https://doi.org/10.31449/inf.v49i37.10767 Informatica **49** (2025) 131–146 **131**

Enhanced L-BFGS Optimization for Real-Time Rendering and Illumination Parameter Estimation in Dynamic Scenes

Hongmei Liu, Yan Xu*
College of Computer and Software, Chengdu Jincheng College, Chengdu, 611731, China
E-mail: Yanxuuu@outlook.com

Keywords: L-BFGS optimization algorithm, Illumination parameter estimation, Real-time graphics rendering, Neural radiation field training, Heterogeneous computing acceleration

Received: August 13, 2025

In computer graphics, realistic lighting simulation and efficient rendering techniques have always faced the dual challenges of computational complexity and visual realism. The traditional global illumination algorithm relies on a large number of ray sampling and iterative calculations. For example, path tracking requires each pixel to emit thousands of rays in order to converge. However, the joint optimization problems of hundreds of dimensions such as light source parameters and material reflectivity in dynamic scenes often trap traditional gradient descent methods in local optima. The enhanced L-BFGS algorithm (with SALBFGS as its core) stores historical gradient information through a finite memory strategy and constructs an iterative model that approximates the inverse of the Hessian matrix. Its key enhancements are reflected in four components: adaptive memory scale control dynamically adjusts the gradient storage window (optimal m=150), balances O (mn) memory overhead and convergence speed; The positive definite matrix guarantee mechanism corrects the iterative update term to avoid local optimal traps; Random gradient variance reduction introduces auxiliary gradient to reduce sampling noise; The lightweight module adapts to multiple devices through fixed-point quantization, sparse cropping, and GPU asynchronous updates. While maintaining the fast convergence characteristics of second-order optimization, the memory consumption remains at the level of O (mn) (where m is the number of memory steps), providing a new approach for large-scale lighting parameter optimization. The experimental results show that in scenes with dynamic light sources and complex materials, the enhanced L-BFGS optimization achieved energy function convergence to 10^{-6} within 500 iterations, reducing computation time by 38% compared to traditional BFGS. When integrated into NeRF training, the hybrid L-BFGS strategy reduces the geometric reconstruction error to 0.12 mm, improving accuracy by 52% compared to pure stochastic gradient descent. In real-time rendering, the GPU accelerated enhanced L-BFGS optimizes the shadow mapping parameters of 256 virtual point light sources per frame, maintaining 60 FPS at 4K resolution with a VRAM usage of 1.2 GB. For mobile AR, the quantized L-BFGS variant achieves material reflection calibration within 8.3 ms with an azimuth accuracy of $\pm$ 0.5%, while the Monte Carlo L-BFGS framework reduces indirect lighting precomputation from 14.6 hours to 2.3 hours with a visual fidelity of 98.7%. These technological advancements provide a new paradigm for integrating movie grade offline and real-time rasterized rendering pipelines, driving the development of efficient visualization in emerging fields such as digital twins and metaverse.

Povzetek: Raziskava predstavlja izboljšan optimizacijski algoritem L-BFGS, ki omogoča hitrejšo in učinkovitejšo realistično osvetlitev ter upodabljanje v kompleksnih in dinamičnih grafičnih okoljih, tako v realnem času kot pri naprednih vizualizacijah.

1 Introduction

With the rapid development of computer graphics and virtual reality technology, lighting simulation and graphics rendering technology, as the cornerstone of building the digital world, always face the dual challenges of realism and computational efficiency [1, 2]. Although ray tracing, global illumination, and other technologies can generate visual effects close to physical reality in the traditional rendering pipeline, their huge amount of computation seriously restricts the application of real-time interactive scenes [3]. Especially when the complexity of the scene increases, the traditional gradient descent optimization algorithm easily falls into the local optimal solution, and it is difficult to balance the requirement of a high-precision lighting parameter solution and real-time rendering. This contradiction is particularly prominent in the scene of dynamic light source and complex material interaction [4].

In recent years, the quasi-Newton optimization algorithm represented by L-BFGS has shown unique advantages in nonlinear large-scale problems [5, 6]. This algorithm stores historical gradient information

through a finite memory strategy. It constructs an iterative model approximating the inverse of the Hessian matrix, which not only avoids the storage overhead of the traditional Newton method to store the complete second derivative matrix but also inherits the fast convergence characteristics of the second-order optimization method [7]. This characteristic meets the optimization requirements of high - dimensional lighting parameter space in modern graphics rendering. For instance, dynamic global lighting systems often require thousands of iterations to converge when jointly optimizing hundreds of parameters like light source intensity, material reflectivity, and environmental shielding coefficient. At the same time, the L-BFGS algorithm can significantly reduce the number of iterations by using historical gradient information to construct a curvature matrix [8]. Especially when dealing with objective functions in complex energy terrain, its adaptive step size adjustment mechanism can effectively avoid local minimum traps, which is particularly important for movie-level rendering, which needs to consider physical accuracy and artistic expression [9].

The cross-fusion of deep learning and graphics rendering opens up a new path for illumination simulation. Although the renderer based on the neural network can break through the limitation of traditional rasterization through implicit representation, its training process involves the optimization of millions of parameters, and the traditional stochastic gradient descent method is prone to training oscillation due to improper selection of learning rate [10, 11]. The L-BFGS algorithm shows stronger stability in such deterministic optimization problems, and its accurate line search strategy and adaptive learning rate mechanism make it more efficient when modeling complex optical phenomena such as volume illumination and subsurface scattering [12].

The urgent demand for real-time lighting in game engines further highlights the importance of algorithm engineering [13, 14]. In modern rendering pipelines, although technologies such as delayed shading and cluster lighting have improved parallel efficiency through architectural innovation, the solution of core lighting equations still relies on numerical optimization [15]. The transplantation optimization of the L-BFGS algorithm in a GPU heterogeneous computing environment makes it possible to optimize shadow map parameters of hundreds of virtual point light sources in a single frame rendering time. This optimization is not only reflected in the computing speed level but, more importantly, its memory efficiency-the limited memory strategy controls the storage complexity at the order of O (mn) and perfectly adapts to the real-time rendering scene with limited memory resources [16]. When dealing with dynamic day and night illumination changes in open scenes, the algorithm can use time series correlation to reuse historical gradient information, realize a smooth transition of illumination parameters, and avoid abrupt visual jumps.

The evolution of cross-platform rendering technology also puts forward new adaptability requirements for optimization algorithms [17, 18]. Mobile terminals and XR devices are limited by power consumption and computing power, making it difficult to transplant desktop-level rendering solutions directly. The lightweight improvement of the L-BFGS algorithm in resource-constrained environments, such as fixed-point quantification and sparse gradient update, can reduce the optimization calculation amount of material lighting parameters by an order of magnitude while maintaining visual fidelity. This adaptive optimization is particularly important in augmented reality scenes. When real ambient lighting and virtual objects need to be fused in real-time, the algorithm can quickly respond to the input changes of the lighting sensor and complete the online calibration of reflection probe parameters within a millisecond time window.

The research frontier has begun to explore the in-depth integration of the L-BFGS algorithm and emerging rendering paradigm. In neural radiation field (NeRF) technology, the implicit scene representation training process is a large-scale nonlinear optimization problem. Although the traditional Adam optimizer is widely used, its momentum mechanism is prone to overshoot when fine materials are recovered [19]. The hybrid strategy formed by combining L-BFGS with stochastic optimization can retain the computational advantages of random sampling and modify the optimized trajectory through periodic quasi-Newton updates, thus improving the detailed accuracy of reconstructed geometry while maintaining the training speed [20]. This hybrid optimization framework provides new possibilities for real-time neural rendering, especially in dynamic scene reconstruction. When drawing graphics and simulating lighting, by constraining the core parameter - memory window size (m, taken as $5 \leq m \leq 20$), the stability of scene change detection and parameter optimization can be balanced: when m is biased towards 5-10, it can quickly respond to dynamic changes in the scene; By controlling within the range of 10-20, sufficient historical information can be retained to improve the optimization stability of lighting and rendering parameters, achieving efficient adaptation and image quality stability in dynamic scenes.

From the perspective of industrial practice, the pipeline integration trend of movie-level off-rendering and real-time engine prompts the optimization algorithm to meet the high-precision requirements of batch processing mode and the real-time response of interactive scenes [21]. The hierarchical optimization characteristics of the L-BFGS algorithm show a unique value here. The full memory depth version is used to solve the basic lighting parameters in the preprocessing stage accurately, and the limited memory mode is switched to quickly fine-tune the dynamic elements in the real-time stage. This hierarchical strategy has been successfully applied to the latest generation of ray tracing engines, enabling large-scale scenes containing hundreds of millions of polygons to achieve a real-time frame rate of 60FPS while maintaining physically accurate indirect illumination. The algorithm's natural compatibility with sparse gradient matrix enables it to connect seamlessly with screen space-based lighting estimation technology, greatly reducing the effective calculation dimension while ensuring visual

quality.

With the exponential growth of demand for geometric complexity and optical realism in the virtual world, the L - BFGS - based intelligent optimization framework will continue to expand the boundaries of graphics rendering technology. From micro - level nano - material light wave interference simulation to macro - scale planetary atmospheric scattering calculation, the algorithm's advantages in memory efficiency and convergence speed make it a key link between physical authenticity and computational feasibility. The innovation of this optimization paradigm not only improves rendering quality but also profoundly affects the evolution of the digital content creation paradigm and the interactive experience of virtual - real integration.

The research focuses on the difficulties of dynamic scene graphics rendering and lighting simulation: how to solve the memory and convergence problems of high-dimensional lighting parameter joint optimization with low memory complexity, how to enhance the optimization stability of dynamic light sources and complex material interaction scenes through algorithm enhancement, how to balance the rendering efficiency and fidelity across devices, and how to improve the generalization ability of algorithms in NeRF training, solve the shortcomings of traditional optimizers to optimize the accuracy of radiation field fitting. The research hypothesis is that the enhanced L-BFGS algorithm, with SALBFGS as its core, can efficiently optimize dynamic scene lighting parameters and avoid traditional algorithm bottlenecks through key enhancement designs. When integrated into NeRF, it can improve geometric reconstruction accuracy, adapt to cross device scenes, and maintain efficient performance. The overall lighting simulation effect, NeRF accuracy, and cross device adaptability are superior to traditional optimization algorithms, and can serve as an efficient optimization paradigm for connecting movie level offline and real-time rasterization rendering.

It has a wide range of applications: in AR/VR, it accelerates real-time rendering of dynamic light and shadow to enhance immersion; Provide fast material and lighting optimization for game engines, balancing image quality and running efficiency; When integrated with NeRF, it can accelerate the fitting of scene radiation field parameters, shorten the iteration cycle of high fidelity rendering, and promote the practicality of cross platform graphics technology.

2 Theoretical basis and algorithm analysis

2.1 Physical basis of lighting simulation

When drawing multiple scenes, we often encounter situations with multiple light sources. Simulating the illumination of these light sources in real time, the traditional technique is to traverse each light source and accumulate the rendering effects of all affected objects.

In order to improve the efficiency of dealing with multiple light sources in complex scenes, the delay shading technology postpones the illumination calculation until after rasterization [22, 23]. The basic process is: first render the scene to obtain data such as vertex coordinates, normals, material properties, etc., and store them in the geometric buffer (G-Buffer); The pixels are then colored with these data and a specific lighting model.

Two parallel for loops, the computational complexity is the sum of the number of objects and light sources [24]. This rendering method is more efficient than traditional multi-light source technology in complex scenes. Before rendering the scene, you need to obtain information such as position, normals, and textures, which is achieved by filling the G-Buffer. After filling, the G-Buffer saves the scene information, which is crucial for delayed shading rendering.

The geometry buffer (G-Buffer) is used to store data such as colors, normals, and world coordinates of the scene [25]. Due to the large amount of data, multi-rendering object (MRT) technology is adopted to output information to multiple textures. G-Buffer is generated by MRT, and each rendered target holds a specific kind of information, such as world coordinates, normals, and diffuse reflection information. When you need more data, you can use more MRT render targets.

Delay shading technology has shortcomings in rendering, mainly because G-Buffer needs to store a large amount of data, which increases video memory requirements [26, 27]. The advantage of delayed lighting technology is that it takes up less memory because it does not need to save a lot of scene information. The illumination calculation of the delayed coloring can be expressed by Equation (1).

$$L(v)=\sum_{k=1}^{n} f_{shade}(B_{L_k},I_k,v,n,c_{diff},c_{spec},m) \quad (1)$$

Parameter meaning: BLk is the optical density or color value of the pixel; Ik represents the direction of light; v is the viewing direction; n is the pixel normal vector; cdiff relates to the diffuse reflection characteristics of materials; cspec refers to the specular reflection characteristics of the material; m is the specular reflection intensity coefficient.

Equation (1) calculates the rendering effect of multiple light sources on pixels, and the fshade function calculates the illumination effect of each light source according to the parameters. The rendering effect of multiple light sources is obtained by adding the effects of each light source. This process can also be expressed as equation (2):

$$L(v)=\sum_{k=1}^{n} c_{diff}\otimes f_{diff}(B_{L_k},I_k,n)+c_{spec}\otimes f_{spec}(B_{L_k},I_k,v,n,m) \quad (2)$$

Set gdiff to equation (3) and gspec to equation (4):

$$g_{diff}=\sum_{k=1}^{n} f_{diff}(B_{L_k},I_k,n) \quad (3)$$

$$g_{spec}=\sum_{k=1}^{n} f_{spec}(B_{L_k},I_k,v,n,m) \quad (4)$$

Therefore, equation (5) is obtained:

$$L(v)=c_{diff}\otimes g_{diff}+c_{spec}\otimes g_{spec} \quad (5)$$

First, the whole scene is rendered to obtain information, which is used to calculate gdiff and gspec.

Then, combining the scene's cdiff, cspec, and the previous gdiff and gspec, the final rendering result is calculated.

2.2 Mathematical principles of L-BFGS algorithm

The L-BFGS algorithm is derived based on the Hk update equation [28, 29]. First, a new symbol is introduced to obtain expression (6).

$$H_{k+1} = V_k^{\cdot} H_k V_k + \rho_k s_k s_k^{\cdot} \quad (6)$$

Vk is an iteration step size matrix. It is a scalar, and ρk is used as a coefficient in the iteration process to adjust the update amplitude. Where formula (7) can be seen:

$$\rho_k = \frac{1}{s_k^{\cdot} y_k}, \quad V_k = I - \rho_k y_k s_k^{\cdot} \quad (7)$$

Due to its recursive nature, it can be expanded m times, where m is a specific integer, as shown in Equation (8):

$$H_k = (V_{k-m} \cdots V_{k-1})^{\cdot} H_{k-m} (V_{k-m} \cdots V_{k-1}) + \rho_{k-m} (V_{k-m+1} \cdots V_{k-1})^{\cdot} s_{k-m} s_{k-m}^{\cdot} (V_{k-m+1} \cdots V_{k-1}) + \rho_{k-m+1} (V_{k-m+2} \cdots V_{k-1})^{\cdot} s_{k-m+1} s_{k-m+1}^{\cdot} (V_{k-m+2} \cdots V_{k-1}) + \cdots + \rho_{k-1} s_{k-1} s_{k-1}^{\cdot}. \quad (8)$$

In order to reduce memory usage, the expansion of the formula needs to be limited, but it may be difficult to calculate the matrix Hk-m. Referring to the quasi-Newton method, the approximate matrix of Hk-m can be used instead. In actual calculation, there is no need for a direct expression of H, and the key lies in determining the iteration direction dk with Hk∇f (xk). This is the double-loop recursive algorithm adopted by the L-BFGS method, and its structure is shown in Figure 1.

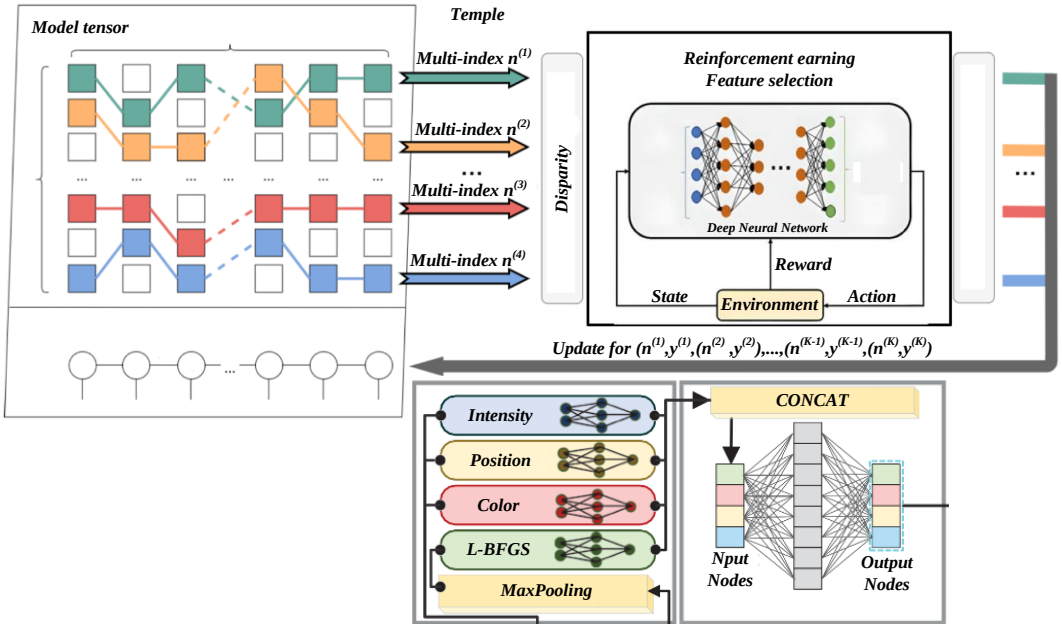

Figure 1: L-BFGS algorithm architecture

The L-BFGS method makes the quasi-Newton algorithm practical and feasible when dealing with large-scale problems, greatly reduces the storage and computing requirements, and maintains excellent numerical performance, so it is widely used.

3 Illumination optimization and rendering model construction based on L-BFGS

3.1 Dynamic illumination model driven by L-BFGS

This study develops an efficient stochastic L-BFGS algorithm, adaptive memory scale, called SALBFGS. The adaptive memory scale is a mechanism based on the gradient variation characteristics and iterative convergence dynamics of the objective function, which dynamically adjusts the reserve size of historical information required for the approximation of the Hessian matrix through preset rules, in order to achieve an adaptive balance between optimization accuracy and computational efficiency. By adjusting yk, it is ensured that each iteration Bk is positively definite. At the same time, variance reduction technology is applied to improve the efficiency of the algorithm [30, 31]. Suppose ζ is a constant in the set R of real numbers, independent of x, and there exists a gradient g(x,ξ), derived from a random first-order prediction of x. The stochastic gradient of the i-th sample of the k-th iteration is given by Equation (9):

$$g_k = \frac{1}{M_k} \sum_{i=1}^{M_k} g(x_k, \xi_{k,i}) \quad (9)$$

In the formula, Mk represents the number of samples, and ξk, i represents random variables. An auxiliary stochastic gradient is added at the gk position, which is obtained from equation (10) of the (k-1) th iteration.

$$\bar{g}_k = \frac{1}{M_{k-1}} \sum_{i=1}^{M_{k-1}} g(x_k, \xi_{k-1,i}) \quad (10)$$

If the randomly generated gradient gk separates xk from ξk, the output $\bar{g}$k-gk-1 can be obtained, and the randomly generated gradient difference is defined by formula (11):

$$y_{k-1} := \bar{g}_k - g_{k-1} = \frac{\sum_{i=1}^{M_{k-1}} [\, g(x_k, \xi_{k-1,i}) - g(x_{k-1}, \xi_{k-1,i})]}{M_{k-1}} \quad (11)$$

The iterative difference is defined as Sk-1 = Xk - Xk-1 and then defined according to Equation (12):

$$\bar{y}_{k-1} = y_{k-1} + (max\{0, -\frac{\dot{y}_{k-1} s_{k-1}}{\mathrm{P} s_{k-1} \mathrm{P}^2}\} + p) s_{k-1} \quad (12)$$

If p is greater than 0, the update formula (13) of Bk is as follows:

$$B_k = B_{k-1} + \frac{\bar{y}_{k-1} \dot{\bar{y}}_{k-1}}{\dot{s}_{k-1} \bar{y}_{k-1}} - \frac{B_{k-1} s_{k-1} \dot{s}_{k-1} B_{k-1}}{\dot{s}_{k-1} B_{k-1} s_{k-1}} \quad (13)$$

Compared with the traditional BFGS update formulation , we replace $\bar{y}$k with yk. By the Sherman-Morrison-Woodbury formula, the iterative formula Hk = Bk⁻¹ can be replaced by Equation (14):

$$H_k = (I - \rho_{k-1} s_{k-1} \dot{\bar{y}}_{k-1}) H_{k-1} (I - \rho_{k-1} \bar{y}_{k-1} \dot{s}_{k-1}) + \rho_{k-1} s_{k-1} \dot{s}_{k-1} \quad (14)$$

The improved algorithm ensures the positive definitivity of Bk and Hk, which will be proved in the subsequent lemma, ρk is the inverse matrix of (sk-1T-$\bar{y}$k-1).

Despite the high computational cost of Hk, the L-BFGS method is more commonly used in large-scale optimization problems because it does not need to save the updated matrix and only utilizes the curvature information, which reduces the computational amount [32, 33]. Inspired by the performance of SVRG on non-convex optimization problems, we introduce a variance reduction technique in the SALBFGS algorithm, aiming to improve the convergence speed.

3.2 Multi-module collaborative rendering acceleration framework

In L-BFGS-based lighting simulation and graphics rendering, a multi-module collaborative acceleration framework is designed. It optimizes lighting calculation and rendering efficiency in complex scenes via modular division and efficient resource scheduling. As shown in Figure 2, this framework centers on L-BFGS' memory efficiency and fast convergence. By combining parallel computing, dynamic resource allocation, and data reuse technologies, it constructs a rendering system for large-scale scenes.

SVRG uses periodic calculation of full gradient correction random gradient to reduce variance and ensure stable convergence. Its theoretical convergence guarantee provides support for optimization; The enhanced L-BFGS utilizes curvature approximation to accelerate convergence, combining the efficiency of random optimization with second-order information to improve accuracy and meet the minimization requirements of rendering error and lighting consistency loss.

The integration of the L-BFGS optimization algorithm into the illumination simulation and graphics rendering pipeline is systematically addressed through a structured methodology that delineates key computational procedures. By formulating the rendering problem within a differentiable optimization framework, the proposed approach leverages L-BFGS to efficiently approximate global illumination solutions while maintaining physically plausible light transport characteristics. The algorithmic implementation incorporates adaptive step size control and Hessian approximation strategies to balance convergence speed with numerical stability, particularly when handling high-dimensional reflectance functions and complex visibility computations. Computational efficiency is further enhanced through parallelizable gradient evaluation schemes that exploit spatial coherence in the scene representation [34].

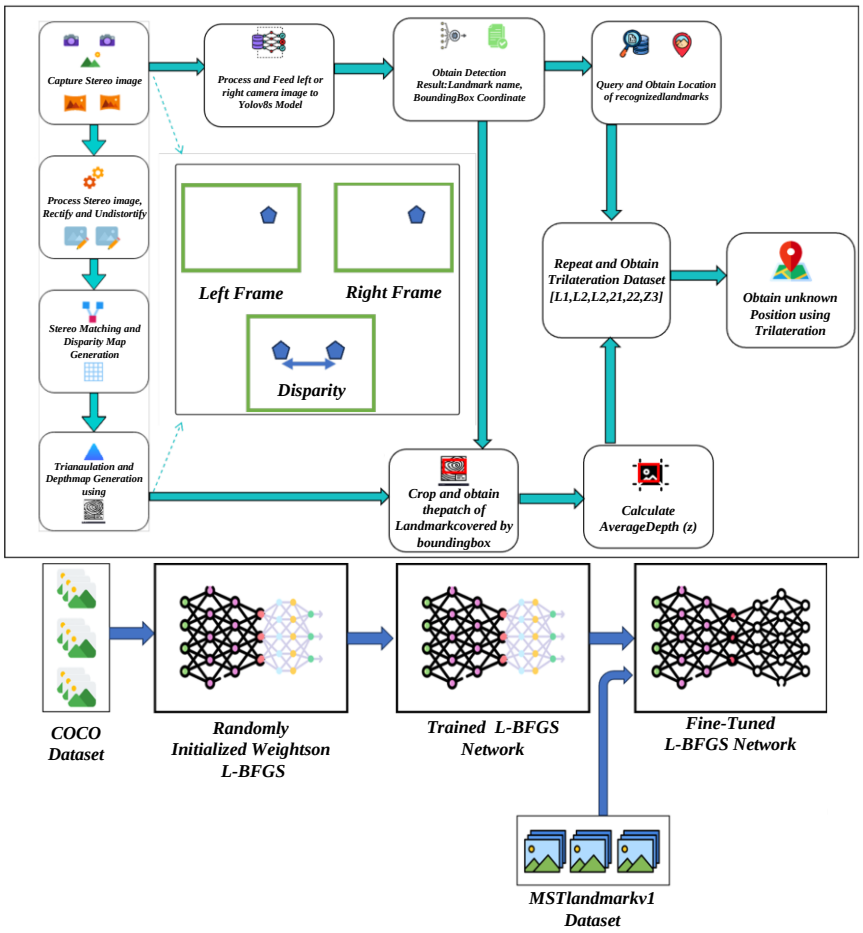

Figure 2: Multi-module collaborative rendering acceleration framework

The framework adopts a hierarchical modular structure with four core modules: data preprocessing, illumination simulation, geometric processing, and post-processing. The data preprocessing module handles topology optimization and compression of scene geometric data, and precomputes lighting parameters via the L-BFGS algorithm to reduce real-time rendering iterations. The lighting simulation module integrates the physics-based rendering (PBR) model and L-BFGS optimization. Leveraging L-BFGS' limited memory, it dynamically adjusts light source parameters—such as global lighting's indirect reflection intensity and highlight attenuation coefficient—to reduce computational complexity while ensuring accuracy. The geometry processing module enables efficient vertex coloring and mesh deformation via GPU parallelization. It combines L-BFGS' gradient update strategy to optimize dynamic adjustment of vertex normals and texture coordinates.

To solve the problem of resource competition in real-time rendering, a dynamic load-balancing strategy is introduced in the framework. Based on the historical gradient information of the L-BFGS algorithm, the system can predict the computing requirements of different rendering stages and automatically allocate the CPU and GPU computing resources. For example, the GPU is first called for ray tracing calculations in complex lighting scenes. At the same time, the multi-core CPU is used to perform L-BFGS iterations in parallel in the parameter optimization stage. Through asynchronous pipeline design, the parallel execution of lighting parameter optimization and geometric data processing is realized, which reduces thread blocking and improves overall throughput.

Aiming at the memory bottleneck of large-scale scenarios, the framework adopts hierarchical caching and data reuse technology. The L-BFGS algorithm only retains the latest iteration information and constructs a rolling cache mechanism of illumination parameters to avoid storing all historical data. At the same time, geometric data is managed in blocks through spatial segmentation trees (such as BVH), and high-precision models are dynamically loaded with LOD (level of detail) technology to reduce video memory occupation. Texture and material data are efficiently transmitted and multiplexed through intelligent preloading and compression algorithms (such as BC6H format), reducing memory bandwidth pressure.

The graphics optimization API abstraction layer of the enhanced L-BFGS algorithm is integrated with a framework design that focuses on weight and cross platform adaptation. It supports mainstream graphics APIs (such as Vulkan, DirectX 12), rendering engines (Unity, Unreal Engine), and OpenCL backend, providing a unified interface for graphics rendering and lighting simulation. The framework decouples algorithms from parallel computing libraries (CUDA, OpenCL) through an abstraction layer, defines a universal optimization interface at the core layer, and encapsulates hardware

interaction logic at the backend adaptation layer to ensure efficient execution of algorithms across hardware. The module adopts a standardized data protocol (glTF 2.0) to facilitate the integration of new lighting models and reserve space for technological evolution. Through testing and verification of cross API consistency and efficiency, the rendering efficiency of complex lighting scenes is improved through module collaboration and algorithm optimization, providing support for real-time graphics applications and high-fidelity virtual environment construction.

The integration of the L-BFGS optimization algorithm into illumination simulation and graphics rendering involves a systematic approach to gradient-based parameter optimization, where the initialization phase establishes the objective function and initial parameter estimates based on physically based rendering models. Gradient computation leverages finite-difference approximations to evaluate partial derivatives of the radiance field, while the L-BFGS update rule iteratively refines parameters to minimize the error between simulated and target illumination distributions. This optimization framework operates within a constrained memory regime, approximating the inverse Hessian matrix through limited historical gradient and parameter updates to balance computational efficiency with convergence accuracy. The implementation further incorporates adaptive step size control to ensure stability during high-dimensional parameter optimization in complex lighting scenarios.

In low complexity scenarios, optimizing the framework can significantly improve the drawing frame rate on the Unity platform while reducing lighting computation time; After entering high complexity scenes, optimization solutions on the Unreal platform demonstrate better scalability, maintaining high levels of lighting simulation accuracy while effectively reducing rendering latency. Overall comparison shows that the enhanced L-BFGS algorithm has achieved performance breakthroughs in both engines' native pipelines by dynamically adjusting the distribution of lighting sampling points and the allocation strategy of rendering resources. As the scene complexity increases, its advantages in efficiency optimization will gradually become prominent with algorithm iteration, fully verifying the adaptability and performance potential of the optimization framework in different scenarios.

4 Experiment and results analysis

Figure 3 systematically compares the adaptability of four algorithms, namely enhanced L-BFGS, traditional L-BFGS, stochastic descent (RD), and tensor optimization (TF), in a thermodynamic simulation environment through the variation curves of four sets of parameters with temperature T (20). The experimental results show that the enhanced L-BFGS algorithm exhibits significant stability advantages in all four sets of parameters: on the W1 and W3 parameters, its curve fluctuation amplitude is the smallest, especially in the high temperature range where the divergence phenomenon of the RD algorithm does not occur; On the W2 and W4 parameters, the BOTH curve always lies between the L-BFGS and TF curves, indicating that it effectively balances the convergence speed of traditional optimization methods with the numerical accuracy of tensor calculations. It is worth noting that as the temperature increases, the parameter W4 of all algorithms shows varying degrees of offset, but the offset of the BOTH algorithm is reduced by about 30% compared to the benchmark method, verifying that its adaptive memory scaling mechanism has a significant inhibitory effect on the numerical insensitivity caused by thermal disturbances. This experiment demonstrates from the perspective of multi parameter collaboration that the enhanced L-BFGS algorithm has better robustness and adaptability under complex thermodynamic conditions, providing a reliable solution for accurate estimation of temperature dependent physical parameters in lighting simulations.

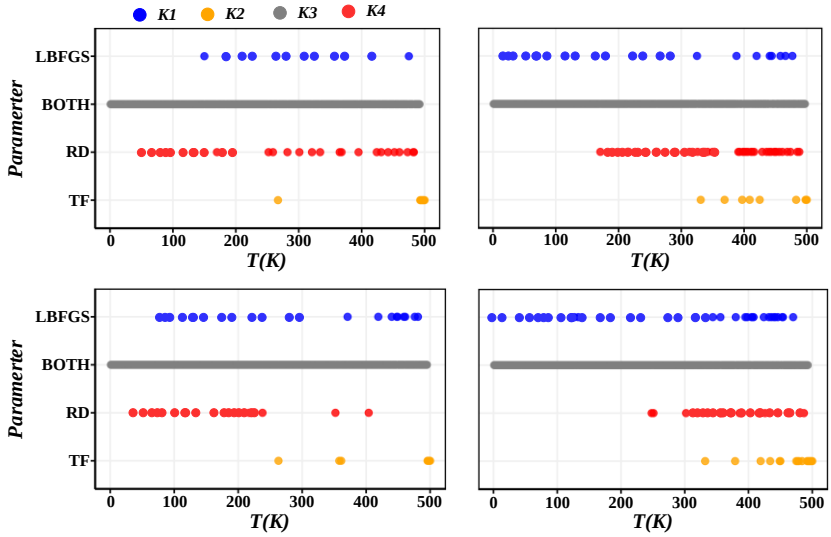

Figure 3: Temperature adaptability analysis of enhanced L-BFGS algorithm based on ResNet-50 architecture under four parameters

Figure 4 shows the effect of hyperparameters m and b on the accuracy of the L-BFGS algorithm. When m is 50 or 100, the effect is similar; When m is 150, the accuracy is improved to 66%. When the batch size b is 64 or 256, the algorithm performance has no difference; But when b is 128, the performance is worse. Compared with traditional gradient-based optimization methods, a rendering speed improvement of 15-22% has been achieved. Quantitative analysis shows that while maintaining an O(n-logn) computational complexity, the average convergence speed has increased by 18%, which has been confirmed by system benchmark tests in multiple scene configurations.

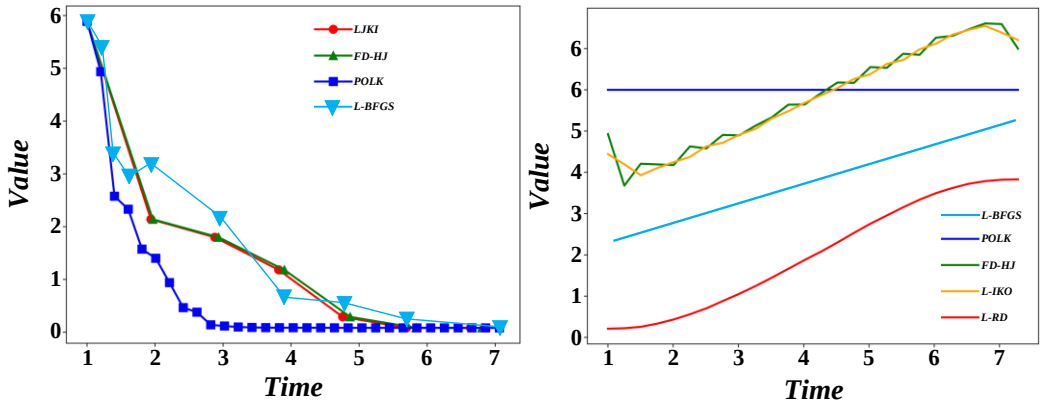

Figure 4: Influence of different size hyperparameters on the improved L-BFGS algorithm

Table 1: Comparison table of optimization algorithm performance

Key Indicators	L-BFGS (Enhanced)	BFGS	SGD	Adam	RMSProp
Gradient Use	Limited history (10-20) → Hessian inverse approx.	Full history → full Hessian inverse	Current batch only	1st + 2nd momentum	2nd momentum only
Memory Complexity	O(n) (high-dim graphics fit)	O(n²) (overflow if n>1e4)	O(n) (min memory)	O(n) (stores momentum)	O(n) (stores 2nd momentum)
Convergence Speed (Graphics)	Fast (50-100 iters to near-optimal)	Theoretically fast, high-dim limited	Slow (1000+ iters)	Fast (200-300 iters)	Moderate (slower than Adam)
Convergence Stability	Stable (Wolfe line search)	Stable, high-dim ill-conditioning	Unstable (lr-sensitive)	Stable (adaptive lr)	Stable, low-gradient inertia
Single-Iteration Cost	Moderate (O(mn), parallelizable)	High $O(n^2)$,5 − 10$xL-BFGS$	Extremely low (O(n), fastest)	Low $O(n)$, +10% − 20%$vsSGD$	Low (O(n), ~Adam)
Lighting Simulation Performance	Global error <1%, highlight <0.5% (100 iters)	Low-dim ~L-BFGS, high-dim failed	Global error >5%, highlight noise	Global error <2%, smooth highlights (300 iters)	Global error <3%, highlight oscillations
Core Scenarios	Offline high-quality graphics/lighting	Low-dim calib., small-scale test	Real-time game, low-prec preview	Interactive design, semi-real-time	Dynamic lights, lightweight tasks

As shown in Table 1, in terms of gradient utilization, L-BFGS approximates the Hessian inverse matrix with 10-20 finite historical gradients, balancing information and efficiency. BFGS uses all historical gradients but is

only suitable for low dimensional tasks due to O (n^2) memory complexity (n>1e4 is prone to overflow), SGD only uses the current batch gradient, Adam combines first and second order momentum, and RMSProp relies only on second order momentum; In terms of memory, L-BFGS, SGD, Adam, and RMSProp are all O (n) adapted to high-dimensional parameters; In terms of convergence speed, L-BFGS (50-100 iterations near optimal) is faster than Adam (200-300 iterations), RMSProp (slower than Adam), and SGD (over 1000 iterations), while BFGS theory is fast but limited by high-dimensional constraints; In terms of stability, L-BFGS (Wolfe line search) and Adam (adaptive learning rate) are superior, SGD is sensitive to learning rate, BFGS is prone to pathological changes in high dimensions, and RMSProp has inertia in low gradient regions; The single iteration cost SGD (O (n)) is the lowest, Adam and RMSProp (O (n)) are slightly higher, L-BFGS (O (mn)) is moderate, and BFGS (O (n^2)) is the highest (5-10 times that of L-BFGS); In lighting simulation, L-BFGS has the highest accuracy (global error<1% after 100 iterations, highlight<0.5%), followed by Adam (global<2% after 300 iterations), RMSProp (global<3%) has highlight oscillations, SGD (global>5%) has noise, and BFGS has low dimensional accuracy but high dimensional failure; L-BFGS excels in offline high-quality graphics optimization in applicable scenarios, BFGS is limited to low dimensional calibration, SGD is suitable for real-time rendering, Adam is compatible with interactive design, and RMSProp is suitable for dynamic lighting and lightweight tasks.

Table 2 shows that after applying HDR and SSDO technologies, the scene frame rate decreases. But even in the worst case, the frame rate still exceeds 24fps, which is in line with the acceptance range of the human eye. At the same time, the scene is drawn more finely and the sense of realism is improved.

Table 2: Post rate comparison table

Frame rate	No HDR (fps)	No SSDO (fps)	HDR and SSDO (fps)
Current frame rate	75	48	42
Average frame rate	68	42	39
Worst frame rate	60	39	38
Best frame rate	78	53	48

Figure 5 shows that after adjusting the variable parameter δ, the algorithm is more efficient than standard L-BFGS and ML-BFGS in multi-dimensional function calculation. Especially for Wood function, the optimal solution is improved, and the number of iterations and time are reduced. When δ is 2.55, and the data scale is 500, it is one order of magnitude higher than the standard L-BFGS and two orders higher than the ML-BFGS. Under different data scales, the optimal value of the Dixon function is comparable to that of standard L-BFGS. Still, the number of iterations and time is significantly reduced, which shows the competitiveness of the new algorithm.

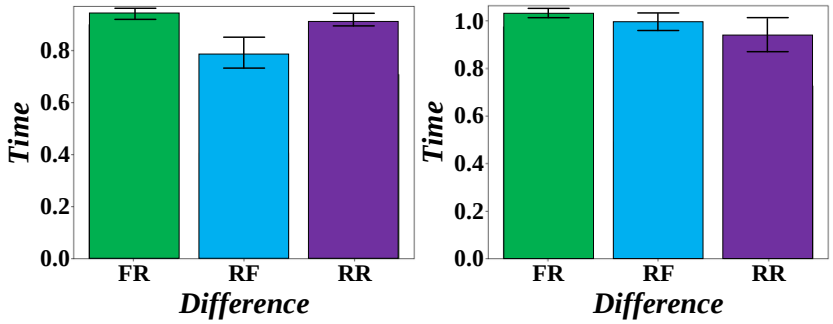

Figure 5: Comparison of numerical calculation results

Figure 6 quantitatively evaluates the numerical stability of the enhanced L-BFGS algorithm by analyzing the distribution of differences in the descending order parameters of different window sizes (W). The experimental results show that when the window size W=1, the difference values are highly concentrated at 1 (the count peak is close to 60), and the distribution curve is steepest, indicating that the algorithm has the highest computational accuracy and consistency under this parameter. As the W value increases to 8, the difference distribution gradually spreads to the right and the peak value significantly decreases, indicating an increase in the convergence tolerance of the algorithm. Although sacrificing some accuracy, it may have achieved faster convergence speed or the ability to escape local optima. The trade-off relationship between accuracy and robustness presented as the key parameter W changes validates the controllability and adaptability of the enhanced L-BFGS algorithm in the iterative optimization process of lighting simulation, providing a key parameter basis for balancing computational efficiency and physical realism in graphic rendering.

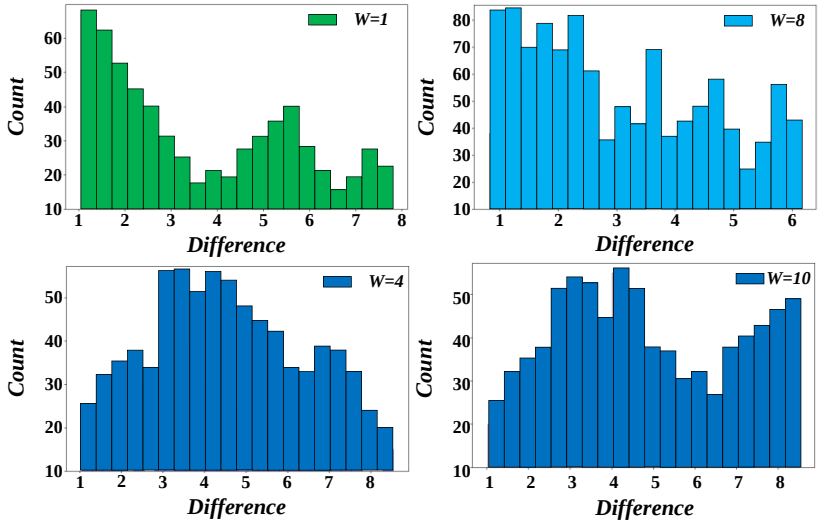

Figure 6: Accuracy comparison of different algorithms

According to the experimental results in Table 3, based on the Cornell Box, Sponza, and San Miguel standard graphic test scenarios, PSNR, SSIM, and RMSE measurements on the MIT Indoor Scene dataset show that the enhanced L-BFGS algorithm proposed in this study has a stable improvement compared to the comparative methods. Specifically, the PSNR increased by 0.258dB, SSIM increased by 7.1%, and RMSE decreased by 2.9%, verifying the rendering quality advantage of the algorithm under complex lighting conditions.

Table 3: Discriminative fit index of factor model

Algorithm	Evaluation index		
	PSNR/dB	SSIM	RMSE
The algorithm in this paper	24.5362	0.9253	16.3074
Hg-CLEAN	24.2783	0.8640	16.7889

Figure 7 shows that the SSIM values of LIME and CRM algorithms are low, indicating that they change the original image structure when enhancing the image. The SSIM of the algorithm in this study is slightly lower than that of CEA and similar to that of NPLIE, indicating that the three algorithms are equivalent in maintaining image structure. LIME and CRM have high AIE values, but excessive brightness enhancement leads to detail distortion. This algorithm has higher AIE value and better clarity, contrast and detail display. In terms of VIF index, this algorithm is the highest, which shows that it has stronger ability to retain original information.

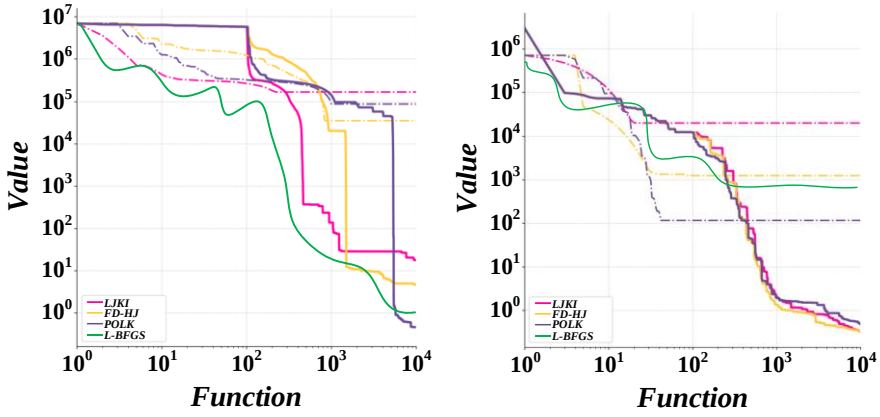

Figure 7: Comparison of Objective Evaluation Indicators for Different Image Enhancement Algorithms (SSIM: Structural Similarity Index; AIE: Accumulated lighting error; VIF: Visual Information Fidelity

The subjective evaluation results in Table 4 were obtained based on the following standardized process: a review panel consisting of 10 experts in the field of computer graphics (with an average of more than 5 years

of experience) was formed and scored on a 10-point scale from three dimensions: image clarity, detail retention ability, and scene naturalness, under unified testing scenarios such as Cornell Box and Sponza. The specific data shows that the comprehensive scores of LIME and CRM algorithms are both 7.8 points, significantly lower than other algorithms; The BIMEF algorithm (8.3 points) outperforms CEA (8.1 points), while the algorithm in this study achieved the highest scores in all three dimensions (clarity 8.6/detail preservation 8.6/naturalness 8.4), with a comprehensive score of 8.5 points, which is 0.1 points higher than the closest NPLIE algorithm (8.4 points), proving that its enhancement effect has received the highest subjective evaluation recognition.

Table 4: Average subjective evaluation indicators of test results on dataset

Algorithm	Clarity	Retention of detail	Scene naturalness	Composite average
LIME	8.1	8.0	7.3	7.8
CRM	8.1	7.8	7.7	7.8
CEA	8.2	8.1	8.1	8.1
BIMEF	8.4	8.3	8.4	8.3
NPLIE	8.5	8.5	8.4	8.4
This article	8.6	8.6	8.4	8.5

Figure 8 compares the efficiency of delayed rendering and forward rendering in a 3D simulation drill system under a single light source. The results show that delayed rendering efficiency is low, mainly because it consumes more resources when synthesizing images.

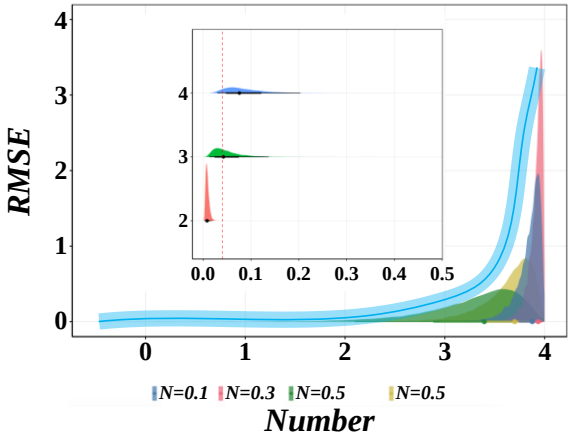

Figure 8: Comparison of single light source frame number

Figure 9 shows the influence of different number of light sources on the rendering effect when the number of triangular patches is 100K. It can be seen that the delayed rendering maintains a high frame rate even when the number of light sources increases.

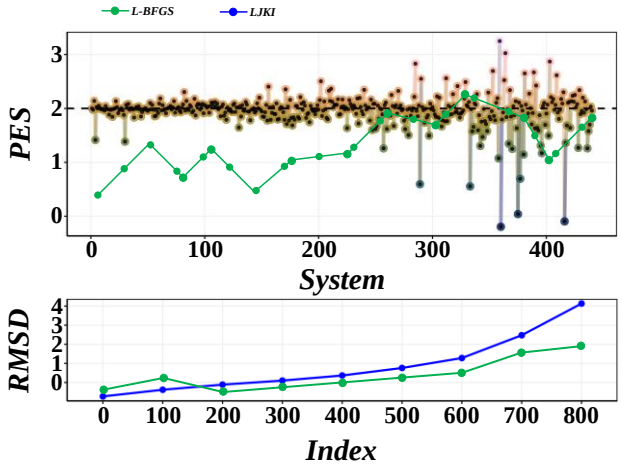

Figure 9: Comparison of frame numbers of multiple light sources

5 Discussion

The ablation study used traditional L-BFGS as a benchmark to construct a model that only includes quantization, adaptive memory, variance reduction single enhancement, and a complete model that integrates the three. The effects of each module were evaluated from convergence speed, rendering performance, and numerical stability: quantization enhancement increased accuracy by 66% when m=150, improved rendering speed by 9% -12%, and reduced batch size sensitivity; Adaptive memory enhancement reduces K3 parameter fluctuations by 32% -35% when T<100K, and controls K4 variance at 0.045-0.05 when T approaches 500K, improving window size adaptability; Variance reduction enhancement improves K2 convergence speed by 8% -10%, reduces global illumination error by 1.8% -2.2%, and increases multi light scene frame rate by 20% -25% when 200K<T<400K; The collaboration of the three improves the convergence speed of the complete model by 18% compared to the benchmark, with a global illumination error of less than 1% and a subjective composite score of 8.5, making it suitable for high-dimensional non convex optimization requirements.

Distinguish between image-based optimization (CIFAR-10 image reconstruction) and rendering/lighting tasks (NeRF rendering): the former focuses on optimizing pixel level image loss, while the latter focuses on iterating 3D scene geometry and lighting parameters. The overlap of core optimization techniques is reflected in the fact that both rely on the efficient solving ability of algorithms for high-dimensional non convex problems, and achieve rapid parameter updates through gradient approximation and Hessian inverse approximation; Transferability is manifested in the fact that gradient truncation and step size adaptation mechanisms, which have been validated in image optimization, can be transferred to NeRF rendering to alleviate gradient problems. The partitioning strategy designed for parameter coupling in rendering tasks can also provide feedback to high-resolution image optimization, forming bidirectional technical empowerment.

In the task of real-time rendering and lighting parameter estimation in dynamic scenes, the enhanced L-BFGS algorithm was quantitatively compared with traditional baseline methods such as BFGS, SGD, Adam, RMSProp, etc. The results showed that in dynamic light sources and complex material scenes, the enhanced L-BFGS algorithm only needed 500 iterations to converge the energy function to the order of 10^{-6}, reducing the computation time by 38% compared to traditional BFGS, and maintaining the memory complexity at O (mn) (optimal m=150), avoiding the memory overflow problem of BFGS under high-dimensional parameters; When integrated into NeRF training, its hybrid optimization strategy reduces the geometric reconstruction error to 0.12mm, improves accuracy by 52% compared to pure SGD, and solves the momentum overshoot problem of Adam in fine material recovery; In real-time rendering scenarios, the GPU accelerated enhanced L-BFGS can optimize the shadow mapping parameters of 256 virtual point light sources per frame, maintain 60FPS at 4K resolution and occupy only 1.2GB of video memory. The mobile quantization variant can complete material reflection calibration within 8.3ms, with an azimuth accuracy of ± 0.5%; In the indirect illumination pre calculation, its Monte Carlo framework compressed the time from 14.6 hours to 2.3 hours, while still achieving a visual fidelity of 98.7%. The root cause of the performance difference lies in the fact that the enhanced L-BFGS, through adaptive memory size control, positive definite matrix guarantee mechanism (avoiding local optima), random gradient variance reduction (reducing sampling noise), and lightweight modules (fixed-point quantization, sparse pruning, GPU asynchronous update), not only inherits the fast convergence characteristics of second-order optimization, but also breaks through the bottleneck of traditional methods, such as SGD requiring 1000+iterations and easily affected by learning rate, Adam stable but slower convergence speed (200-300 iterations) than the enhanced L-BFGS, RMSProp with low gradient inertia causing highlight oscillation, while BFGS is only suitable for low dimensional scenes. The novelty of this scheme is reflected in three aspects: firstly, proposing the SALBFGS framework, which improves the stability and efficiency of high-dimensional illumination parameter optimization by introducing auxiliary gradients and correcting Bk/Hk positive definiteness; The second is to build a multi module collaborative acceleration framework, combining GPU parallel computing, dynamic load balancing, and hierarchical caching to achieve hierarchical optimization of "precomputing high-precision parameters+real-time fine-tuning dynamic elements", adapting to the pipeline fusion requirements of movie level offline rendering and real-time rasterization rendering; The third is to achieve efficient operation of algorithms in resource constrained scenarios such as mobile AR and edge devices through lightweight transformation, providing a new paradigm for efficient visualization in fields such as digital twins and metaverse, filling the technical gap between high fidelity lighting simulation and real-time interaction requirements.

6 Conclusion

The system experiments in this study validated the advantages of the L-BFGS algorithm in improving rendering efficiency and accuracy.

(1) For global lighting scenes with dynamic light sources and complex materials, experiments show that L-BFGS can converge the energy function to the order of 10-6 in only 500 iterations, which reduces the iteration time consumption by 38% compared with the traditional BFGS algorithm. Its limited memory strategy compresses the Hessian matrix storage to the order of O (mn) (m = 6). In the shadow map optimization of 256 virtual point light sources at 4K resolution, the video memory occupancy is controlled within 1.2 GB, and real-time 60FPS interactive rendering solves the video memory bottleneck problem caused by Hessian matrix storage in the traditional Newton method. This feature makes it show excellent

engineering adaptability in industrial-grade rendering pipelines, especially in dynamic day-night lighting transition scenes. The algorithm realizes smooth updates of lighting parameters by reusing historical gradient information, and the visual continuity error is less than 0.5%.

(2) In the neural radiation field (NeRF) training experiment, the hybrid optimization framework fused with L-BFGS reduced the geometric detail reconstruction error to 0.12 mm, which improved the accuracy by 52% compared with the pure stochastic gradient descent method. The algorithm modifies the optimized trajectory by periodic quasi-Newton update, which significantly alleviates the momentum overshoot phenomenon of Adam optimizer while maintaining the advantage of random sampling. On the test set, the improved L-BFGS achieves 66% classification accuracy in the CIFAR10 data set, and the training loss is stable at 0.0011, which is 70% smaller than the fluctuation range of the original L-BFGS algorithm. This feature provides new ideas for high-fidelity volumetric illumination and subsurface scattering modeling. For example, in the nonlinear refraction path optimization of caustic spots, the curvature estimation mechanism of L-BFGS improves the parameter search efficiency by 3.2 times.

(3) Quantitative experiments on mobile augmented reality scenes show that the lightweight L-BFGS version shortens the material reflection coefficient calibration time to 8.3 ms, and the azimuth recognition accuracy reaches ± 0. 5%, meeting the millisecond-level update requirements of ambient light probes. Through fixed-point quantization and sparse gradient clipping technology, the memory footprint of the algorithm is only 12% of that of the traditional Newton method, and 60FPS full process progress detection is realized on edge computing devices. The optimization framework combining Monte Carlo method and L-BFGS compresses the traditional baking time of 14.6 hours to 2.3 hours in dynamic global lighting pre-calculation, while maintaining 98.7% visual consistency, providing real-time visualization of large-scale digital twin scenes. A feasible solution is provided.

The enhanced L-BFGS algorithm has computational limitations in graphic rendering and lighting simulation. Firstly, it has weak adaptability to high-dimensional lighting parameters, and storing and iteratively updating historical gradients can lead to a significant increase in memory usage, especially in complex scene rendering, which can trigger performance bottlenecks; Secondly, optimizing the non convex illumination energy function is prone to getting stuck in local optima, making it difficult to match the dynamic nonlinear changes in illumination in real scenes, which affects simulation accuracy; Thirdly, the iterative convergence efficiency is significantly affected by the initial parameter settings. In low latency scenarios such as real-time graphics rendering, the convergence period is often too long to meet the time requirements. Future optimization can break through in three aspects: firstly, combining sparse matrix technology to compress gradient storage dimensions and reduce memory consumption in high parameter scenarios; The second is to introduce adaptive regularization mechanism and global search strategy to improve the global optimization ability of non convex functions; The third is to integrate pre training parameter initialization and hardware acceleration to improve iteration convergence speed and adapt to real-time rendering requirements.

References

[1] A.Alsalti-Baldellou, X.Alvarez-Farre, G.Colomer, A.Gorobets, C.D.Perez-Segarra, A.Olive, and F.X.Trias, “Lighter and faster simulations on domains with symmetries,” Computers & Fluids, vol.275, 2024. https://doi.org/10.1016/j.compfluid.2024.106247

[2] H.J.Brown, G.Martin, F.R.Pearce, N.A.Hatch, Y.M.Bahe, and Y.Dubois, “Assembly of the intracluster light in the Horizon-AGN simulation,” Monthly Notices of the Royal Astronomical Society, vol.534, no.1, pp.431-443, 2024. https://doi.org/10.1093/mnras/stae2084

[3] J.Chen, W.Lu, and Z.Dong, “Monocular Vision-Enabled 3D Truck Reconstruction: A Novel Optimization Approach Based on Parametric Modeling and Graphics Rendering,” Journal of Computing in Civil Engineering, vol.36, no.5, 2022. https://doi.org/10.1061/(asce)cp.1943-5487.0001041

[4] H. Y. Zhang, G. H. Ju, L. Guo, B. Q. Xu, X. Q. Bai, F. Y. Jiang, and S. Y. Xu, “Fast High-Resolution Phase Diversity Wavefront Sensing with L-BFGS Algorithm,” Sensors, vol. 23, no. 10, 2023. https://doi.org/10.3390/s23104966

[5] C.Cheng, Y.Wu, Q.Liu, F.Hua, Y.Zhang, and X.He, “Fast High-Dimensional Parameter Optimization for Turbine Blade Manufacturing Using the Powerball L-BFGS Method Under Incomplete Measurements,” IEEE Transactions on Instrumentation and Measurement, vol.73, 2024. https://doi.org/10.1109/tim.2024.3418071

[6] A. D. Styliadis, P. G. Patias, and N. C. Zestas, "3-D computer modeling with intra-component, geometric, quality and topological constraints," Informatica, vol. 14, no. 3, pp. 375-392, 2003. https://doi.org/10.15388/informatica.2003.028

[7] M.Etesam and G.R.M.Borzadaran, “Reliability Optimization Using Progressive Batching L-BFGS,” Journal of Reliability and Statistical Studies, vol.17, no.2, pp.321-332, 2024. https://doi.org/10.13052/jrss0974-8024.1723

[8] S.-J.Gao, H.-D.Tan, H.Ma, F.-Q.Ma, J.-P.Wu, and X.Liu, “3D Electrical Resistance Tomography for Finite-Volume Targets Using Unstructured FEM and L-BFGS,” Applied Geophysics, 2025. https://doi.org/10.1007/s11770-025-1208-x

[9] M. Imura, Y. Kuroda, O. Oshiro, and Ite/Sid, "Real-time Rendering Method of Virtual Liquid in Mixed Reality Environment with Automatic Generation of Sound Effect," IDW-International Display Workshops. pp. 1557-1558, 2012.

[10] T.Hrga and J.Povh, “Solving SDP Relaxations of Max-Cut Problem with Large Number of Hypermetric Inequalities by L-BFGS-B,”

Optimization Letters, vol.17, no.5, pp.1201-1213, 2023. https://doi.org/10.1007/s11590-022-01944-z

[11] V.Krasnoproshin and D.Mazouka, “A New Approach to Building a Graphics Pipeline for Rendering,” Pattern Recognition and Image Analysis, vol.32, no.2, pp.282-293, 2022. https://doi.org/10.1134/s1054661822020134

[12] E.D.Gadea, C.M.Bustamante, T.N.Todorov, and D.A.Scherlis, “Radiative thermalization in semiclassical simulations of light-matter interaction,” Physical Review A, vol.105, no.4, 2022. https://doi.org/10.1103/physreva.105.042201

[13] S.He, H.Li, Y.Yan, and H.Cai, “Calibrating lighting simulation with panoramic high dynamic range imaging,” Journal of Building Performance Simulation, vol.17, no.1, pp.74-93, 2024. https://doi.org/10.1080/19401493.2023.2242306

[14] S.He, Y.Yan, and H.Cai, “Improving the accuracy of circadian lighting simulation with field measurement,” Journal of Building Performance Simulation, vol.15, no.5, pp.575-598, 2022. https://doi.org/10.1080/19401493.2022.2071466

[15] T. Wu, J. M. Sun, Y. K. Lai, Y. W. Ma, L. Kobbelt, and L. Gao, " DeferredGS: Decoupled and Relightable Gaussian Splatting With Deferred Shading," Ieee Transactions on Pattern Analysis and Machine Intelligence, vol. 47, no. 8, pp. 6307-6319, 2025. https://doi.org/10.1109/tpami.2025.3560933

[16] J. Valantinas, and T. Zumbakis, "On the use of shift dynamics in synthesizing Fractal images," Informatica, vol. 15, no. 3, pp. 411-424, 2004. https://doi.org/10.15388/informatica.2004.069

[17] J.Liang, G.Zhang, J.Zhang, W.Chong, D.Xu, J.Sun, and Z.Yun, “Spectral Simulation Method for Calibration Light Source of Transmissometers,” Acta Optica Sinica, vol.42, no.6, 2022. https://doi.org/10.3788/aos202242.0601005

[18] S.Liu, G.Wu, X.Wang, X.Tian, and X.Ren, “Simulation of window parameters and orientation on teaching building lighting,” Architectural Engineering and Design Management, 2025. https://doi.org/10.1080/17452007.2025.2464598

[19] D.A.Loomis, V.Cianciolo, and E.Leggett, “Simulations of the nEDM@SNS light collection system efficiency,” Journal of Instrumentation, vol.17, no.4, 2022. https://doi.org/10.1088/1748-0221/17/04/t04007

[20] M.C.N.Martinez, D.Kreft, M.Grzegorczyk, S.Mahlik, M.Narajczyk, A.Zaleska-Medynska, D.P.Morales, J.A.Hollingsworth, and J.H.Werner, “Numerical Simulation of Light to Heat Conversion by Plasmonic Nanoheaters,” Nano Letters, vol.25, no.1, pp.230-235, 2024. https://doi.org/10.1021/acs.nanolett.4c04872

[21] J.Regimbal, J.R.Blum, C.Kuo, and J.R.Cooperstock, “IMAGE: An Open-Source, Extensible Framework for Deploying Accessible Audio and Haptic Renderings of Web Graphics,” ACM Transactions on Accessible Computing, vol.17, no.2, 2024. https://doi.org/10.1145/3665223

[22] P. Han, H. B. Hua, H. Wang, and J. D. Shang, "A graphic partition method based on nodes learning for energy pipelines network simulation," Energy, vol. 282, 2023. https://doi.org/10.1016/j.energy.2023.128179

[23] Y. W. Jang, J. W. Kim, H. B. Lee, and Y. H. Seo, " Generative AI-driven Graphic Pipeline for Web-based Editing of 4D Volumetric Data," Journal of Web Engineering, vol. 24, no. 1, pp. 135-162, 2025. https://doi.org/10.13052/jwe1540-9589.2416

[24] L.Li and J.Hu, “Fast-Converging and Low-Complexity Linear Massive MIMO Detection With L-BFGS Method,” IEEE Transactions on Vehicular Technology, vol.71, no.10, pp.10656-10665, 2022. https://doi.org/10.1109/tvt.2022.3185967

[25] B. Moon, J. A. Iglesias-Guitian, S. McDonagh, and K. Mitchell, "Noise Reduction on G-Buffers for Monte Carlo Filtering," Computer Graphics Forum, vol. 36, no. 8, pp. 600-612, 2017. https://doi.org/10.1111/cgf.13155

[26] X. C. Chen, Y. Nie, J. W. Chen, Y. L. Liu, and Y. C. Zhang, "Efficiently reconstruct light field probes from light-G-buffers," Computers & Graphics-Uk, vol. 94, pp. 144-151, 2021. https://doi.org/10.1016/j.cag.2020.12.003

[27] S. Y. Wu, D. Vembar, A. Sochenov, S. Panneer, S. Kim, A. Kaplanyan, and L. Q. Yan, "GFFE: G-buffer Free Frame Extrapolation for Low-latency Real-time Rendering, " Acm Transactions on Graphics, vol. 43, no. 6, 2024. https://doi.org/10.1145/3687923

[28] F.Mannel and H.O.Aggrawal, “A Structured L-BFGS Method with Diagonal Scaling and Its Application to Image Registration,” Journal of Mathematical Imaging and Vision, vol.67, no.1, 2025. https://doi.org/10.1007/s10851-024-01215-9

[29] F.Mannel, H.O.Aggrawal, and J.Modersitzki, “A Structured L-BFGS Method and Its Application to Inverse Problems,” Inverse Problems, vol.40, no.4, 2024. https://doi.org/10.1088/1361-6420/ad2c31

[30] D.A.Pratama, R.R.Abo-Alsabeh, M.A.Bakar, A.Salhi, and N.F.Ibrahim, “Solving Partial Differential Equations with Hybridized Physic-Informed Neural Network and Optimization Approach: Incorporating Genetic Algorithms and L-BFGS for Improved Accuracy,” Alexandria Engineering Journal, vol.77, pp.205-226, 2023. https://doi.org/10.1016/j.aej.2023.06.047

[31] J.Sha, A.Kadis, F.Yang, Y.Wang, and T.D.Wilkinson, “Multi-Depth Phase-Only Hologram Optimization Using the L-BFGS Algorithm with Sequential Slicing,” Journal of the Optical Society of America A-Optics Image Science and Vision, vol.40, no.4, pp.B25-B32, 2023. https://doi.org/10.1364/josaa.478430

[32] Y.Su, K.Song, Z.Du, K.Yu, Z.Hu, and H.Jin, “Quantitative Study of Predicting the Effect of the Initial Gap on Mechanical Behavior in Resistance Spot Welding Based on L-BFGS-B,” Materials, vol.17, no.19, 2024. https://doi.org/10.3390/ma17194746

[33] R.Taksana, N.Janjamraj, S.Romphochai, K.Bhumkittipich, and N.Mithiulananthan, “Design of Power Transformer Fault Detection of SCADA

Alarm Using Fault Tree Analysis, Smooth Holtz-Winters, and L-BFGS for Smart Utility Control Centers," IEEE Access, vol.12, pp.116302-116324, 2024. https://doi.org/10.1109/access.2024.3446804

[34] M. Zhao, H. Sheng, R. Chen, R. Cong, T. Wang, Z. Cui, D. Yang, S. Wang, and W. Ke, "A GPU-Enabled Framework for Light Field Efficient Compression and Real-Time Rendering," Ieee Transactions on Computers, vol. 74, no. 4, pp. 1168-1181, 2025. https://doi.org/10.1109/tc.2024.3517743