

Beyond 360° Video: 6DOF light field synthesis: Neural rendering overcomes a basic limitation of conventional 360° video - 3DOF are available - by allowing 6DOF light field video synthesis5 [17]. This enables the user wearing a VR headset to do natural translational head movement (that is, egocentric translation). Thus, the visual system gets accurate binocular disparity cues and motion parallax cues, which are crucial physiologically for depth perception. This renders a deep 'perceived presence' feeling, providing an excellent immersive experience that is dramatically better than other panoramic media both ecologically and in comfort for the user.

5.3.4. 3D content creation and AIGC

Currently, neural rendering is mainly getting synergized with generative AI in particular denoising diffusion probabilistic models—and this combination is without doubt the most active and rapidly evolving research frontier. This synergy is revolutionizing the 3D content creation process, leading from manual labour-intensive authoring towards fully automatic data-driven synthesis. Text-/Image-to-3D Generation. A key advance here is DreamFusion [18], which introduces score distillation sampling (SDS). Bearing in mind the lack of large-scale 3D training datasets, DreamFusion sidesteps explicit 3D supervision altogether by optimizing a NeRF representation so that its multi-view renderings match a given textual description. Crucially, this optimization is constrained by a 2D diffusion model—in DreamFusion's case, Imagen—that is used as a parametric prior to ensure photorealism and fidelity to semantics, and thus"distilling"2D generative knowledge to 3D geometry without 3D supervision.

3D Content Editing. Recent developments allow editing neural fields with ease through simple interfaces like free-form sketching or with simple language instructions. One example capability is semantic inpainting: given a user-supplied edit in a single reference view, the encoding of a neural field automatically propagates this edit to all views. This propagation then promotes consistency between all views thanks to the implicit geometrical and photometric priors embodied by the neural network, preserving the structure of the whole scene [19].

6. Summary and prospect

Neural rendering now stands on a veritable peak of accelerating breakthroughs. Its fundamental idea —move from explicit physical modelling to data-driven inference—is radically changing computer graphics and the digital media industry, as a whole. From the conceptual milestone of NeRF to the efficiency milestone of Instant-NGP and 3D Gaussian Splatting, and to many recent breakout synergies with generative approaches such as DreamFusion, the field experience an unprecedented, innovative pace. Current research directions feature interest in high real-time rates and intelligent creation coexisting with high photorealistic quality. Despite challenges in dynamic scene modelling and cross-domain generalization, the ongoing interactions with AIGC, the research in 3D foundation models and progress towards lightweight networks herald a bright future. As such, neural rendering is poised to be an essential key building block for the next generation of immersive scenarios,

DOI: 10.54254/2755-2721/2026.36116

including the omniverse and digital twins, continuously unveiling considerable application potentials and commercial opportunities.

There are also challenges of dynamic scene modeling with non-rigid motion, and motion blur and artifacts can easily arise. Most of existing methods are designed for certain scene reconstruction, lack general generality, and require re-training across categories. High fidelity rendering requires high performance Gpus, mobile deployment has limitations on computing capability and memory. Internal entanglement of neural field makes it difficult to accurately separate and control independent attributes such as material and geometry. In future, neural rendering will involve generative AI such as diffusion model deeply to improve controllability and efficiency of 3D generation. Neural rendering will draw lessons from NLP and 2D vision experience to construct large 3D pre-trained large models, learn general priors from massive data to realize few shot or zero shot rendering [11, 20]. Through model compression, knowledge distillation and dedicated hardware acceleration, it can run in real time on edge devices such as mobile phones and XR glasses. At the same time, physical laws and high-level semantics are integrated into the model, so that the generated content has both visual realism and commonsense behavior logic, which is crucial for applications such as digital twin and scientific simulation.

References

[1] Pharr, M., Jakob, W., & Humphreys, G. (2023). *Physically based rendering: From theory to implementation*. MIT Press.

[2] Tewari, A., Fried, O., Thies, J., et al. (2020). State of the art on neural rendering. *Computer Graphics Forum, 39*(2), 701-727.

[3] Sitzmann, V., Thies, J., Heide, F., et al. (2019). DeepVoxels: Learning persistent 3D feature embeddings. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2437-2446.

[4] Sitzmann, V., Zollhofer, M., & Wetzstein, G. (2019). Scene representation networks: Continuous 3D-structure-aware neural scene representations. *Advances in Neural Information Processing Systems, 32*.

[5] Mildenhall, B., Srinivasan, P. P., Tancik, M., et al. (2021). NeRF: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM, 65*(1), 99-106.

[6] Müller, T., Evans, A., Schied, C., et al. (2022). Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics, 41*(4), 1–15.

[7] Barron, J. T., Mildenhall, B., Tancik, M., et al. (2021). Mip-NeRF: A multiscale representation for anti-aliasing neural radiance fields. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 5855–5864.

[8] Pumarola, A., Corona, E., Pons-Moll, G., et al. (2021). D-NeRF: Neural radiance fields for dynamic scenes. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 10318–10327.

[9] Wang, P., Liu, L., Liu, Y., et al. (2021). NeuS: Learning neural implicit surfaces by volume rendering for multi-view reconstruction. *Advances in Neural Information Processing Systems, 34*, 27171–27183.

[10] Kerbl, B., Kopanas, G., Leimkühler, T., et al. (2023). 3D Gaussian splatting for real-time radiance field rendering. *ACM Transactions on Graphics, 42*(4), 139:1–139:14.

[11] Yu, A., Ye, V., Tancik, M., et al. (2021). PixelNeRF: Neural radiance fields from one or few images. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 4578–4587.

[12] Thies, J., Zollhofer, M., Nießner, M. (2019). Deferred neural rendering: Real-time neural rendering with neural textures. *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 7638–7647.

[13] Fridovich-Keil, S., Yu, A., Tancik, M., et al. (2022). Plenoxels: Radiance fields without neural networks. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5501–5510.

[14] Fang, Z., Cai, L., Wang, G. (2021). MetaHuman Creator: The starting point of the metaverse. *2021 International Symposium on Computer Technology and Information Science (ISCTIS)*, 154–157.

[15] Gao, H., Xu, J., Su, H., et al. (2021). Dynamic neural radiance fields for monocular 4D facial avatar reconstruction. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8649–8658.

[16] Srinivasan, P. P., Mildenhall, B., Tancik, M., et al. (2020). Lighthouse: Predicting lighting volumes for spatially-coherent illumination. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 8080-8089.

DOI: 10.54254/2755-2721/2026.36116

[17] Overbeck, R. S., Erickson, D., Evangelakos, D., et al. (2018). A system for acquiring, processing, and rendering panoramic light field stills for virtual reality. *ACM Transactions on Graphics, 37*(6), 1-15.
[18] Poole, B., Jain, A., Barron, J. T., et al. (2022). DreamFusion: Text-to-3D using 2D diffusion. *arXiv preprint arXiv:2209.14988*.
[19] Wang, C., Chai, M., He, M., et al. (2022). CLIP-NeRF: Text-and-image driven manipulation of neural radiance fields. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 3835-3844.
[20] Mirzaei, A., Aumentado-Armstrong, T., Derpanis, K. G., et al. (2022). Spin-NeRF: Multiview segmentation and perceptual inpainting with neural radiance fields. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 16501-16520.