



Frequency-importance gaussian splatting for real-time lightweight radiance field rendering

Lizhe Chen¹ · Yan Hu¹ · Yu Zhang¹ · Yuyao Ge^{1,2,3} · Haoyu Zhang¹ · Xingquan Cai¹ 

Received: 21 October 2023 / Revised: 8 February 2024 / Accepted: 21 February 2024

© The Author(s), under exclusive licence to Springer Science+Business Media, LLC, part of Springer Nature 2024

Abstract

Recently, there have been significant developments in the realm of novel view synthesis relying on radiance fields. By incorporating the Splatting technique, a new approach named Gaussian Splatting has achieved superior rendering quality and real-time performance. However, the training process of the approach incurs significant performance overhead, and the model obtained from training is very large. To address these challenges, we improve Gaussian Splatting and propose Frequency-Importance Gaussian Splatting. Our method reduces the performance overhead by extracting the frequency features of the scene. First, we analyze the advantages and limitations of the spatial sampling strategy of the Gaussian Splatting method from the perspective of sampling theory. Second, we design the Enhanced Gaussian to more effectively express the high-frequency information, while reducing the performance overhead. Third, we construct a frequency-sensitive loss function to enhance the network's ability to perceive the frequency domain and optimize the spatial structure of the scene. Finally, we propose a Dynamically Adaptive Density Control Strategy based on the degree of reconstruction of the background of the scene, which adaptive the spatial sample point generation strategy dynamically according to the training results and prevents the generation of redundant data in the model. We conducted experiments on several commonly used datasets, and the results show that our method has significant advantages over the original method in terms of memory overhead and storage usage and can maintain the image quality of the original method.

Keywords Real-time rendering · Radiance field · Novel view synthesis · Lightweight

1 Introduction

Recently, significant advancements have been made in Neural Radiance Field (NeRF) [21] methodologies. Nevertheless, a majority of the NeRF methodologies rely on implicit volume rendering approaches [7, 20], which create bottlenecks in regards to performance overhead

✉ Xingquan Cai
caixingquan@ncut.edu.cn

Extended author information available on the last page of the article

and rendering in real-time. Scholars and experts are researching ways to use explicit representations of scenes in NeRF to speed up its rendering and training. Among these, The Gaussian Splatting [13] technique is currently more effective than the traditional implicit-rendering-based NeRF. It offers nice performance and rendering quality, owing to its explicit scene representation approach.

Despite its potential, Gaussian Splatting encounters several challenges. Firstly, it poorly represents high frequency information. Secondly, it lacks sparsity in the spatial structure, leading to an exceedingly large model size, especially evident in outdoor scenarios. Thirdly, it generates a significant amount of redundant data during training iterations. These combined factors result in extremely high VRAM consumption and a high computational overhead. Consequently, training and rendering outcomes on consumer-grade graphics cards like the RTX4060 become suboptimal, severely constraining its potential for deployment on multimedia devices. Therefore, lightweight improvements are urgently required to address these challenges.

To address the inherent limitations of Gaussian Splatting, particularly its inefficiency in representing high-frequency details and the consequent bloated model sizes, our work pivots on a crucial innovation: the integration of frequency domain information into the scene representation. By strategically incorporating frequency domain information, our method significantly reduces the Gaussians needed for accurate scene depiction. The essence of our approach lies in its ability to discern most informative frequency components of the scene, thereby optimizing the Gaussian distribution for a more efficient and effective training and rendering process.

Our primary goal is to alleviate the computational burden of Gaussian Splatting in both training and rendering phases, refining the spatial representation of the radiance field without sacrificing rendering quality. While the original Gaussian Splatting method delivers commendable visual quality and rendering performance, it suffers from a hefty performance overhead during training. Moreover, models derived from training on select large scene datasets are excessively bulky, often exceeding 1 GB. This is dozens of times larger than models from traditional techniques.

In this paper, we introduce three primary enhancements. First, we design the Enhanced Gaussian with greater expressiveness for frequency information, which reduces the performance overhead required to convey high-frequency information in the scene by optimizing the model's ability to express this information. Second, we construct a frequency-sensitive loss function, which strengthens the network's perception of the scene's frequency information to optimize the sparse spatial structure of the radiance field. Finally, our dynamically adaptive density control strategy adjusts sample point generation based on scene background reconstruction, minimizing redundant data.

- We introduce the Enhanced Gaussian, a modified version of the Gaussian, designed specifically to better capture high-frequency information. This not only improves representation but also reduces the performance overhead typically associated with detailing high-frequency nuances.
- We propose a frequency-sensitive loss function. This function amplifies the network's ability to discern frequency-domain information within a scene, optimizing its spatial structure in the process.
- We develop a dynamically adaptive density control strategy. This strategy, grounded on the degree of scene background reconstruction, effectively curtails the generation of superfluous data.

2 Related works

Neural Radiance Fields (NeRF)-based novel view synthesis techniques are garnering increasing attention in the fields of computer vision and computer graphics. The original NeRF method employs Multilayer Perceptron (MLP) [32] and implicit volume rendering techniques to render new images from novel viewpoints. NeRF achieves better visual quality representation compared to other traditional novel view synthesis methods [1–3]. Subsequent work on improving and optimizing NeRF demonstrates its excellent potential. However, the original NeRF approach demands considerable computational resources and training time, and it can only deal with some small-scale scenes.

Scholars have initially addressed the inherent limitations in the NeRF [28, 33]. For instance, Mip-NeRF solves the aliasing issue in NeRF and improve rendering speed when dealing with images of different resolutions. This method achieves anti-aliasing effects and multi-scale. Although the method has achieved improvements in both quality and efficiency compared to the original NeRF, it still falls short of real-time rendering standards [4]. The method also faces challenges for dealing with unbounded scene, the mip-NeRF360 [5] method was proposed to address this issue to some extent. There are also some other works attempt to extend NeRF for dynamic scene rendering [8, 16, 17, 27, 35].

Other research has focused on leveraging the characteristics of the NeRF method to achieve various unique effects [22, 34] or develop effective tools [11, 15, 19, 26, 34, 38]. e.g., NeRFocus explicitly models thin lens imaging and derives composite cones that can be used to render each pixel with an equivalent lens effect, thus implement simulating focal length transformations of physical cameras in NeRF.

Currently, experts and scholars are primarily concentrating on optimizing the NeRF method for both rendering and training speeds. Many studies focus on improving the capacity of MLP [6, 12, 24, 31]. e.g., InstantNGP proposes a coding method based on hash search, which only requires a small-scale neural network to achieve the effect of a fully connected network without compromising accuracy. With this coding method, InstantNGP is able to create an efficient training and rendering process [23]. Plenoxels represents the scene as a sparse 3D grid with spherical harmonics. This representation utilizes gradient methods and normalization, optimizing it through calibrated images without any neural components. Plenoxels can maintain the same rendering quality as NeRF while reducing optimization time by two orders of magnitude [9].

our method is directly inspired by Gaussian Splatting, a method utilizes an interleaved control of 3D Gaussian and their densities, and use the explicit Splatting rendering algorithm [29] to render the novel view, to separate the rendering stage from the neural network component.

As shown in Fig. 1, in each iteration, the network first uses the Splatting technique to render the point cloud data into 3D Gaussian and forming a complete image. Then, the rendered image is compared with ground truth image to calculate the loss. Finally, the geometric information of the point cloud data is adaptively adjusted based on the loss. Due to the use of the explicit representation of the 3D scene, this method has a significant advantage in rendering speed compared to traditional methods based on implicit volume rendering. On multiple datasets, it exhibits better visual quality and higher training efficiency.

However, the original method's inefficient spatial structure of sample points leads to substantial performance overhead during network iterations. The most evident manifestation of this is that the model occupies a large space, and the trained model size on some large-scale datasets exceeds 1GB, which is dozens of times larger than other implicit NeRF methods.

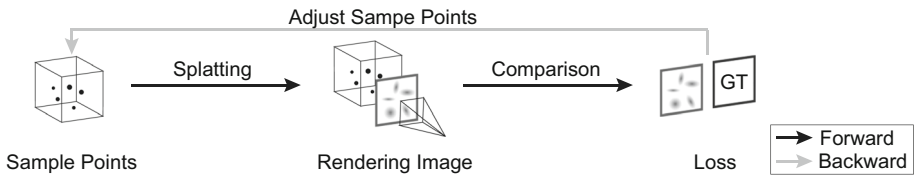


Fig. 1 Pipeline of Gaussian Splatting. Pipeline of Gaussian Splatting. First, initial sample points are obtained, either from a random source or from Structure-from-Motion (SfM)[30]. Then, in each iteration, the sample points are rendered into images, and are adjusted based on the rendered images. By repeating this process iteratively, this method achieves higher image quality and significantly outperforms traditional methods in terms of training time

To address the issues in Gaussian Splatting, we have implemented lightweight processing in this paper, significantly reducing the performance overhead of the neural network and the size of the model. Experimental results have shown that the improved method exhibits excellent performance in terms of performance overhead, while the rendering quality remains nearly unchanged compared to the original approach. In certain specialized scenes, our improved method even outperforms the original method in terms of rendering quality.

3 Methodology

The Gaussian Splatting has achieved better performance and quality in the field of novel view synthesis compared to other NeRF methods. However, there is still much room for improvement in terms of performance overheads. This is due to the poor representation of high-frequency information, insufficient spatial sparsity of the scene, and the large amount of redundant data generated during the training process in the method. In this section, we first analyze the advantages and shortcomings of the spatial sampling strategy of the Gaussian Splatting using sampling theory. We then detail the three proposed improvements: designing an Enhanced Gaussian with improved frequency expression capability, constructing frequency-sensitive loss functions, and introducing a dynamically adaptive density control strategy based on the degree of scene background reconstruction.

3.1 Analysis of spatial sampling strategy in gaussian splatting

The use of neural radiance field can be regarded as performing Monte Carlo integration on the 3D scene:

$$scene = \int c(P)dP = \frac{V}{N} \sum_{i=1}^n c(P_i) \quad (1)$$

As shown in the above, P_i represents the position information of the i th sample point, V is the volume of the sampling area, N is the number of sample points, and $c(P)$ is the scene information contained in the sample point P . In NeRF, the scene information for each sample point is iteratively obtained by the neural network, with all sample points contributing to the 3D scene reconstruction. The traditional neural radiance field method employs volume rendering with stationary sampling points, which necessitates a significant number of sample points to be placed in the scene for enhancing the rendering quality.

However, since the placement of sample points doesn't account for the scene's spatial characteristics, many points contribute little to the overall scene representation, even though having a large number of them can improve accuracy. These traditional neural radiance fields

methods incur high performance overheads and often struggle to accurately represent scene information. Although in some studies, researchers have improved the sampling process of scene information using layered sampling methods, e.g., mip-NeRF360 adopts a non-linear sampling method based on scene depth, these sampling methods still generate a large number of redundant sample points in the scene, because they still adopt an implicit 3D scene representation and require iterative computation to solve the scene information.

In Gaussian Splatting, the use of explicit spatial representation offers greater flexibility in the number and geometry of sample points compared to traditional implicit methods. Specifically, the sampling range of each sample point is controlled by a Gaussian and its covariance matrix, as shown in equation:

$$scene = \sum G(P) = \sum \exp(-\frac{1}{2}(X(P))^T \sigma(P) X(P)) \tag{2}$$

In the equation, $G(P)$ represents the Gaussian at sample point P , and $\sigma(P)$ represents the covariance matrix of P . The covariance matrix can be expressed using the geometric information of the sample points:

$$\sigma(P) = R(P)S(P)(S(P))^T(R(P))^T \tag{3}$$

$X(P)$, $R(P)$, $S(P)$ represent the position vector, rotation matrix, and scaling matrix of point P . By adjusting $\sigma(P)$, Gaussian Splatting can selectively sample scene information, eliminating irrelevant sample points and adding more points for complex regions. This approach significantly optimizes the training and rendering speed of neural networks compared to traditional implicit-rendering-based NeRF methods. We further visualize the advantages of the Gaussian Splatting method over traditional methods in terms of sampling strategy in Fig. 2.

However, this strategy has drawbacks. Farther from the Gaussian’s center, the gradient diminishes, and the function’s change smoothens, leading to potential inaccuracies. When using it to fit specific scene information, this information tends to diffuse outward due to the smooth gradient, leading to errors in the final rendering effect. We refer to this error as the

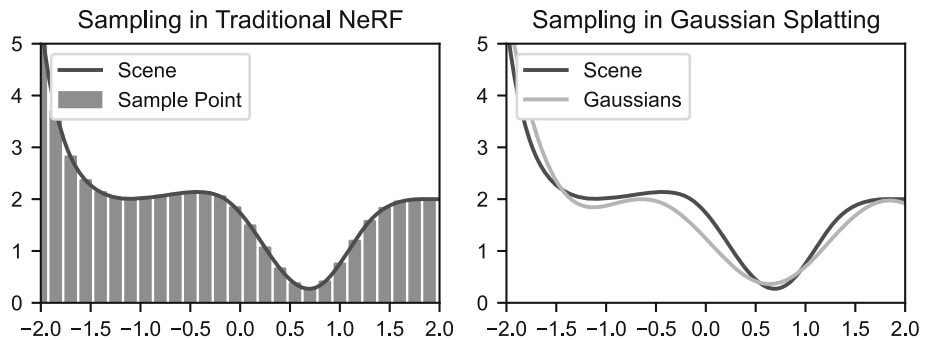


Fig. 2 A 2D simplified example illustrating the sampling methods in traditional NeRF and Gaussian Splatting. The x-axis represents specific positions in the scene, while the y-axis represents the information contained in those positions. In this Gaussian Splatting example, we only use 3 Gaussians to achieve a relatively accurate fit to the scene. In contrast, traditional NeRF requires more than 20 sample points to achieve a similar level of accuracy

'High-Frequency Diffusion Phenomenon'. While simple clamping methods can mitigate the diffusion of the Gaussian to some extent, the covariance of matrices of the Gaussians in space vary. Therefore, clamping any dimension will result in a loss of accuracy. To capture high-frequency scene details accurately, the network in original method will generate numerous adjacent or overlapping sample points at high-frequency regions. This counters the diffusion issue from gradient changes and bolsters high-frequency representation, but this strategy will greatly increase the performance overhead of the neural network and it is difficult to achieve good results.

The 'High-Frequency Diffusion Phenomenon' elucidates a challenge in using Gaussian Splatting for training. Specifically, for large-scale outdoor scenes, the neural network's learning rate must decrease to ensure convergence. This is because large-scale outdoor scenes usually contain a large amount of high-frequency information, and the number of sample points generated by the neural network at a high learning rate is too few to effectively weaken the High-Frequency Diffusion Phenomenon in large-scale outdoor scenes, making it difficult for the neural network to fit the scene information well.

On the other hand, in models trained with the original method, numerous sample points in low-frequency areas contribute minimally to rendering. We call these points Low-Contribution Sample Points (LCSPs). The size of the LCSPs is extremely small, and these points can often be replaced by a small number of larger sample points without reducing the rendering quality of the scene. We believe that these LCSPs are caused by the insufficient perceptual ability of the neural network for different frequency information in the scene.

As illustrated in Fig. 3, the neural network, lacking the capability to directly perceive scene frequency, might misinterpret low-frequency information as clustered high-frequency data. This leads to the generation of redundant sample points, these are LCSPs.

Incorporating frequency domain information into Gaussian Splatting is a strategic move to address the limitations of the original method. By enhancing each Gaussian's expression and encapsulating high-frequency details more effectively, our method seemingly increases the computational load and memory footprint per Gaussian. However, this apparent increase is counterbalanced by a significant reduction in the total number of Gaussians required for scene representation. This counterintuitive strategy results in an overall boost in performance and a decrease in memory consumption. The enhanced capacity of each Gaussian to accurately represent high-frequency information endows our method with a distinct advantage in rendering complex scenes, surpassing the original approach in both efficiency and detail fidelity. We will provide a detailed introduction to these methods in the following text.

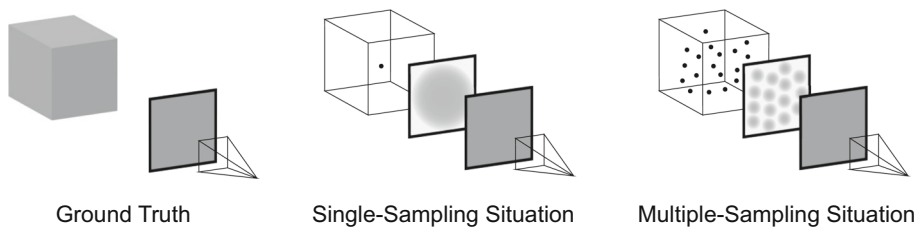


Fig. 3 An example of neural networks generating numerous LCSPs at low-frequency information. the spatial structure's rationality was compromised due to the non-linear relationship between the number of Gaussians and the degree of scene reconstruction fidelity. Beyond a certain density threshold, increasing the number of Gaussians yields diminishing returns in scene detail enhancement. This inefficiency stems from the original method's inadequate adaptive density control strategy and its neglect of frequency domain information, leading to excessive Gaussian density and, consequently, substantial redundant memory overhead. This redundancy not only burdens memory resources but also indirectly hampers rendering speed and prolongs training duration

3.2 Design of enhanced gaussian

he Gaussian's smooth gradient's limited representation capability leads to the High-Frequency Diffusion Phenomenon. To address this issue, efforts need to be made to enhance the model's ability to represent high-frequency information more effectively. In Gaussian Splatting, we introduce the Enhanced Gaussian $G_h(P)$ to replace the conventional Gaussian $G(P)$. The Enhanced Gaussian is:

$$G_h(P) = \exp\left(-\frac{1}{2}[(X(P))^T \sigma(P)(X(P))]^q\right) \quad (4)$$

In the equation, q represents the enhancement coefficient of the Enhanced Gaussian. By controlling the size of this coefficient, the network can easily control the sample point's ability to express different frequency information.

As the enhancement coefficient rises, the gradient of the Enhanced Gaussian sharpens, allowing it to capture abrupt color transitions in images, representing high-frequency information more effectively. While the Enhanced Gaussian is not immune to the High-Frequency Diffusion Phenomenon, its effect is so minimal that it can be largely ignored. Furthermore, compared to the general Gaussian, the Enhanced Gaussian only introduces a constant-level additional computational overhead, which, when compared to the performance savings achieved by using this method, is negligible. In addition, Enhanced Gaussian has the same differentiability property as the general Gaussian and is equally simple and controllable, which means the network can easily iterate on the enhancement coefficient. Figure 4 demonstrates the advantage of the Enhanced Gaussian over the general Gaussian in fitting high-frequency information.

The Enhanced Gaussian is designed with a focus on the inherent capability of Gaussian functions to represent high-frequency information. By optimizing the formulation of Gaussians, we enhance their ability to capture and express the intricate details within a scene. This optimization allows for a more accurate and detailed scene representation with fewer Gaussians, directly addressing the issue of excessive Gaussian density and the associated memory overhead.

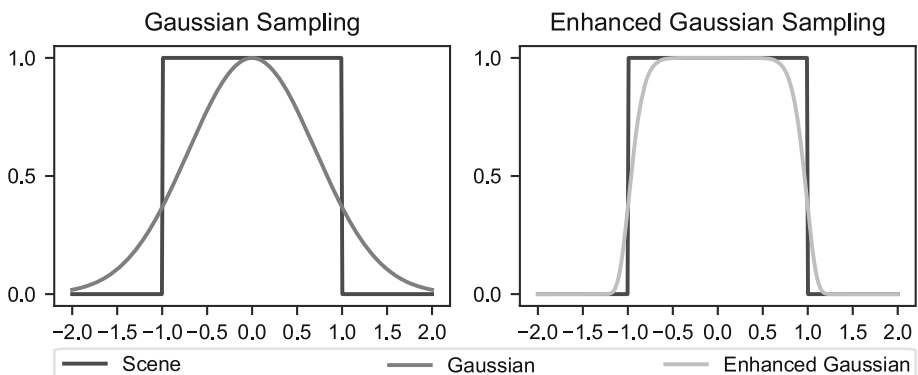


Fig. 4 Comparison of fitting capability for high-frequency information between Gaussian and Enhanced Gaussian. The x-axis represents specific positions in the scene, while the y-axis represents the information contained in those positions. The Enhanced Gaussian's edge gradient is dynamically adjustable, maintaining differentiability, which enhances its capability to represent high-frequency scene details. This makes the Enhanced Gaussian more adept at representing diverse scene information, particularly high-frequency details, compared to a standard Gaussian

While the Enhanced Gaussian introduces increased complexity in both expression and computation compared to standard Gaussian functions, its application yields a substantial reduction in the number of Gaussians required to represent high-frequency information. This mitigates the performance trade-offs associated with Enhanced Gaussian, making it a viable solution. Despite its increased intricacy, Enhanced Gaussian remains an efficient means to capture intricate high-frequency details within a scene while simultaneously reducing the computational burden and memory overhead.

According to the equation, it is known that when the enhancement coefficient of the Enhanced Gaussian reaches 16, a good representation effect of high-frequency information in the scene has been achieved. In this paper's implementation, for performance consideration, we limit the enhancement coefficient to integers between 1 and 16. Although setting the enhancement coefficient to a floating-point value between 0 and 1 might improve representation of very low-frequency details, such precision is unnecessary for low-frequency data and does not justify the additional performance overhead. The marginal improvement doesn't justify the added performance overhead compared to using integer coefficients.

3.3 Construction of frequency-sensitive loss function

The original method combines the structural similarity and photometric error of the image as a loss function, as shown in Equation:

$$L = (1 - \lambda)L_1 + \lambda L_{D-SSIM} \quad (5)$$

Here, λ is a constant that controls the balance between the two terms. While this loss function effectively measures the similarity between the rendered result and the actual image, it does not directly integrate frequency information into the neural network. We acknowledge prior works that propose methods for normalizing frequency information in neural networks [18, 25, 37]. However, these methods are tailored for NeRF-based approaches using implicit scene representations, focusing on encoding high-frequency positional data more efficiently into the network. In contrast, our goal is to reduce the model's size, and we are particularly concerned with penalizing unnecessary dense sampling points resulting from low-frequency information.

To enhance the neural network's ability to perceive scene frequency information, this paper proposes a frequency-sensitive loss function that is easy to utilize without introducing added overhead. Moreover, this frequency-sensitive loss function exhibits good transferability.

For the rendered image G obtained after training, we first use Fourier transforms to divide the image into high-frequency components G_H and low-frequency components G_L based on a frequency threshold constant t . In the same way, we obtain the high-frequency component GT_H and the low-frequency component GT_L of the GT image. Then, we calculate the structural similarity index of the high-frequency component image, low-frequency component image, and original image separately $L_{HD-SSIM}$, $L_{LD-SSIM}$, L_{D-SSIM} , and combine them with the photometric error to obtain:

$$L = \frac{(\gamma + \zeta)L_{D-SSIM} + \gamma L_{HD-SSIM} + \zeta L_{LD-SSIM}}{2\gamma + 2\zeta} \lambda + L_1(1 - \lambda) \quad (6)$$

Here, γ and ζ serve as sensitivity coefficients for high-frequency and low-frequency information, respectively. These coefficients modulate the neural network's sensitivity to the different frequency components. The Frequency-Sensitive Loss Function proposed in this work is differentiable. Each of its constituent terms, including the structural similarity

indexes ($L_{HD-SSIM}$ and $L_{LD-SSIM}$), the photometric error terms (L_1 and L_{D-SSIM}), and the coefficients (γ and ζ), are themselves continuous and differentiable functions. Combining these differentiable terms linearly in the loss function preserves its differentiability. Therefore, the Frequency-Sensitive Loss Function supports gradient-based optimization during training.

This loss function's innovation lies in its ability to divide the rendered image into high-frequency and low-frequency components (G_H and G_L) based on a frequency threshold. A similar separation is applied to the ground truth (GT) image. As shown in Fig. 5, By doing so, we can directly visualize the scene's complexity. When comparing these frequency components, it becomes evident how frequency information impacts scene representation. Unlike traditional methods that focus solely on photometric errors, our approach considers the relationship between Gaussian density and scene complexity.

We believe that the high-frequency coefficient of the Frequency-Sensitive Loss should be set slightly higher than the low-frequency coefficient. This is because, when sampling low-frequency information, a relatively lower sampling density can adequately represent the low-frequency information, while a higher sampling density is required to better represent the high-frequency information. By setting the high-frequency coefficient slightly higher than the low-frequency coefficient, the neural network will have a higher inclination to learn the high-frequency information in the scene. This conclusion is supported by the Frequency Principle (F-Principle) [36] of neural networks.

Since the structural similarity indexes ($L_{HD-SSIM}$ and $L_{LD-SSIM}$) are measured separately in the high-frequency and low-frequency regions, this loss function intuitively optimizes the spatial structure of the neural radiance field scene during the neural network's iteration. This optimization effectively prevents the oversampling problem in low-frequency

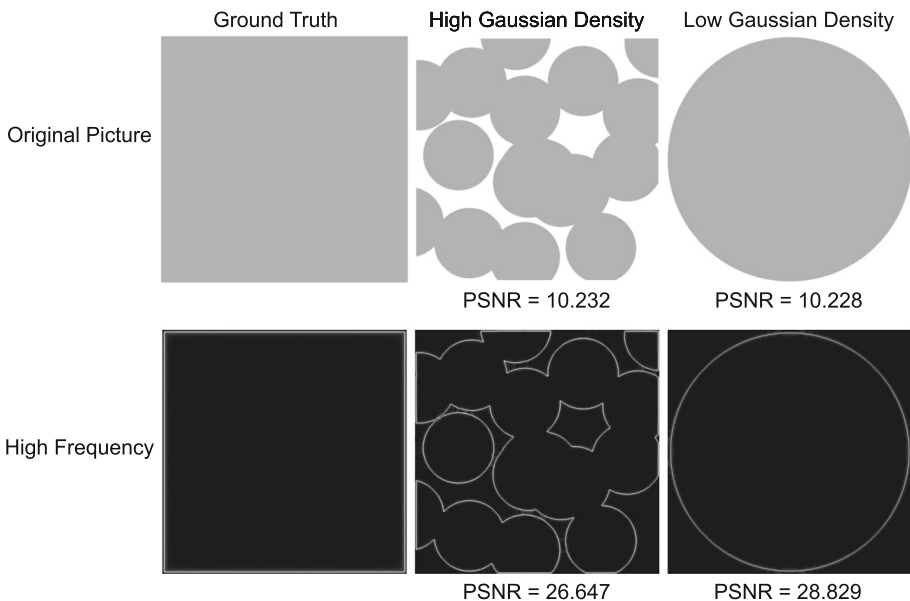


Fig. 5 When fitting relatively simple information within a scene, disregarding frequency domain information results in Gaussian density having little decisive impact on the fitting quality. However, when incorporating the scene's frequency information into consideration, the use of lower-density Gaussians demonstrates superior fitting performance for the scene. This illustrates that the introduction of frequency information into the loss function indeed plays a role in constraining Gaussian densities within the scene, reducing memory overhead

information, resulting in sparser and more reasonable sample point distributions in space. Figure 6 illustrates how our proposed frequency-sensitive loss function optimizes the model's spatial structure.

The introduction of Frequency-Sensitive Loss Functions ensures that the spatial density distribution of Gaussians aligns with the complexity of the scene. This alignment mandates that areas of the scene with higher complexity and richer high-frequency content are represented with a denser distribution of Gaussians, whereas simpler areas require fewer Gaussians. This adaptive approach to Gaussian distribution significantly reduces the redundancy observed in the original method, where the density of Gaussians was not necessarily reflective of the scene's complexity.

3.4 Dynamically adaptive density control strategy

In Gaussian Splatting, the density control of scene sampling depends on two independent strategies: Split and Clone. The Split strategy is used to handle situations where the sample points result in an excessive reconstruction of the scene. In such cases, when a Gaussian becomes too large to represent a certain geometric shape, it is split into two smaller Gaussians. On the other hand, the Clone strategy addresses situations where the scene is under-reconstructed. When a Gaussian becomes too small to represent a certain geometric shape, a Gaussian of the same size is generated in the neighboring position of this Gaussian.

In the early stages of training with the original method, the Gaussian splatting process has not fully reconstructed the scene's simple background. Many Gaussians represent only small portions of space, and the Cloned Gaussians could be effectively merged to represent larger, nearly identical regions simultaneously. Conversely, as the scene reconstruction approaches completion, applying Splits to regions already reconstructed would further increase the Gaussian density in those areas. The original method's size-based restrictions for Split and Clone operations on Gaussians seemed arbitrary and lacked adaptability to the scene's evolving reconstruction. They did not account for the evolving nature of Clone and Split operations throughout the scene reconstruction process. Given the number of sample points in the scene, encountering these minor Split and Clone issues becomes statistically inevitable.

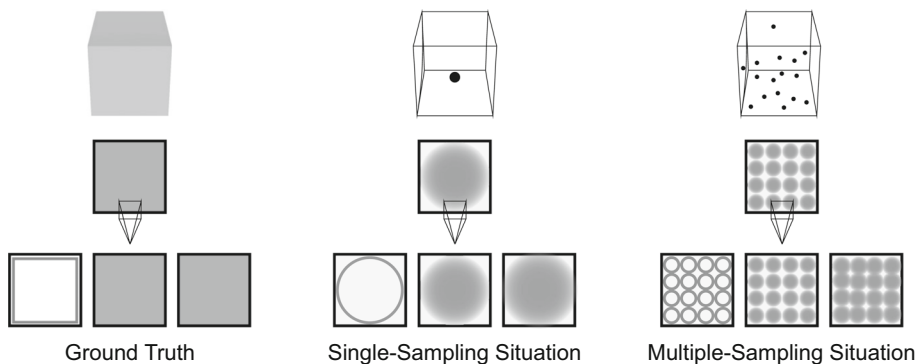


Fig. 6 The working principle example of Frequency-Sensitive Loss. Due to the introduction of frequency domain perception, the neural network will naturally avoid fitting low-frequency information with a large number of sample points. In this example, the neural network prefers using a single sample point to form an image. This is because images formed with multiple sample points introduce excessive high-frequency information which makes it have a larger error compared to the image formed by using a single sample point

While the original method restricts the execution of Split and Clone operations based on sample point size, this limitation proved unsatisfactory. In the original method, a single constant limit controlled the size of sample points eligible for Split and Clone. This caused challenges in preventing LCSP generation with a small limit and compromised rendering quality with a large limit.

To address this issue, we have developed a dynamically adaptive density control strategy that adapts throughout the training process. This strategy aims to prevent the generation of LCSPs while minimizing any negative impact on the training quality. For a given sample point P , we calculate its Split weight $f1(P)$ and Clone weight $f2(P)$ using these equations, respectively:

$$f1(P) = \frac{L}{(\alpha_1 L + \beta_1)^2} \quad (7)$$

$$f2(P) = L(\alpha_1 L + \beta_1)^2 \quad (8)$$

These weights help determine how the point should be handled in the training process. In the equations, L is the numerical value of the forward propagation loss function, which can indicate the degree of scene reconstruction. α_i and β_i are constants. During adaptive density control, only sample points with $f1(P)$ greater than 1 can undergo Clone, while only sample points with $f2(P)$ greater than 1 can undergo Split.

In comparison to the original method, our approach takes into account the progress of scene reconstruction and dynamically adjusts the usage of Clone and Split strategies based on the stage of scene reconstruction. This significantly mitigates the issue of excessive Gaussian density caused by over-Cloning in the early stages and over-Splitting in the later stages of scene fitting. This macro-level inclusion of the frequency domain in Gaussian generation considerations highlights a key difference between our Gaussian and the original Gaussian.

The Dynamically Adaptive Density Control Strategy imposes direct constraints on the generation of Gaussians during different training phases. By dynamically adjusting the threshold for Gaussian generation based on the evolving needs of the scene reconstruction, this strategy prevents the unnecessary proliferation of Gaussians. This targeted control not only enhances the training efficiency by focusing computational resources where they are most needed but also contributes to the overall reduction in model size and memory usage.

4 Evaluations and experiments

In this section, we first introduce the preliminary preparations for the experiment. Next, we designed a feasibility evaluation experiment to assess model size compression without sacrificing rendering quality. Next, we assessed the quality of our method by comparing it with other approaches. Finally, we conducted ablation experiments to demonstrate the effects of our various improvement steps.

4.1 Experimental preparation

4.1.1 Datasets

To prepare for the experiments, we carefully set up the experimental environment and collect the necessary datasets. We ensure that the training and testing data are representative of various scenes. The following datasets were used in the experiment.

Synthetic NeRF. This is the dataset used in the paper on the NeRF. The Synthetic NeRF dataset consists of 8 simple synthetic scenes: chair, drums, ficus, hotdot, Lego, materials, mic, and ship. This paper conducted experiments on all these scenes. Among them, drums and materials scenes are often used to test the model's ability to fit reflected light, as they both have specular reflection materials.

Tanks & temples [14]. The Tanks & temples dataset includes multiple open outdoor scenes. In this paper's experiment, only the Train and Truck scenes were utilized. We obtained the dataset from the link provided by Plenoxels. The Train scene comprises 258 images for the training set, while the Truck scene consists of 226 images for the training set.

Deep Blending [10]. The Deep Blending dataset is from Deep Blending for Free-Viewpoint Image-Based Rendering, which contains several indoor and outdoor scenes, and in this paper, only the Playroom and DRJOHNSON scenes are used, which are both indoor scenes and have a lot of high-frequency information. The DRJOHNSON and Playroom scenes contain 260 and 226 images, respectively.

Mip-NeRF 360. This is the dataset used in the paper on the Mip-NeRF 360 method. The dataset has a total of several large scale complex scenes captured using a professional camera, and our experiments are conducted on bicycle, counter, kitchen, garden, bonsai, stump, room and treehill. Each of these scenes provides hundreds of images. This dataset, commonly used to evaluate the NeRF method's performance with large-scale scenes, highlights the bicycle and stump scenes as the most challenging to synthesize.

4.1.2 Parameter settings and hardware configuration

To better compare the performance improvements presented in this paper, the same hyperparameter configuration as the original Gaussian Splatting method paper was used in our experiments. All experimental results were obtained using an NVIDIA 4060 GPU. However, due to differences in experimental environments, the original data from each method could not be used directly in our experiments. Instead, we downloaded the implementations of these methods from Github and obtained the experimental data by running them in our local experimental environment.

4.1.3 Evaluating indicators

In our experiments, we follow the design of the original methodology and use the eighth image of each dataset as a test set for consistent and meaningful comparisons. We adopt PSNR, L-PIPS and SSIM, three common parameters in the field of NeRF, as metrics to evaluate the quality of the generated images. Building upon an existing method, this paper also evaluates VRAM usage and model size as metrics in the experiments

4.2 Feasibility evaluation experiment

To assess the feasibility of representing scenes with high quality using fewer sample points, we designed a feasibility evaluation experiment. In this experiment, we train the original method on large-scale outdoor scenes, like gardens and stumps, across multiple iterations. By limiting the maximum number of sample points, we aim to test the potential of representing scene information with fewer sample points. The experimental data is shown in Table 1.

As illustrated in Table 1, MSPR means Maximum Sample Point Ratio, which represents ratio of the maximum number of sample points in the model in this round of experiments

Table 1 Results of feasibility evaluation experiment

Dataset	MSPR	Iterations	PSNR(Restricted)	PSNR(Unrestricted, Iterations=30K)
garden	0.1	30K	25.22	27.58
	0.5	30K	26.09	
	0.5	100K	26.31	
	0.5	300K	26.72	
stump	0.1	30K	23.20	26.19
	0.5	30K	25.01	
	0.5	100K	25.42	
	0.5	300K	25.83	
bicycle	0.1	30K	22.41	25.32
	0.5	30K	23.63	
	0.5	100K	24.11	
	0.5	300K	24.45	
treehill	0.1	30K	20.48	22.47
	0.5	30K	21.22	
	0.5	100K	21.69	
	0.5	300K	22.08	

compared to the total number of sample points in the unrestricted model, and Iterations represents the number of iterations of training. From the experimental data, we observe that the model with restricted sample points, even without additional optimization, shows minimal difference in rendering quality after numerous iterations compared to the unrestricted model. Analysis shows that this is because when the number of sample points is restricted, a large number of iterations can reduce the number of LCSPs to some extent. However, since the original method cannot express high-frequency information in scenes at a small cost, the rendering quality of the restricted model is always inferior to that of the unrestricted model.

The experimental results suggest that achieving high-quality rendering with a limited number of sample points is feasible. Furthermore, we posit that models trained using the original method contain significant redundant data. Even after eliminating this redundancy, satisfactory rendering results are achievable.

4.3 Quality evaluation experiment

In our Quality Evaluation Experiment, we meticulously assessed image quality and resource utilization across various scenes. Image quality was evaluated using PSNR and SSIM for accuracy, and LPIPS for perceptual similarity, while resource utilization was gauged through Memory Usage (Mem) for storage space, video memory usage during training (VARM), and FPS for real-time rendering performance. These metrics were presented in two distinct sets of tables: 'Quality Results of the Quality Evaluation Experiment' for PSNR, SSIM, and LPIPS, and 'Resource Utilization Results of the Quality Evaluation Experiment' for Mem, FPS, and VRAM.

We conducted quality evaluation experiments on both small-scale (e.g., Lego, Mic, Ship, Drums, Ficus) and large-scale scenes (e.g., garden, Trunk, bicycle, DRJOHNSON) to assess our proposed method's effectiveness compared to the original approach. The experimental results compared to the original method in small-scale scenes are presented in Tables 2 and

Table 2 Quality results of quality evaluation experiment in small-scale scenes

Datasets	Gaussian Splatting			Ours		
	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS
Chair	0.989	36.72	0.026	0.984	36.53	0.028
Mic	0.976	36.49	0.019	0.979	34.83	0.027
Ship	0.963	32.52	0.023	0.948	34.51	0.011
Lego	0.979	36.73	0.021	0.984	36.28	0.029
Ficus	0.977	35.12	0.022	0.985	34.23	0.039
Drums	0.968	27.23	0.083	0.946	26.81	0.066
Materials	0.943	30.98	0.052	0.954	31.11	0.058
Hotdog	0.984	38.19	0.018	0.989	38.98	0.027

3. We also conducted comparative experiments with other classic methods to obtain a better evaluation of our proposed method's quality.

As these table illustrates, our proposed approach outperforms the original method in performance metrics. Regarding visual quality, our enhanced method produces results comparable to those of the original method. Furthermore, the utilization of our enhanced method results in an average reduction in model size by a factor of 5.26 compared to models generated using the initial method. This reduction can be attributed to the lowered resource overhead required for high-frequency information representation and spatial structure optimization within scenes. However, our method lowers the maximum VRAM usage by about only 5.61%, we think the increase in VRAM usage due to image storage limits the effectiveness of our improvements in these scenarios. Figure 7 provides a visual representation of our improvements' impact on the model.

Tables 4 and 5 present the outcomes of quality evaluation experiments on large-scale scenes, highlighting the performance of our method. The analysis of experimental data indicates that our approach consistently maintains a significant advantage in terms of performance overhead. With regard to rendering quality, it's observed that our method experiences a slight reduction in certain large, complex outdoor scenes, particularly those with abundant high-frequency details, such as the 'stump' and 'garden' datasets. This marginal decline, while present, is largely imperceptible and does not detract from the overall utility of our method, especially when considering the substantial benefits in memory and computational efficiency it offers.

Table 3 Resource utilization results of quality evaluation experiment in small-scale scenes

Datasets	Gaussian Splatting			Ours		
	Mem(MB)	FPS	VRAM(GB)	Mem(MB)	FPS	VRAM(GB)
Chair	51.73	244+	2.23	6.13	244+	2.13
Mic	57.42	244+	4.25	12.22	244+	3.72
Ship	59.98	244+	4.31	13.95	244+	3.93
Lego	85.64	244+	5.38	19.83	244+	5.24
Ficus	39.11	244+	4.74	8.58	244+	4.52
Drums	65.06	244+	5.16	12.71	244+	5.09
Materials	35.42	244+	5.33	6.39	244+	4.94
Hotdog	26.30	244+	5.42	7.03	244+	5.25

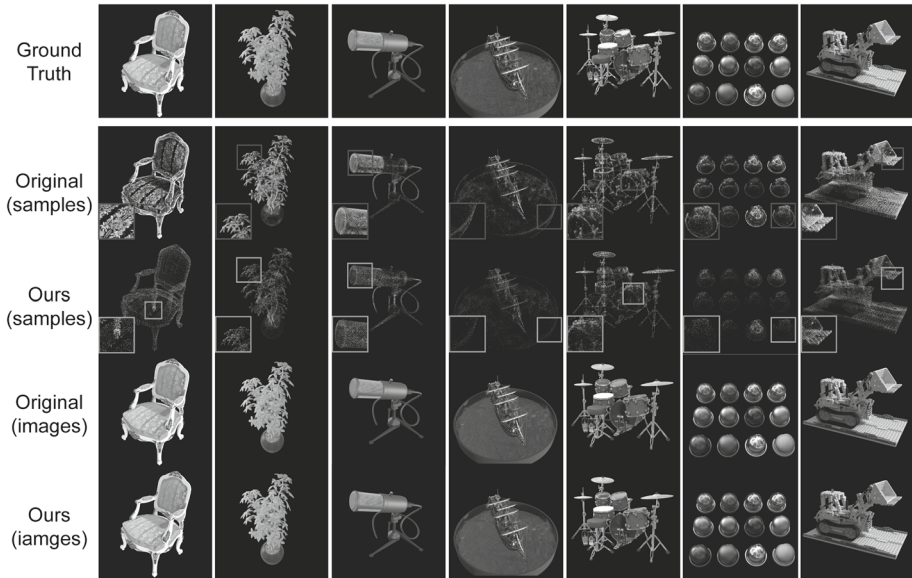


Fig. 7 In small-scale scenes, our improvement method performs well and significantly optimizes the spatial sampling structure of Gaussian Splatting. We posit that the idea of integrating frequency information into neural networks can be applied to other NeRF methods that use explicit scene representation

Importantly, this performance reduction is primarily confined to large outdoor environments. In contrast, for indoor scenes or smaller-scale settings, our method demonstrates superior performance benefits. This distinction underscores the adaptability and effectiveness of our approach across a diverse range of scenarios, reaffirming its value and applicability in both resource-constrained and detail-rich rendering tasks.

The original method’s strength, we believe, stems from its unrestricted stacking of numerous sample points at specific positions. This leads to a better representation of high-frequency

Table 4 Quality results of quality evaluation experiment in large-scale scenes

Datasets	Gaussian Splatting			Ours		
	SSIM	PSNR	LPIPS	SSIM	PSNR	LPIPS
bicycle	0.732	25.32	0.186	0.729	24.61	0.194
garden	0.849	27.58	0.197	0.823	27.16	0.204
stump	0.798	26.19	0.245	0.789	25.52	0.230
treehill	0.642	22.47	0.298	0.638	22.66	0.331
room	0.933	30.03	0.202	0.923	31.92	0.203
counter	0.901	28.84	0.194	0.908	29.05	0.189
kitchen	0.917	30.18	0.128	0.919	30.99	0.139
bonsai	0.911	32.57	0.173	0.947	32.24	0.170
Truck	0.896	25.13	0.128	0.903	26.58	0.133
Train	0.815	21.34	0.206	0.839	22.91	0.203
Dr Johnson	0.903	28.68	0.252	0.872	29.03	0.297
Playroom	0.895	30.04	0.253	0.905	30.26	0.245

Table 5 Resource utilization results of quality evaluation experiment in large-scale scenes

Datasets	Gaussian Splatting			Ours		
	Mem(MB)	FPS	VRAM(GB)	Mem(MB)	FPS	VRAM(GB)
bicycle	1433.11	85.53	16+	761.33	135.29	13.5
garden	1348.45	103.51	14.4	276.91	226.84	5.7
stump	1141.37	139.04	16+	741.36	145.52	12.9
treehill	915.76	161.62	16+	224.05	203.75	8.2
room	388.53	198.88	16+	93.68	244+	8.3
counter	283.46	213.79	16+	97.77	244+	7.4
kitchen	442.31	200.80	15.0	84.29	244+	6.9
bonsai	302.90	184.95	14.2	148.15	239.40	7.1
Truck	607.64	157.30	14.5	152.46	179.65	9.2
Train	271.39	239.86	16+	67.15	244+	7.2
Dr Johnson	842.41	178.28	16+	89.37	244+	11.9
Playroom	598.05	206.58	16+	102.85	238.68	10.3

information than the Enhanced Gaussian method. This is also the reason behind the large model space occupied by the original method for these scenes. Regarding performance overhead, our proposed method cuts the average model size by a factor of 4 and decreases VRAM usage during training by 41.92%. In the best scenario, our method cuts the model size by a factor of nearly 10, and decreases VRAM usage during training over 50%. Figure 8 presents a comparison between the results obtained with the original method and our method across different large-scale scene datasets.

It's noteworthy that our method demonstrates superior performance over the original approach across both small-scale and large-scale datasets, maintaining advantages in memory overhead and storage space utilization. This might appear counterintuitive at first glance. Our analysis suggests that this is due to the fact that, although our method increases the cost of rendering each Gaussian and incurs additional computational expenses in loss function calculations during training, it significantly reduces the total number of Gaussians. Moreover, our method prevents the generation of excessive, non-contributory Gaussians through the Split and Clone processes during training, leading to at least a twofold decrease in the total number of Gaussians compared to the original method. This reduction in Gaussians accounts for the improved performance overhead on the same datasets. In other words, our method does not directly enhance the performance of the original Gaussian rendering; instead, it achieves reduced memory consumption and performance overhead by decreasing the number of Gaussians required to describe the scene. This also explains why, on certain datasets like stump, our method substantially reduces the required storage space but does not significantly improve the frame rate—the datasets still contain too many Gaussians, and the efficiency of rendering each Gaussian is slightly lower than that of the original method.

Besides comparing our proposed method with the original, we also evaluated its performance against other state-of-the-art NeRF approaches that use implicit scene representation, include InstantNGP and Mip-NeRF360. InstantNGP is renowned for its rapid and efficient training, standing as the current fastest implicit geometry representation method. Meanwhile, Mip-NeRF 360 offers the highest rendering quality among existing NeRF approaches that employ implicit geometry representation. Table 6 illustrates the comparative data between our proposed method and the aforementioned techniques.

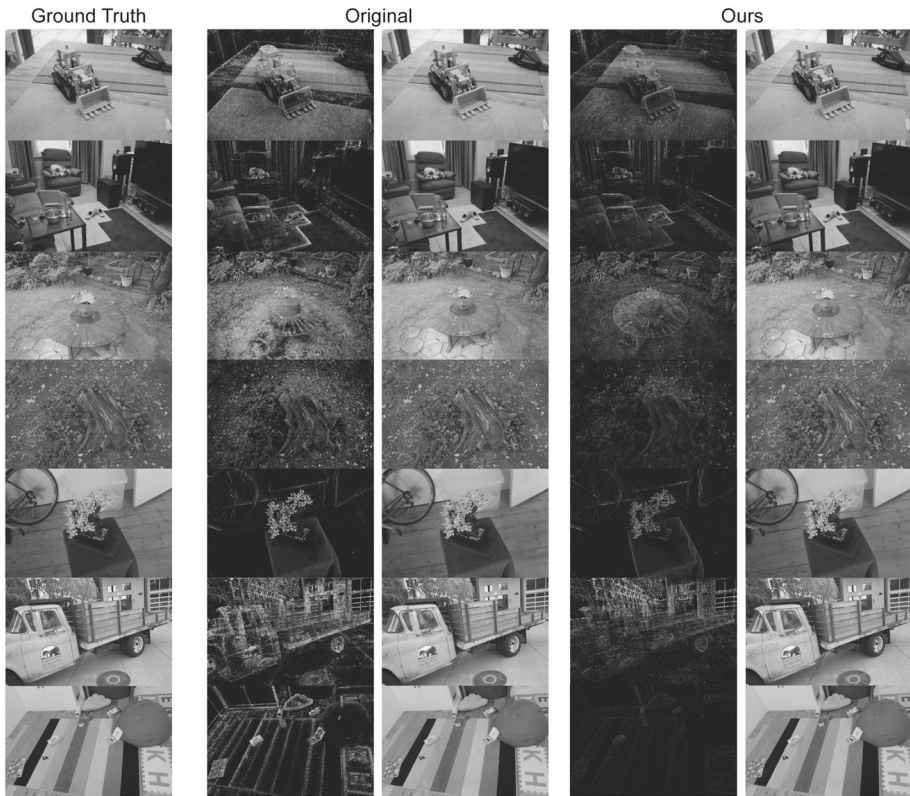


Fig. 8 On large-scale datasets, our method still achieves excellent performance. Excluding the VRAM overhead from data loading, our method’s VRAM overhead is less than half that of the original method, yet it maintains a consistent rendering quality. These findings underscore the effectiveness of integrating frequency information into NeRF’s explicit consideration

While our method exhibits a slower training speed compared to InstantNGP, it boasts superior rendering efficiency. This can be attributed to the implicit scene representation, which doesn’t store the position information of sample points and instead relies on iterative methods for gradual scene information inference. Additionally, we noticed that our method outperforms InstantNGP in terms of rendering quality for both small-scale and large-scale scenes. We surmise that InstantNGP’s use of hash encoding for position information, while accelerating scene information queries, might be a source of errors. The method’s reliance on

Table 6 Comparison results with other methods using the Mip-NeRF 360 dataset

Dataset Method\Metric	Indoor(avg)			Outdoor(avg)		
	PSNR	Training	FPS	PSNR	Training	FPS
mip-NeRF 360	31.37	24h+	0.003	25.06	24h+	0.001
InstantNGP	29.13	13m	12.51	23.26	33m	7.37
Gaussian Splatting	30.41	2h02m	232.72	25.39	3h26m	80.38
Ours	31.05	58m	244+	24.99	1h42m	178.52

Table 7 Results of ablation experiment with outdoor scene in tanktemples dataset

Dataset Method\Metrics	Truck			Train		
	PSNR	Mem	VRAM	PSNR	Mem	VRAM
No Enhanced Gaussian	26.02	428	10.6	21.87	158	13.2
No Frequency-Sensitive Loss	25.93	441	9.7	22.02	129	10.3
No Dynamic Clone	26.41	524	13.3	22.97	252	13.4
No Dynamic Split	26.24	483	12.9	22.56	176	11.7
Full	26.58	388	9.2	22.91	67	7.2

implicit scene representation also leads to a slower rendering speed, a fundamental challenge that's hard to address.

When juxtaposing our method with mip-NeRF 360, we observe a minor decline in rendering quality relative to it. Yet, our method stands out with distinct advantages in both training and rendering phases. While this advantage is evident when comparing the original method to mip-NeRF 360, our method further accentuates it.

4.4 Ablation experiment

We conducted ablation experiments targeting the three improvements presented in this paper to ascertain their positive impact on the original method. In these experiments, we sequentially removed each step of improvement to evaluate their overall contribution to the method. We employed high-resolution, large-scale scene datasets to underscore our method's enhanced impact. Employing these datasets enabled us to more effectively evaluate each step in our method. Tables 7 and 8 display the experimental data from the ablation experiments.

As the tables illustrate, when comparing the model that uses the Enhanced Gaussian to the one employing the regular Gaussian, there's a notable decline in rendering quality without the Enhanced Gaussian after an equal number of iterations. This quality degradation can be attributed to the High-Frequency Diffusion Phenomenon. Specifically, without using Enhanced Gaussian, the neural network will not be able to fit scene information well at a low cost to cope with High-Frequency Diffusion Phenomenon. Simultaneously, as most sampling points are utilized to fit high-frequency information, the representation quality of low-frequency information in the scene deteriorates. The experimental results indicate that our proposed Enhanced Gaussian has a significant advantage in representing high-frequency information. Figure 9 showcases the rendering results' transformation when applying the Enhanced Gaussian.

Table 8 Results of ablation experiment with indoor scene in deep blending dataset

Dataset Method\Metrics	Playroom			Dr Johnson		
	PSNR	Mem	VRAM	PSNR	Mem	VRAM
No Enhanced Gaussian	30.15	335	15.7	28.77	383	16+
No Frequency-Sensitive Loss	30.07	135	11.3	28.71	209	12.7
No Dynamic Clone	30.31	404	16+	28.98	777	16+
No Dynamic Split	30.13	322	14.4	28.84	542	15.5
Full	30.26	102	10.3	29.03	89	11.9

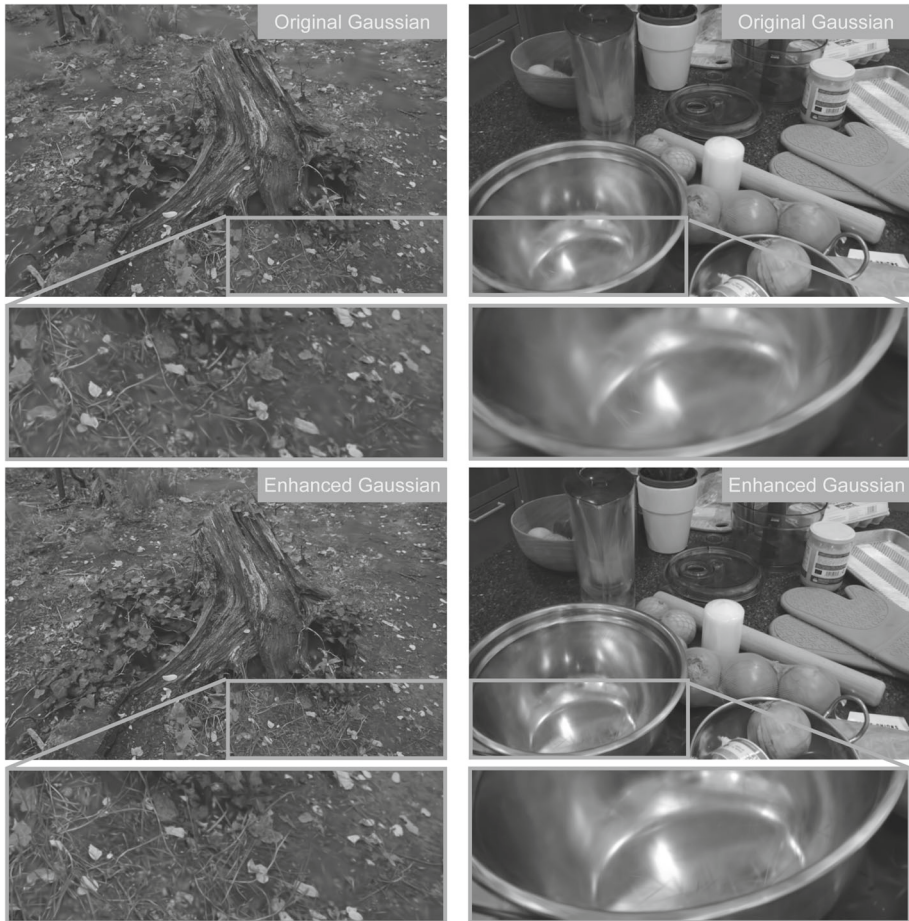


Fig. 9 Without using Enhanced Gaussian, the image exhibits significant blurring and loses a lot of details. This outcome arises from the Enhanced Gaussian’s capability to bolster the model’s representation of high-frequency scene information, thereby refining the image

Upon removing the frequency-sensitive loss function from our method, we noted a distinct reduction in rendering quality and a minor uptick in model size. This occurred because the frequency-sensitive loss function played a crucial role in optimizing the spatial structure of neural radiance field sampling. The function adeptly lowers the sampling density for low-frequency details while amplifying it for high-frequency elements in the scene, resulting in a more accurate scene representation. Figure 10 depicts the training outcomes influenced by the frequency-sensitive loss function under different parameter settings.

The integration of frequency information into the loss function has nuanced implications for rendering quality. Unlike the original Gaussian Splatting, where a higher density of Gaussians was generally better for scene representation, our approach aligns Gaussian density with the scene’s frequency content. This ensures that once a Gaussian adequately represents a portion of the scene, its density does not increase unnecessarily, preventing over-saturation and optimizing memory usage.

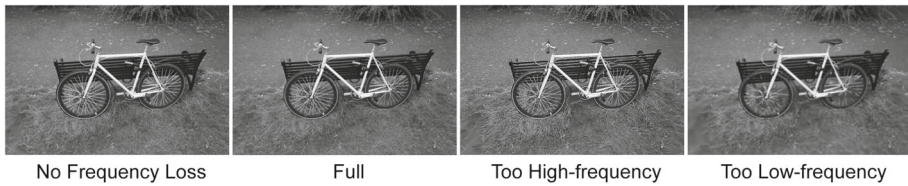


Fig. 10 In the experiments, although using a frequency-sensitive loss function helps optimize rendering quality and model size, the setting of the loss function must be within a certain range. If the loss function overly favors high-frequency information, the rendered result exhibits undue sharpening and blurring. On the other hand, if the loss function leans excessively towards low-frequency information, the rendered image appears predominantly blurry

The results indicate that deploying a frequency-sensitive loss function with optimal settings accelerates the neural network's recovery of intricate high-frequency scene details and alleviates the computational strain from high-frequency data reconstruction. Additionally, if the sensitivity coefficient of high-frequency information is overly high, the neural network will give priority to fitting high-frequency information in the scene. This leads to sharper images but also introduces more artifacts.

Figure 11 shows partial experimental results of the ablation experiment on the dynamically adaptive density control strategy. In this experiment, the Clone strategy and the Split strategy of dynamically adaptive density control were successively removed. From the experimental data and the accompanying graph, we deduce that the dynamic adaptive density control method substantially diminishes the count of LCSPs. Specifically, both the dynamically adaptive Clone strategy and Split strategy generate fewer LCSPs during training.

Concurrently, the dynamically adaptive Split strategy augments the neural network's grasp of scene frequency, facilitating rapid scene reconstruction. The dynamically adaptive Clone strategy enhances the rendering quality of high-frequency information in the scene after training. The results indicate that the dynamically adaptive control strategies for Split and Clone are crucial for model lightweighting.

5 Discussion and conclusions

We show the benefits and limitations of explicit Gaussian Splatting in comparison to the implicit NeRF method for modelling from a sampling theory perspective. We suggest three enhancements to Gaussian Splatting based on this examination. First, we design the Enhanced Gaussian to strengthen the model's ability to express high-frequency information. Next, we devise a frequency-sensitive loss function to improve the neural network's perception of frequency-domain information and optimize the spatial sampling structure of the radiance field scene. Finally, we propose a dynamically adaptive density control strategy that adjusts the generation strategy of sample points based on the degree of scene background reconstruction. This approach significantly reduces the quantity of redundant data in the model, resulting in a lightweighting of the Gaussian Splatting approach that maintains its rendering quality while reducing resource overhead. Our research suggests that the radiance field with explicit scene representation holds considerable promise, meriting further enhancement.

However, our optimization strategy currently still overlooks extraneous data within a segment of the scene, such as some LCSPs (Low-Contrast Static Points). Specifically, our method cannot completely block the generation of LCSPs. This leads to the presence of redundant data in the models generated by our approach. It is worth noting that this limitation primar-



Fig. 11 Experimental results have shown that, in the initial stage of scene fitting, enhancing Split helps to quickly reconstruct the background of the scene, whereas limiting Clone greatly reduces the redundant scene fitting overhead. During the final scene fitting phase, constraining Split effectively manages the model’s size, and bolstering Clone empowers the neural network to refine the fitting of high-frequency details

ily impacts the performance in large-scale outdoor scenes, where the abundance of LCSPs can slightly reduce the efficiency of our method. Conversely, in smaller-scale indoor environments, where LCSPs are less prevalent, our approach does not exhibit this performance decrement, maintaining high effectiveness and accuracy. We anticipate that by thoroughly purging the model of this redundant data, coupled with an effective sampling noise reduction strategy and the utilization of Vulkan or CUDA, we can achieve enhanced radiance field training and rendering outcomes on mobile platforms. In upcoming research, our objective is to probe the efficiency of discrete radial fields grounded on sample points, sidestepping subjective evaluations.

In summary, we introduce a potent lightweight enhancement for the 3D Gaussian Splatting Radiance Field, which significantly curtails performance demands and resource overheads, all the while retaining rendering excellence.

5.1 Algorithm description

In this section, we offer pseudocode outlines of key algorithms to aid in understanding and replication of our work.

The first algorithm detailed below computes the Frequency-Sensitive Loss function. This specialized loss function is crucial for maintaining the balance between high-frequency detail fidelity and overall image accuracy, thereby significantly enhancing the quality of synthesized views by appropriately weighting different frequency components.

Algorithm 1 Frequency-sensitive loss function

```

1: function FREQUENCYSENSITIVELOSS(rendered, ground_truth, low_freq_radius,  $\gamma$ ,  $\zeta$ ,  $\lambda\_val$ )
2:   low_rendered, high_rendered  $\leftarrow$  PROCESSIMAGE(rendered, low_freq_radius)
3:   low_truth, high_truth  $\leftarrow$  PROCESSIMAGE(ground_truth, low_freq_radius)
4:   l1_high  $\leftarrow$  L1LOSS(high_rendered, high_truth)
5:   l1_low  $\leftarrow$  L1LOSS(low_rendered, low_truth)
6:   ssim_high  $\leftarrow$  SSIM(high_rendered, high_truth)
7:   ssim_low  $\leftarrow$  SSIM(low_rendered, low_truth)
8:   loss  $\leftarrow$   $\lambda\_val \times ((1-\gamma) \times l1\_low + \gamma \times ssim\_low) + (1-\lambda\_val) \times ((1-\zeta) \times l1\_high + \zeta \times ssim\_high)$ 
9:   return loss
10: end function

```

Following the Frequency-Sensitive Loss function, we introduce an image processing algorithm crucial for the separation of images into their high and low-frequency components. This procedure is instrumental in preparing the input data for our main algorithm by isolating frequency bands, thereby enabling targeted processing of different image details.

Algorithm 2 Image frequency separation.

```

1: function PROCESSIMAGE(image_path, low_freq_radius = 20)
2:   image ← OPENIMAGE(image_path)                                ▷ Read and convert to grayscale
3:   image_array ← TOARRAY(image)
4:   f_transform ← FFT2(image_array)
5:   f_shift ← FFTSHIFT(f_transform)
6:   mask ← CREATEMASK(rows, cols, low_freq_radius)              ▷ Isolate low frequencies
7:   f_low ← f_shift × mask
8:   f_high ← f_shift × (1 − mask)
9:   low_inverse_shift ← IFFTSHIFT(f_low)
10:  low_img_back ← IFFT2(low_inverse_shift)
11:  high_inverse_shift ← IFFTSHIFT(f_high)
12:  high_img_back ← IFFT2(high_inverse_shift)
13:  low_freq_image ← TOIMAGE(low_img_back)
14:  high_freq_image ← TOIMAGE(high_img_back)
15:  return low_freq_image, high_freq_image
16: end function

```

The Dynamically Adaptive Density Control Strategy plays a pivotal role in optimizing the Gaussian Splatting process by adapting the density control mechanism throughout the training phases. This strategy intelligently mitigates the issues of excessive Clone and Split operations, thereby preventing the unnecessary proliferation of Light Cone Sample Points (LCSPs) and enhancing the efficiency of scene reconstruction. Below is the pseudocode that encapsulates the essence of this adaptive strategy:

Algorithm 3 Dynamically adaptive density control.

```

1: function ADAPTIVEDENSITYCONTROL(P, L,  $\alpha_1$ ,  $\beta_1$ )
2:    $f_1 \leftarrow \frac{L}{(\alpha_1 \times L + \beta_1)^2}$                                 ▷ Calculate Split weight
3:    $f_2 \leftarrow L \times (\alpha_1 \times L + \beta_1)^2$                     ▷ Calculate Clone weight
4:   if  $f_1 > 1$  then
5:     PERFORMCLONEOPERATION(P)
6:   end if
7:   if  $f_2 > 1$  then
8:     PERFORMSPLITOPERATION(P)
9:   end if
10: end function

```

Due to lab confidentiality and project phase, the full source code will be shared three months post-article acceptance at <https://www.github.com/FI3GS/Frequency-Importance-Gaussian-Splatting>. For immediate needs, please contact us for potential access to specific code or files. Further details and the complete codebase can be found in the supplementary material.

Data Availability The datasets used in this study include the Mip-NeRF360 dataset, available at http://storage.googleapis.com/gresearch/refraw360/360_v2.zip, the Deep Blending dataset, accessible at <http://www-sop.inria.fr/revs/publis/2018/HPPFDB18/datasets.html>, the Tanks & temples dataset, found at <https://>

drive.google.com/file/d/11KRfN91W1AxAW6lOFs4EeYDbeoQZCi87/view?usp=sharing, and the Synthetic NeRF dataset, located in the nerf_synthetic folder at https://drive.google.com/drive/folders/128yBriWlIG_3Nj5Rp7APSTZslqddJdfc1. Original experimental data for this study can be obtained from the corresponding author upon reasonable request after the project's completion.

Declarations

Conflicts of interest The authors declare no financial or personal conflicts of interest related to this research. The publication of this study aims to disseminate scientific knowledge, and all results are based on objective experiments and analyses. We assure the integrity and transparency of the writing and research process of this paper, adhering to ethical guidelines

References


1. Ahn I, Kim C (2013) A novel depth-based virtual view synthesis method for free viewpoint video. *IEEE Trans Broadcast* 59(4):614–626
2. Avidan S, Shashua A (1997) Novel view synthesis in tensor space. In: *Proceedings of IEEE computer society conference on computer vision and pattern recognition*, pp 1034–1040. <https://doi.org/10.1109/CVPR.1997.609457>
3. Avidan S, Shashua A (1998) Novel view synthesis by cascading trilinear tensors. *IEEE Trans Vis Comput Graph* a(a):293–306
4. Barron JT, Mildenhall B, Tancik M, et al (2021) Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In: *Proceedings of the IEEE/CVF international conference on computer vision*, pp 5855–5864
5. Barron JT, Mildenhall B, Verbin D, et al (2022) Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp 5470–5479
6. Chen A, Xu Z, Geiger A et al (2022) Tensorf: Tensorial radiance fields. In: Avidan S, Brostow G, Cissé M et al (eds) *Computer Vision - ECCV 2022*. Springer Nature Switzerland, Cham, pp 333–350
7. Drebin RA, Carpenter L, Hanrahan P (1988) Volume rendering. *ACM Siggraph Comput Graph* 22(4):65–74
8. Du Y, Zhang Y, Yu HX, et al (2021) Neural radiance flow for 4d view synthesis and video processing. In: *2021 IEEE/CVF international conference on computer vision (ICCV)*, IEEE Computer Society, pp 14304–14314
9. Fridovich-Keil S, Yu A, Tancik M, et al (2022) Plenoxels: Radiance fields without neural networks. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp 5501–5510
10. Hedman P, Philip J, Price T et al (2018) Deep blending for free-viewpoint image-based rendering. *ACM Trans Graph (ToG)* 37(6):1–15
11. Jain A, Mildenhall B, Barron JT, et al (2022) Zero-shot text-guided object generation with dream fields. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp 867–876
12. Karnewar A, Ritschel T, Wang O, et al (2022) Relu fields: The little non-linearity that could. In: *ACM SIGGRAPH 2022 conference proceedings*, pp 1–9
13. Kerbl B, Kopanas G, Leimkühler T et al (2023) 3d gaussian splatting for real-time radiance field rendering. *ACM Trans Graph (ToG)* 42(4):1–14
14. Knapitsch A, Park J, Zhou QY, et al (2017) Tanks and temples: Benchmarking large-scale scene reconstruction. *ACM Trans Graph* 36(4)
15. Kobayashi S, Matsumoto E, Sitzmann V (2022) Decomposing nerf for editing via feature field distillation. In: Koyejo S, Mohamed S, Agarwal A, et al (eds) *Advances in neural information processing systems*, vol 35. Curran Associates, Inc., pp 23311–23330. https://proceedings.neurips.cc/paper_files/paper/2022/file/93f250215e4889119807b6fac3a57aec-Paper-Conference.pdf
16. Li L, Shen Z, Wang Z et al (2022) Streaming radiance fields for 3d video synthesis. *Adv Neural Inf Process* 35:13485–13498
17. Li Z, Niklaus S, Snavely N, et al (2021) Neural scene flow fields for space-time view synthesis of dynamic scenes. In: *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR)*, pp 6498–6508

18. Lin CH, Ma WC, Torralba A, et al (2021) Barf: Bundle-adjusting neural radiance fields. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 5741–5751
19. Martin-Brualla R, Radwan N, Sajjadi MS, et al (2021) Nerf in the wild: Neural radiance fields for unconstrained photo collections. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 7210–7219
20. Max N (1995) Optical models for direct volume rendering. *IEEE Trans Vis Comput* 1(2):99–108
21. Mildenhall B, Srinivasan PP, Tancik M et al (2021) Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM* 65(1):99–106
22. Mildenhall B, Hedman P, Martin-Brualla R, et al (2022) Nerf in the dark: High dynamic range view synthesis from noisy raw images. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 16190–16199
23. Müller T, Evans A, Schied C et al (2022) Instant neural graphics primitives with a multiresolution hash encoding. *ACM Trans Graph (ToG)* 41(4):1–15
24. Neff T, Stadlbauer P, Parger M, et al (2021) Donerf: Towards real-time rendering of compact neural radiance fields using depth oracle networks. In: *Computer graphics Forum*, Wiley Online Library, pp 45–59
25. Park K, Sinha U, Barron JT, et al (2021) Nerfies: Deformable neural radiance fields. In: Proceedings of the IEEE/CVF International Conference on Computer Vision, pp 5865–5874
26. Poole B, Jain A, Barron JT, et al (2022) Dreamfusion: Text-to-3d using 2d diffusion. [arXiv:2209.14988](https://arxiv.org/abs/2209.14988)
27. Pumarola A, Corona E, Pons-Moll G, et al (2021) D-nerf: Neural radiance fields for dynamic scenes. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp 10318–10327
28. Rebin D, Jiang W, Yazdani S, et al (2021) Derf: Decomposed radiance fields. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 14153–14161
29. Ren L, Pfister H, Zwicker M (2002) Object space ewa surface splatting: A hardware accelerated approach to high quality point rendering. In: *Computer graphics forum*, Wiley Online Library, pp 461–470
30. Schonberger JL, Frahm JM (2016) Structure-from-motion revisited. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 4104–4113
31. Sun C, Sun M, Chen HT (2022) Direct voxel grid optimization: Super-fast convergence for radiance fields reconstruction. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 5459–5469
32. Taud H, Mas J (2018) Multilayer perceptron (mlp). *Geomatic approaches for modeling land change scenarios*, pp 451–455
33. Verbin D, Hedman P, Mildenhall B, et al (2022) Ref-nerf: Structured view-dependent appearance for neural radiance fields. In: 2022 IEEE/CVF conference on computer vision and pattern recognition (CVPR), IEEE, pp 5481–5490
34. Wang Y, Yang S, Hu Y, et al (2022) Nerfocus: Neural radiance field for 3d synthetic defocus. [arXiv:2203.05189](https://arxiv.org/abs/2203.05189)
35. Xian W, Huang JB, Kopf J, et al (2021) Space-time neural irradiance fields for free-viewpoint video. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 9421–9431
36. Xu ZQJ, Zhang Y, Luo T, et al (2019) Frequency principle: Fourier analysis sheds light on deep neural networks. [arXiv:1901.06523](https://arxiv.org/abs/1901.06523)
37. Yang J, Pavone M, Wang Y (2023) Freenerf: Improving few-shot neural rendering with free frequency regularization. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 8254–8263
38. Yen-Chen L, Florence P, Barron JT, et al (2022) Nerf-supervision: Learning dense object descriptors from neural radiance fields. In: 2022 International conference on robotics and automation (ICRA), pp 6496–6503. <https://doi.org/10.1109/ICRA46639.2022.9812291>

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.

Authors and Affiliations

Lizhe Chen¹ · Yan Hu¹ · Yu Zhang¹ · Yuyao Ge^{1,2,3} · Haoyu Zhang¹ ·
Xingquan Cai¹ 

Lizhe Chen
chenlizhe@mail.ncut.edu.cn

Yan Hu
Anita.Hu@ncut.edu.cn

Yu Zhang
zhangyu@mail.ncut.edu.cn

Yuyao Ge
yuyao.ge.work@gmail.com

Haoyu Zhang
emailzhanghaoyu@163.com

¹ College of Information, North China University of Technology, Beijing 100144, China

² Institute of Computing Technology, Chinese Academy of Sciences, Beijing 100190, China

³ University of Chinese Academy of Sciences, Beijing 101408, China