Proceedings of CONF-SEML 2026 Symposium: Computational Analysis and Modeling in Complex Intelligent Systems
DOI: 10.54254/2755-2721/2026.GU34231

Deep Learning Applications in Real-Time Rendering for Games

Weihao Dong

Brunel College, North China University of Technology, Beijing, China
2379523@brunel.ac.uk

Abstract: In recent years, deep learning has been increasingly incorporated into practical rendering pipelines of computer graphics, achieving favorable results in denoising, super-resolution, material reconstruction and light estimation. Rather than completely replacing conventional rendering pipelines, neural networks are commonly adopted as auxiliary modules in real-time rendering to compensate for the deficiencies of traditional methods in efficiency and visual quality. This study summarizes mainstream applications of deep learning in game-oriented real-time rendering, and explores the correlation between neural rendering schemes and classic rendering techniques under practical game development scenarios. It first outlines fundamental traits of game real-time rendering, highlighting the critical necessity to strike a balance among visual fidelity, frame consistency and computational efficiency. Common application branches are then sorted out, covering super-resolution reconstruction, image denoising, light estimation and material simulation, alongside classic algorithms and technical properties. Taking water scene rendering as a typical case, this paper verifies that deep learning mainly solves partial rendering challenges rather than replacing the whole rendering architecture. A comparative analysis is conducted between neural-based and traditional real-time rendering methods, evaluating their strengths, drawbacks and applicable scenarios. The final conclusion indicates that deep learning currently serves as an effective auxiliary optimization tool for game rendering. The long-term advancement of this domain relies on continuous upgrades in running performance, frame stability, operational controllability and compatibility with commercial game production pipelines.

Keywords: deep learning, real-time rendering, game graphics, super-resolution

1. Introduction

With the continuous growth of the game industry and the increasing demand for visual quality, interactivity, and immersion, real-time rendering has become one of the most fundamental graphic technologies in modern game development, directly shaping the final visual presentation of scenes, characters, materials, and lighting effects on screen. At the same time, modern game rendering has to satisfy a strict real-time budget while pursuing higher realism, finer detail, and stronger temporal stability. Consequently, the trade-off between image quality and rendering performance has remained a central issue in this field.

© 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.GU34231

In recent years, deep learning has begun to influence computer graphics and rendering more directly, especially in tasks that are closely connected with game rendering. Research has already covered neural super-resolution and frame reconstruction, denoising for Monte Carlo and ray-traced images, neural global illumination and radiance caching, learned appearance and shading models, as well as several scene-level visual effects. Taken together, these works show that deep learning is no longer simply an external post-processing tool. Instead, it is increasingly being integrated into different parts of the rendering pipeline, either to improve image reconstruction, reduce sampling cost, or approximate computations that are too expensive for real-time use.

At the same time, the existing literature is still relatively dispersed. Related studies appear in computer graphics, computer vision, neural rendering, and industrial rendering systems, and they often use different goals, evaluation standards, and assumptions about deployment. As a result, strong experimental performance in a paper does not always mean that a method is immediately useful for games. In practice, its value also depends on frame-time cost, temporal stability, robustness across different assets and scenes, compatibility with current engines, and the degree of control it leaves to artists. For this reason, it is necessary to examine deep learning methods from the specific perspective of real-time game rendering.

This paper therefore reviews the use of deep learning in several important parts of the game-rendering pipeline, including super-resolution, denoising, lighting approximation, appearance modeling, and representative scene effects. Instead of treating these topics as separate technical applications, the review considers a broader issue: how deep learning is reshaping the logic of traditional real-time rendering, where it provides clear benefits, and where conventional rendering techniques remain more reliable or practical.

2. Deep learning in real-time rendering for games

Deep learning has entered real-time rendering mainly as a collection of task-oriented modules rather than as a complete replacement for the traditional graphics pipeline. In most practical systems, rasterization, shading, visibility handling, temporal reprojection, and post-processing still form the structural backbone of rendering. Neural methods are most successful when inserted into stages that are computationally expensive, numerically unstable, or difficult to tune with handcrafted heuristics. From this perspective, the most representative directions are super-resolution, denoising, lighting approximation, and appearance modeling [1].

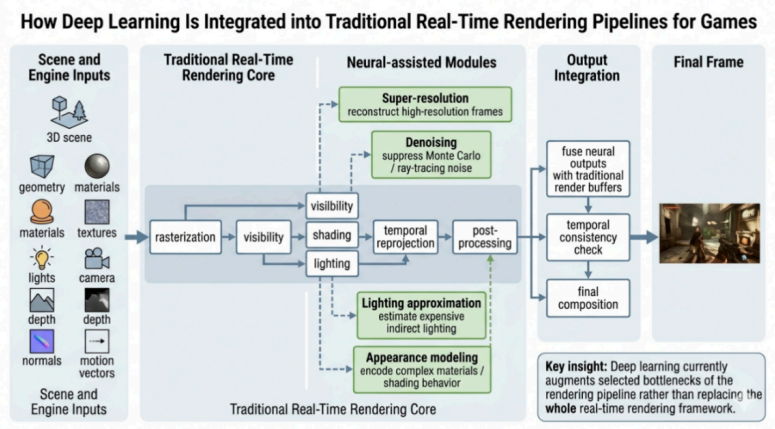

Figure 1. Overview of how deep learning is integrated into traditional real-time rendering pipelines for games (picture credit: original)

DOI: 10.54254/2755-2721/2026.GU34231

Figure 1 shows that traditional real-time rendering still provides the main pipeline structure, while deep learning is introduced as a set of auxiliary modules to enhance reconstruction, suppress noise, approximate expensive lighting, and model more complex appearance. In this sense, deep learning in games is better understood as a selective enhancement of existing rendering workflows than as a fully independent rendering framework.

2.1. Super-resolution

Super-resolution is currently one of the most mature deep-learning directions in game rendering because it directly addresses the cost of producing high-resolution frames. Instead of rendering every frame at the target display resolution, the renderer produces a lower-resolution image and reconstructs a higher-resolution output using a neural network. This approach is especially attractive for games because it is highly compatible with existing engine pipelines: motion vectors, depth, normals, and previous frames can all be exploited as structured guidance [2, 3].

A representative milestone is Neural Supersampling for Real-Time Rendering, which showed that neural reconstruction can produce high-quality and temporally stable results from low-resolution rendered inputs in real time [2]. More recent work such as FuseSR and Neural Super-Resolution for Real-time Rendering with Radiance Demodulation further improved detail recovery and temporal consistency by better exploiting multi-resolution G-buffers, radiance decomposition, and frame-recurrent designs [3, 4]. Meanwhile, Mob-FGSR extended the discussion to mobile scenarios, demonstrating that frame generation and super-resolution can also be adapted to more constrained platforms [5].

Another important trend is the fusion of spatial and temporal reconstruction. ExtraNet emphasized low-latency temporal supersampling through frame extrapolation [6], while Low-latency Space-time Supersampling for Real-time Rendering explicitly unified frame supersampling and extrapolation in a single framework [7]. These studies are important because games do not only require a sharper single frame; they also require stable motion, lower latency, and consistent visual quality over time.

From the perspective of game deployment, super-resolution is currently the most production-friendly neural rendering direction. It operates after the main rendering stage, benefits from structured engine buffers, and addresses a very clear performance bottleneck. At the same time, the main technical challenge remains temporal robustness: sharper details are beneficial only if ghosting, shimmer, and disocclusion artifacts can be controlled [2-7].

2.2. Denoising

Denoising is another central application of deep learning in real-time rendering, particularly in ray tracing and Monte Carlo rendering. Low sample counts are essential for interactive performance, but they produce severe image noise. Traditional denoisers can reduce this noise, yet they often struggle to preserve detail or maintain stability in complex dynamic scenes. Neural denoising addresses this issue by learning a mapping from noisy rendered inputs and auxiliary features to a clean output [8-11].

An early landmark is Interactive Reconstruction of Monte Carlo Image Sequences Using a Recurrent Denoising Autoencoder, which targeted interactive rates and exploited recurrent temporal information for sequence reconstruction [8]. Kernel-Predicting Convolutional Networks for Denoising Monte Carlo Renderings further showed that learned filtering can outperform fixed kernels when depth, albedo, and normals are available as guide buffers [9]. Later, Interactive Monte

DOI: 10.54254/2755-2721/2026.GU34231

Carlo Denoising using Affinity of Neural Pixels improved interactive denoising quality through learned pixel affinities [10]. These works collectively show that denoising in rendering is not simply image smoothing; rather, it is a structured reconstruction problem tightly linked to the properties of rendering buffers and temporal coherence.

For this direction, one useful point of comparison is Spatiotemporal Variance-Guided Filtering (SVGF), which is a representative traditional real-time reconstruction method. Compared with SVGF-style handcrafted filtering, deep-learning denoisers generally offer stronger detail recovery and more flexible feature fusion, especially at very low sampling rates [8-11]. However, they also introduce new deployment challenges such as training cost, content dependency, runtime inference overhead, and temporal artifacts under scene changes. Thus, neural denoising is powerful, but it remains most effective when carefully integrated into hybrid rendering pipelines rather than treated as a drop-in black box.

2.3. Lighting approximation and appearance modeling

Compared with super-resolution and denoising, lighting approximation and appearance modeling affect the rendering process at a deeper level. These methods are not mainly concerned with repairing or reconstructing the final image after rendering. Instead, they try to approximate costly light transport calculations, or represent complex material and appearance information in a more compact and efficient learned form [1].

One representative example is Real-Time Neural Radiance Caching for Path Tracing. This method trains and queries a neural radiance cache during rendering, so that indirect illumination can be estimated in fully dynamic scenes [12]. Its importance lies in the fact that deep learning is used much closer to the core computation of light transport, rather than only being applied as a post-processing enhancement. Similar ideas can also be seen in work such as LightFormer, which studies neural global illumination for dynamic scenes and shows that learning-based illumination models can approximate effects that are normally too expensive for direct real-time computation. Appearance modeling follows a related direction. Real-Time Neural Appearance Models shows that complex appearance, which was traditionally associated with offline-quality rendering, can be encoded in a compact form and decoded at real-time rates [13]. Other studies, including Deep Shading, DeepPRT, Neural Shadow Mapping, and Lightweight Neural Basis Functions for All-Frequency Shading, further suggest that neural methods can be applied to screen-space shading, radiance transfer, soft shadows, and all-frequency shading approximation [14-17].

From a game perspective, these methods are exciting because they suggest that deep learning may eventually change not only how images are reconstructed, but also how materials, shadows, indirect light, and shading behavior are represented inside the engine. Nevertheless, they are generally less mature for broad deployment than super-resolution and denoising, because they touch deeper parts of asset pipelines, shading logic, memory layout, and runtime control [12-18].

2.4. Representative scene effect: water rendering

Water rendering is a useful representative scene effect because it combines many of the difficulties of real-time graphics, including reflection, refraction, transparency, motion, foam, specular instability, and strong sensitivity to temporal artifacts. Traditional game water rendering has therefore long relied on procedural, shader-based, and physically inspired methods, such as projected-grid style surface management and modular wave, foam, and reflection pipelines [19, 20]. From the viewpoint of this review, however, water rendering should not be treated as an independent

DOI: 10.54254/2755-2721/2026.GU34231

technical line parallel to super-resolution or denoising. Instead, it is better understood as a scene effect in which several neural methods may be selectively applied.

Neural super-resolution can help recover fine ripple details; neural denoising can improve noisy reflections in ray-traced water; and neural appearance approaches may eventually model more complex view-dependent reflectance. Figure 2 is a representative visual example of this point. It presents a water-related rendering result in which neural methods are introduced to assist selected subproblems, while the final visual output is still produced through a rendering process rather than through a fully neural replacement of the graphics pipeline [21]. This example supports the argument of this review that, in water rendering, deep learning is often introduced to assist selected subproblems instead of replacing the whole rendering logic. At the same time, water also amplifies the weaknesses of learned methods: flickering highlights, layered transparency, fast-changing reflections, and dynamic surface deformation all make temporal stability especially difficult [2, 4, 9, 13].

Moreover, water is often artist-driven. Designers usually want explicit control over shoreline breakup, foam masks, wave direction, stylization, and interaction cues. This type of control is currently more transparent in rule-based shaders than in black-box learned models. Therefore, water rendering supports a broader conclusion of this review: in many game scenarios, deep learning is most useful as a supporting module for selected subproblems rather than as a full replacement for traditional rendering logic [19-21].



Figure 2. Water-related rendering image from Black Myth: Wukong [21]

2.5. Comparison: deep-learning-based real-time rendering vs. traditional real-time rendering

Traditional real-time rendering is dominated by analytical approximations, handcrafted filters, precomputed transport, probe-based illumination, and explicitly designed shader logic. Representative examples include SVGF for spatiotemporal reconstruction [22], SDF-based dynamic diffuse global illumination [23], precomputed radiance transfer methods [24], and classical water-rendering pipelines based on projected grids and modular surface effects [19, 20]. These methods are usually deterministic, interpretable, and relatively easy to tune for a known visual goal. They also offer direct engineering control, which is valuable in production.

By contrast, deep-learning-based real-time rendering replaces or augments selected stages of the pipeline with learned reconstruction or approximation modules. In super-resolution, neural methods generally outperform purely heuristic upsampling in the recovery of high-frequency detail and temporal reconstruction when sufficiently rich auxiliary features are provided [2-7]. In denoising,

DOI: 10.54254/2755-2721/2026.GU34231

learned methods often provide better detail retention than handcrafted kernels or traditional temporal filters, especially in low-sample ray-traced settings [8-11, 22]. In lighting and appearance, neural methods go even further by approximating transport or shading behavior that would otherwise require more expensive computations or heavier handcrafted precomputation [12-18, 23, 24].

However, deep learning does not dominate in every aspect. Traditional methods still have important advantages in interpretability, robustness under distribution shifts, training-free deployment, and artist control. They are often easier to debug, easier to port across projects, and less dependent on data pipelines. Neural methods, on the other hand, are strongest when the rendering problem is under-constrained, high-dimensional, or expensive enough that handcrafted approximations lose quality rapidly. In practice, this means that the current relationship between the two paradigms is not replacement but selective collaboration: traditional graphics still defines the structure of the pipeline, while deep learning increasingly handles reconstruction, enhancement, and approximation where explicit algorithms become too costly or too brittle [1, 11, 19-24].

3. Challenges and future prospects

The advantages of deep learning in real-time rendering are clear. Neural methods can recover detail from low-resolution or noisy inputs, approximate expensive lighting, and encode complex appearance in compact forms. These abilities are especially valuable in games, where rendering workloads continue to rise with higher resolutions, richer lighting, and stronger expectations for temporal stability [1, 12, 13].

At the same time, several major challenges remain. The first is runtime cost: a method that is fast in a paper may still be difficult to deploy broadly across target hardware. The second is generalization: models trained on limited scenes or styles may fail under new assets, art directions, or lighting conditions. The third is temporal stability, which remains one of the hardest requirements in interactive graphics. The fourth is pipeline integration: neural rendering methods often require training data, inference infrastructure, content-specific tuning, and debugging tools beyond those used in conventional graphics workflows [1, 9, 11].

Future work is likely to develop along two parallel lines. One line will continue improving practical neural modules, especially super-resolution and denoising, where deployment value is already clear. In this direction, the key challenges remain runtime cost, temporal stability, and generalization across diverse game content. The other line will push toward deeper integration, where neural methods participate more directly in light transport, shading, and material representation. Whether these deeper methods become standard in game production will depend not only on image quality, but also on controllability, robustness, runtime predictability, and compatibility with industrial content pipelines [12, 13, 18].

4. Conclusion

This review examined how deep learning is being incorporated into real-time rendering for games through super-resolution, denoising, lighting approximation, appearance modeling, and representative scene effects. The discussion shows that deep learning is currently most mature when used to reconstruct, stabilize, or enhance the outputs of an existing rendering pipeline. More ambitious attempts to replace the inner logic of rendering are promising, but remain less broadly deployable in practical game development.

The comparison with traditional real-time rendering further suggests that the future of game rendering is unlikely to be fully neural in the near term. Instead, the more realistic path is selective

DOI: 10.54254/2755-2721/2026.GU34231

integration: conventional graphics continues to provide the core real-time structure, while neural modules increasingly take over tasks where reconstruction, approximation, or feature fusion can be learned more effectively than they can be hand-designed. In this sense, deep learning should be understood not as the end of traditional real-time rendering, but as a powerful extension of it.

References

[1] Wang, Q., Zhong, Z., Huo, Y., Bao, H., & Wang, R. (2023). State of the art on deep learning-enhanced rendering methods. *Machine Intelligence Research, 20*(6), 799–821.

[2] Xiao, L., Nouri, S., Chapman, M., Fix, A., Lanman, D., & Kaplanyan, A. (2020). Neural supersampling for real-time rendering. *ACM Transactions on Graphics, 39*(4).

[3] Zhong, Z., Zhu, J., Dai, Y., Zheng, C., Chen, G., Huo, Y., Bao, H., & Wang, R. (2023). FuseSR: Super resolution for real-time rendering through efficient multi-resolution fusion. *SIGGRAPH Asia 2023 Conference Papers.*

[4] Li, J., Chen, Z., Wu, X., Wang, L., Wang, B., & Zhang, L. (2024). Neural super-resolution for real-time rendering with radiance demodulation. *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR).*

[5] Yang, S., et al. (2024). Mob-FGSR: Frame generation and super resolution for mobile real-time rendering. *ACM SIGGRAPH 2024 Conference Papers.*

[6] Guo, J., Fu, X., Lin, L., Ma, H., Guo, Y., Liu, S., & Yan, L.-Q. (2021). ExtraNet: Real-time extrapolated rendering for low-latency temporal supersampling. *ACM Transactions on Graphics, 40*(6).

[7] He, R., Zhou, S., Sun, Y., Cheng, R., Tan, W., & Yan, B. (2024). Low-latency space-time supersampling for real-time rendering. *Proceedings of the AAAI Conference on Artificial Intelligence, 38*(3), 2103–2111.

[8] Chaitanya, C. R. A., et al. (2017). Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics, 36*(4).

[9] Bako, S., et al. (2017). Kernel-predicting convolutional networks for denoising Monte Carlo renderings. *ACM Transactions on Graphics, 36*(4).

[10] Işık, M., et al. (2021). Interactive Monte Carlo denoising using affinity of neural pixels. *ACM Transactions on Graphics, 40*(4).

[11] Huo, Y., & Aspert, N. (2021). A survey on deep learning-based Monte Carlo denoising. *Computational Visual Media, 7*(2), 169–185.

[12] Müller, T., Rousselle, F., Novák, J., & Keller, A. (2021). Real-time neural radiance caching for path tracing. *ACM Transactions on Graphics, 40*(4).

[13] Zeltner, T., Aittala, M., Srinivasan, S., Kautz, J., & Ritschel, T. (2024). Real-time neural appearance models. *ACM Transactions on Graphics, 43*(3).

[14] Nalbach, O., Arabadzhiyska, E., Meier, T., Ritschel, R., & Ropinski, T. (2017). Convolutional neural networks for screen space shading. *Computer Graphics Forum, 36*(4), 65–78.

[15] Rao, C., Dey, A. K., & Ramani, K. (2019). Deep precomputed radiance transfer for deformable objects. *Proceedings of the ACM on Computer Graphics and Interactive Techniques, 2*(1).

[16] Datta, S., et al. (2022). Neural shadow mapping. *ACM SIGGRAPH 2022 Conference Proceedings.*

[17] Xu, Z., et al. (2022). Lightweight neural basis functions for all-frequency shading. *SIGGRAPH 2022 Conference Papers.*

[18] Ren, H., et al. (2024). Light-oriented global neural rendering in dynamic scene. *ACM Transactions on Graphics.*

[19] Johanson, C. (2004). Real-time water rendering: Introducing the projected grid concept (Master's thesis). Lund University.

[20] Darles, E., Crespin, B., Ghazanfarpour, D., & Gonzato, J.-C. (2011). A survey of ocean simulation and rendering techniques in computer graphics.

[21] NVIDIA. (2024). Black Myth: Wukong out now with full ray tracing & DLSS 3 - Get the definitive experience on GeForce RTX 40 Series GPUs.

[22] Schied, C., et al. (2017). Spatiotemporal variance-guided filtering: Real-time reconstruction for path-traced global illumination. *Proceedings of High Performance Graphics.*

[23] Hu, J., Yip, M. K., Granskog, J., Carr, N., & Nealen, A. (2021). Efficient real-time dynamic diffuse global illumination using signed distance fields. *The Visual Computer, 37*, 2539–2551.

[24] Kristensen, A. W., Forchhammer, A., & Jensen, S. G. (2005). Precomputed local radiance transfer for real-time lighting design. *ACM Transactions on Graphics, 24*(3).