GaussianSR: High Fidelity 2D Gaussian Splatting for Arbitrary-Scale Image Super-Resolution

Jintong Hu[1*], **Bin Xia**[2*], **Bin Chen**[3*], **Wenming Yang**[1†], **Lei Zhang**[4, 5]

[1]Tsinghua University
[2]The Chinese University of Hong Kong
[3]Peking University
[4]The Hong Kong Polytechnic University
[5]OPPO Research Institute
hujt22@mails.tsinghua.edu.cn, zjbinxia@gmail.com, chenbin@stu.pku.edu.cn
yang.wenming@sz.tsinghua.edu.cn, john.zhang@polyu.edu.hk

Abstract

Implicit neural representations (INRs) have significantly advanced the field of arbitrary-scale super-resolution (ASSR) of images. Most existing INR-based ASSR networks first extract features from the given low-resolution image using an encoder, and then render the super-resolved result via a multi-layer perceptron decoder. Although these approaches have shown promising results, their performance is constrained by the limited representation ability of discrete latent codes in the encoded features. In this paper, we propose a novel ASSR method named GaussianSR that overcomes this limitation through 2D Gaussian Splatting (2DGS). Unlike traditional methods that treat pixels as discrete points, GaussianSR represents each pixel as a continuous Gaussian field. The encoded features are simultaneously refined and upsampled by rendering the mutually stacked Gaussian fields. As a result, long-range dependencies are established to enhance representation ability. In addition, a classifier is developed to dynamically assign Gaussian kernels to all pixels to further improve flexibility. All components of GaussianSR (i.e., encoder, classifier, Gaussian kernels, and decoder) are jointly learned end-to-end. Experiments demonstrate that GaussianSR achieves superior ASSR performance with fewer parameters than existing methods while enjoying interpretable and content-aware feature aggregations.

Introduction

The vision world is continuous, presenting scenes and objects in their natural, uninterrupted forms. However, computer vision tasks predominantly employ pixel-based discrete representations for image processing, which inherently constrains the ability capture the continuous nature with high fidelity across different resolutions. The emergence of implicit neural representations (INR) has revolutionized this challenge by treating images as continuous functions. Pioneered by LIIF (Chen, Liu, and Wang 2021), INR-based methods leverage Multi-Layer Perceptrons (MLPs) to model the RGB values of high-resolution (HR) images as continuous functions of low-resolution (LR) features and pixel coordinates. This continuous representation enables LIIF to perform super-resolution at arbitrary scales without the need for creating and training separate models for each scaling factor. Building upon LIIF, numerous variants and extensions have subsequently emerged (Lee and Jin 2022; Xu, Wang, and Shi 2022; Yang et al. 2021; Cao et al. 2023), further advancing the capabilities of INR-based Arbitrary-Scale Super-Resolution (ASSR).

Although INR-based ASSR methods have demonstrated their effectiveness, several intrinsic limitations remain. Latent codes are stored individually, which necessitates the use of Multi-layer Perceptrons (MLPs) to interpolate these discrete features into a continuous output. This process may not fully preserve key physical properties of the scene, such as lighting and texture consistency, reducing the overall interpretability and physical fidelity of the generated images. Moreover, as depicted in Figure 1(a), when query location x_q moves in 2D domain, the selection of z_t can abruptly switch from one to another if x_q crosses the dashed lines, potentially leading to checkerboard artifacts in the output image. To counteract this, a strategy involving a weighted combination of nearby latent codes is employed, which, while effective, increases the computational load and may become a bottleneck for real-time applications.

How about continuously storing latent codes? After discussing the limitations of existing INR-based methods in handling 2D image super-resolution (SR) with continuous integrity, we explore the potential application of concepts derived from 3D reconstruction technologies to address these issues. Inspired by the recent advancements in 3D Gaussian Splatting (3DGS) for detailed scene rendering and object reconstruction (Kerbl et al. 2023; Keetha et al. 2024; Yan et al. 2024; Matsuki et al. 2024), we propose a method that adapts this technology to more efficiently handle continuous data representation. The key concept of 3DGS involves representing objects as 3D Gaussian fields, allowing for the construction of continuous structures with significantly reduced parameter requirements. By applying this method, overlapped Gaussian spheres can form vari-

*These authors contributed equally.
†Corresponding author.

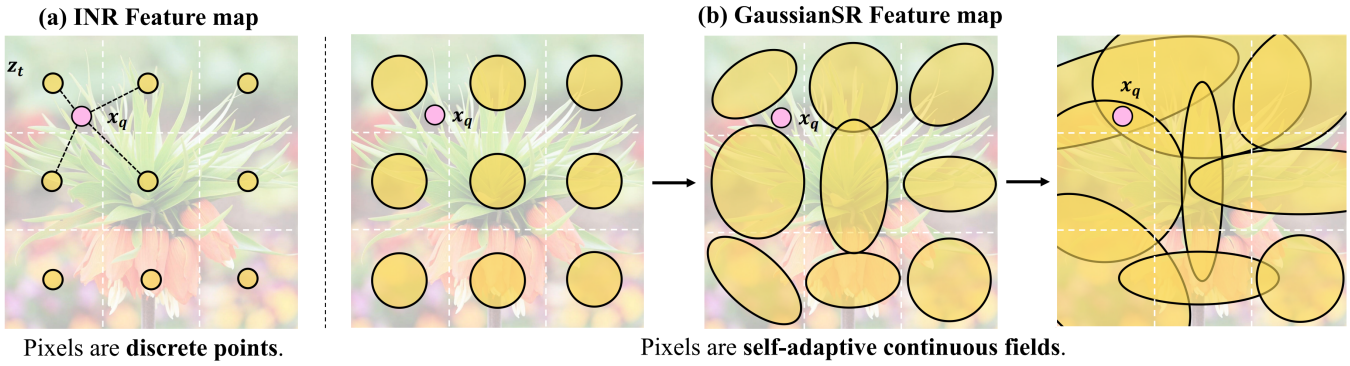

Figure 1: **Comparison of Feature Storage between INR-based ASSR and our GaussianSR.** INR methods treat pixels as discrete points. Instead, our GaussianSR method models each pixel as a continuous Gaussian field. By representing pixels as continuous fields instead of discrete points, GaussianSR can explicitly represent the field values at any position (e.g x_q). GaussianSR achieves arbitrary-scale upsampling in a more elegant and natural way.

ous detailed shapes, offering a potential pathway to achieve higher fidelity in 2D image SR while maintaining computational efficiency.

In this paper, we introduce a paradigm shift for ASSR by proposing the GaussianSR pipeline, which utilizes 2D Gaussian Splatting (2DGS) to overcome the inherent discontinuity of pixel-based INR methods. GaussianSR is built upon the insight that pixel values intrinsically exhibit intensity variations which can be more accurately captured through a continuous Gaussian representation. As illustrated in Figure 1(b), each pixel under our methodology is modeled as a self-adaptive continuous field instead of a single value. This continuous representation naturally and explicitly yields the field value at any query position x_q, avoiding the information loss associated with discrete representations.

Specifically, our method enables the dynamic adaptation of kernels to match varied input qualities effectively. We train a classifier that assigns a Gaussian kernel to each input pixel. Rather than applying a global kernel or a fixed set of kernels, the classifier can tailor the kernel for each pixel based on its specific characteristics, resulting in a more adaptive and context-aware processing of the input data. Moreover, our technique overcomes the limitations of fixed receptive fields inherent in traditional INR-based ASSR methods by employing flexible and adaptable Gaussian kernels, which can be stacked together. This flexibility not only enables the model to capture features across multiple scales but also bolsters its capacity for representing complex, enhanced features, resulting in superior performance in SR tasks. Our efforts have successfully built a bridge between Gaussian representations and image restoration, marking a pioneering step in the field of ASSR. In summary, the contribution of this paper can be summarized in three parts:

- We pioneer building a novel paradigm for ASSR through the 2D Gaussian Splatting representation, namely GaussianSR. Our approach builds a bridge between discrete and continuous feature representation by turning pixel point into Gaussian field.
- We train a classifier that adaptively assigns the appropriate 2D Gaussian kernel to each pixel, which facilitates the adaptive matching of diverse input characteristics with suitable Gaussian kernels, effectively accommodating a wide range of input variations.
- Extensive experiments demonstrate that seamlessly transitioning existing INR-based ASSR to our GaussianSR framework can achieve better performance with reduced parameter, underscoring the efficacy and great capability of our proposed approach.

Related Work

Implicit Neural Representation

Implicit Neural Representations (INRs) elegantly model signals across a continuous domain and have gained prominence in numerous fields, including 3D reconstruction (Park et al. 2019; Chen and Zhang 2019; Saito et al. 2019), scene rendering (Park et al. 2021a; Sitzmann, Zollhöfer, and Wetzstein 2019), robotics (Chen et al. 2021a; Li et al. 2021), and image and video compression (Chen et al. 2021b; Zhang et al. 2022). Notably, NeRF (Mildenhall et al. 2020) innovatively represents complex 3D scenes with continuous neural implicit functions and has enabled novel view synthesis, although it faces challenges with dynamic scenes due to its static nature and lacks explicit geometric representations. Attempting to address dynamic content, Deformable Neural Radiance Fields (Park et al. 2021a) have introduced a warping mechanism to the static NeRF, and Neural Sparse Voxel Fields (Liu et al. 2020) have coupled implicit functions with sparse voxel grids to enhance efficiency and facilitate geometric interpretability. Despite the impressive outcomes of these methods, their discrete storage of each latent code or feature often misses the opportunity to encapsulate the intricacies or characteristics inherent to the signals in question. PixelNeRF (Yu et al. 2021) endeavors to overcome this by tying pixel-level features to the implicit representation, while HyperNeRF (Park et al. 2021b) utilizes hypernetworks to augment the neural encoder's capabilities, potentially allowing for a more nuanced capture of intrinsic properties.

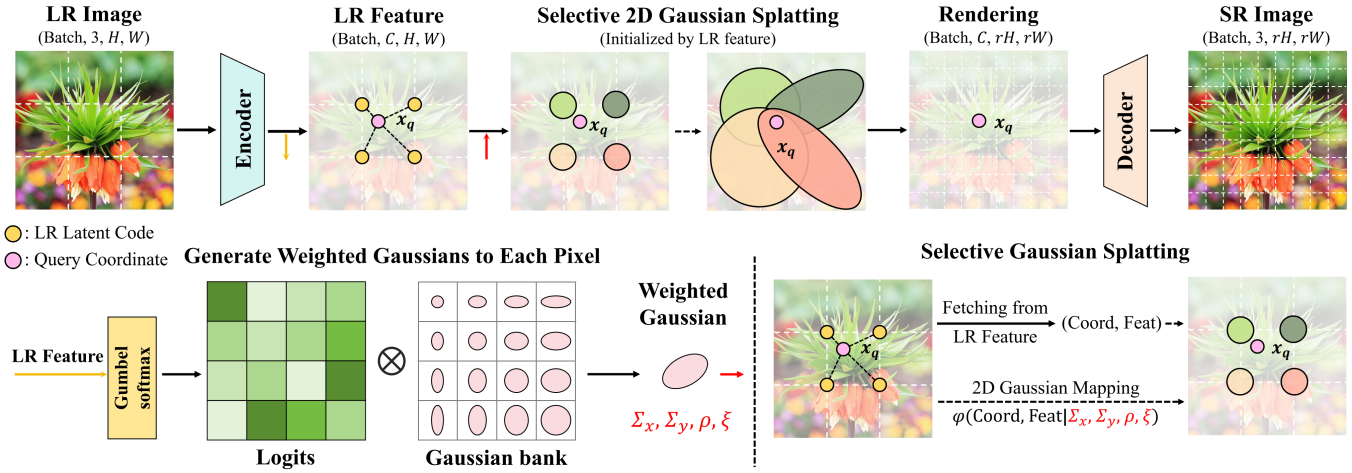

Figure 2: **The main pipeline of GaussianSR.** GaussianSR begins with an encoder that extracts features from the input image, followed by Selective Gaussian Splatting which assigns a learnable Gaussian kernel to each pixel, converting discrete feature points into Gaussian fields. Features at arbitrary query point x_q in the plane are computed using the overlapping Gaussian functions that modulate their influence based on the spatial location. Finally, these continuous-domain features are rendered into a high-resolution space and refined through a decoder to reconstruct the desired RGB output at the specified query coordinates.

Arbitrary-Scale Super-Resolution

Traditional single image super-resolution (SISR) techniques, such as interpolation, have been outperformed by deep learning-based methods, due to their ability to learn hierarchical features and map LR to HR images. Various SR architectures have been proposed, including sub-pixel convolutional layers, residual blocks, dense connections, diffusion process, and state space models, etc.(Shi et al. 2016; Lim et al. 2017; Zhang et al. 2018; Chen et al. 2024; Li et al. 2024; Liu et al. 2024; Chen and Zhang 2024) However, these methods lack flexibility as they are designed for fixed integer-scale upsampling, limiting their applicability.

Arbitrary-scale super-resolution (ASSR) techniques have gained prominence for their adaptability across various scaling factors. MetaSR (Hu et al. 2019) pioneered CNN-based ASSR, while LIIF (Chen, Liu, and Wang 2021) introduced an innovative framework leveraging implicit neural representations to treat images as continuous functions. Subsequent studies, such as LTE (Lee and Jin 2022), SADN (Wu, Ni, and Zhang 2023), and A-LIIF (Li et al. 2022), have aimed to address limitations like spectral bias, multi-scale feature integration, and artifact mitigation. However, these methods treat all features discretely without accounting for inherent variations, lack adaptive receptive fields to naturally and explicitly handle diverse-scale inputs, and rely on generic kernel weights independent of individual samples during inference.

Model Architecture

Preliminary: 3D Gaussian Splatting

3D Gaussian splatting (3DGS) (Kerbl et al. 2023) represents a paradigmatic shift from the prevalent neural radiance fields (NeRF) (Mildenhall et al. 2020) methodology for scene representation and rendering. Grounded in a fundamentally distinct approach, 3DGS circumvents the computational complexities and controllability limitations inherent to NeRF while retaining the capability to synthesize photorealistic novel views from sparse input data. The forward process of 3DGS can be summarized as: • **3D Gaussian Representation**. The scene is represented using a collection of 3D Gaussians, each characterized by learnable properties such as position, opacity, covariance matrix, and color. This explicit representation allows efficient rendering through parallelized workflows. • **Splatting and Tiling**. The 3D Gaussians are first projected onto the 2D image plane through a splatting process. The image is then divided into non-overlapping patches or "tiles" to facilitate parallel computation. • **Sorted Gaussian Rendering**. The projected Gaussians are sorted by depth within each tile, and the final pixel color is computed by alpha compositing, leveraging the sorted order. The impact of a 3D Gaussian i on an arbitrary 3D point p in 3D is defined as follows:

$$f_i(p) = \sigma(\alpha_i) \exp(-\frac{1}{2}(p-\mu_i)\Sigma_i^{-1}(p-\mu_i)) \quad (1)$$

where p represents an arbitrary point in a 3D Cartesian coordinate system. The 3D Gaussian i is parametrized by (1) mean μ_i, (2) covariance Σ_i, (3) opacity $\sigma(\alpha_i)$, (4) color parameters c_i, either 3 values for (R, G, B) or spherical harmonics coefficients. The image formation model of Gaussian splatting can be formulated as:

$$C_{3DGS}(p) = \sum_{i \in N} c_i f_i^{2D}(p) \prod_{j=1}^{i-1}(1 - f_j^{2D}(p)) \quad (2)$$

where f_i^{2D} is a projection of $f_i(p)$ into 2D, *i.e.*, onto an image plane of the camera that is being rendered.

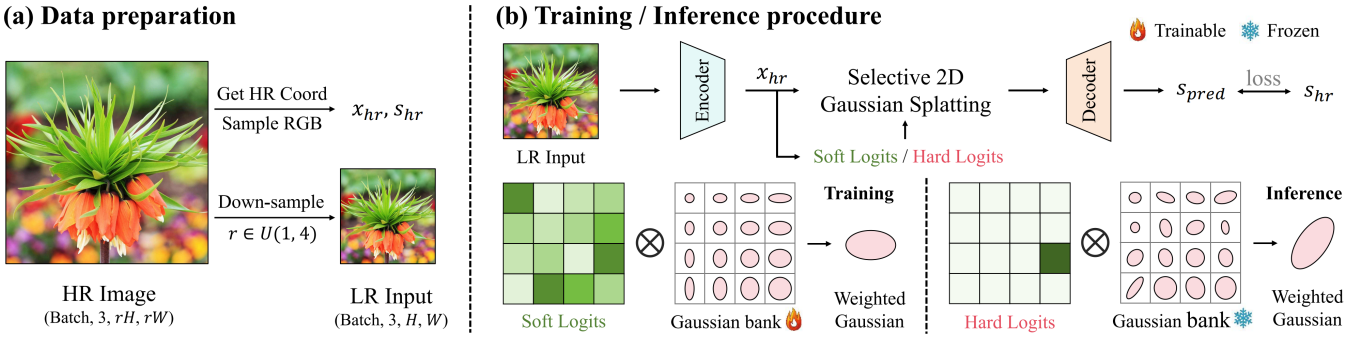

Figure 3: **Training and Inference Process of GaussianSR**. GaussianSR employs the Selective Gaussian Splatting (SGS) module, which adaptively assigns Gaussian kernels to pixels based on their distinctive features. During training, SGS leverages the Gumbel Softmax to generate soft labels, enabling gradient backpropagation and parameter optimization for both logits and the Gaussian bank (which stores standard deviations and opacities). In the inference phase, SGS switches to hard labels, selecting the most likely Gaussian kernel for each pixel based on the optimized parameters.

2D Gaussian Splatting

Although 3DGS (Kerbl et al. 2023) has shown remarkable performance in various 3D tasks, its inherent 3D formulation may not be optimally suited for certain 2D tasks such as image restoration. Motivated by the need for continuous representation, we propose a 2D adaptation of Gaussian splatting, termed 2D Gaussian Splatting (2DGS). Our 2DGS simplifies the approach compared to 3DGS by eliminating operations and parameters specific to 3D scene rendering, such as project transformations and spherical harmonics,

The process of 2DGS consists of two main steps: • **Grid Transformation**. Initially, we create an affine-transformed grid that takes into account the standard deviations of all Gaussian kernels. This grid is tailored to match the high resolution (HR) dimensions of the target output, preparing the scene for subsequent feature mapping. • **Feature Value Rendering**. Following the grid setup, the feature values undergo alpha blending, and then these blended values are projected onto the transformed grid for rendering. This step ensures that the upsampled image preserves natural continuity and visual integrity.

In our framework, the image representation unit is represented as a 2D Gaussian defined by its position $\mu_i \in \mathbb{R}^2$ and the covariance matrix $\mathbf{\Sigma}_i = ([\Sigma_{xi}, \rho_i], [\rho_i, \Sigma_{yi}]) \in \mathbb{R}^{2\times2}$. To prevent Σ_x, Σ_y from being negative, we apply the sigmoid function for activation. The Gaussian field is represented as:

$$f_i(p|\mu_i, \mathbf{\Sigma}_i) = \frac{1}{2\pi\,|\mathbf{\Sigma_i}|} e^{-\frac{1}{2}(p-\mu_i)^{\mathsf{T}}\mathbf{\Sigma_i}^{-1}(p-\mu_i)} \tag{3}$$

where p is an arbitrary coordinate in $\mathbb{R}^2$ space. In situations where multiple Gaussian fields overlap, we use alpha blending to combine their feature values $\mathbf{v}_i \in \mathbb{R}^k$, modulated by corresponding opacity $\xi_i \in \mathbb{R}$:

$$c_i = \sigma(\xi) \cdot \mathbf{v}_i \tag{4}$$

Based on the relative distance between the query coordinate p and the center of each Gaussian field, we can determine the transmittance value $f_i(p|\mu_i, \mathbf{\Sigma}_i)$, which is then used to calculate a weighted sum with c_i, allowing us to render the value at p.

Arbitrary-Scale Super-Resolution Pipeline

The shift from a discrete feature storage methodology to a continuous feature field, characterized by a Gaussian distribution, represents a significant advancement in the field of image restoration. This transformation enables any point within the feature field to be explicitly determined according to the Gaussian distribution, thereby facilitating a more natural implementation of ASSR. Based on 2DGS, we have designed a novel image SR framework called GaussianSR, which introduces Gaussian representation for the first time in image restoration tasks. In this section, we discuss the key components of the GaussianSR framework.

Overall Architecture. The architecture of GaussianSR is illustrated in Figure 2. The process begins with an encoder that extracts features from the input images. These features are then mapped onto a learnable Gaussian field via Selective Gaussian Splatting, assigning a Gaussian distribution to each pixel. At any given query coordinate x_q, multiple overlapping Gaussian fields determine the feature value through their collective Gaussian functions explicitly. The features are subsequently rendered into the high-resolution space and processed through several fully connected layers to restore the channel dimension and produce the RGB value at x_q. Since x_q can be any point within the $\mathbb{R}^2$ space, the framework achieves arbitrary-scale super-resolution.

LR Initialization. LR feature initialization involves fetching features from the input LR image and representing them as initialization points within a continuous Gaussian feature field. Specifically, the value of the LR feature is used as the initial amplitude of the Gaussian, while the coordinates of the LR feature determine the center of the Gaussian field. Mathematically, let $(m_i, n_i, \mathbf{v}_i)$ be the set of LR feature coordinates and their corresponding values, where $i = 1, 2, \cdots, N$, and N is the total number of features. Each LR feature is then represented as a Gaussian distribution $f_i(p|\mu_i, \mathbf{\Sigma}_i)$ within the continuous feature field, with its mean $\mu_i = (m_i, n_i)$ corresponding to the normalized LR feature coordinates and its amplitude initialized to $\mathbf{v}_i$, the actual value of the feature of the LR image. The feature value

at any coordinate p within the continuous Gaussian feature field is determined by summing the function values from all individual Gaussian distributions centered at the LR feature coordinates:

$$C_{2DGS}(p|\mathbf{v}, \mu, \mathbf{\Sigma}, \xi) = \sum_i f_i(p|\mu_i, \mathbf{\Sigma}_i) \cdot c_i \quad (5)$$

where $f_i(\cdot), c_i(\cdot)$ are the same as defined in Equations 3 amd 4. LR feature initialization ensures that subsequent upsampling and reconstruction processes can effectively leverage the spatial and intensity information from the LR image, ultimately leading to high-quality HR image reconstruction.

Selective Gaussian Splatting. Convolutional operations, while effective, are inherently limited by their fixed kernel sizes, incapable of adapting to varying input characteristics. Moreover, high-resolution LR images can induce redundancy among Gaussian kernels due to similar standard deviations and opacities. To address these limitations, we propose the Selective Gaussian Splatting (SGS) technique, which adaptively assigns Gaussian kernels to pixels based on their unique features, minimizing kernel redundancy and parameter overhead. SGS leverages logits from an encoder corresponding to 100 distinct classes per pixel, each associated with a Gaussian kernel differentiated by standard deviation and opacity. Initially varied in shape and transparency (refer to Figure 3), these kernels are optimized during training, aligning with encoder logits to generate customized Gaussian distributions for each pixel location. Gumbel Softmax facilitates gradient backpropagation with soft labels during training and hard labels during inference, enhancing optimization while maintaining interpretability.

Dual-Stream Feature Decoupling. Addressing the issues of high memory consumption inherent in Gaussian splatting methodologies, we introduce the Dual-Stream Feature Decoupling architecture, specifically aimed at enhancing super-resolution processes while maintaining parameter efficiency. This approach strategically decouples encoded features along the channel dimension into two tensors with reduced channels but preserved spatial resolution. For one tensor, an additional feature unfolding step is performed before Gaussian splatting. This unfolding operation rearranges the input tensor into a lower-resolution representation, effectively reducing the memory consumption for the subsequent Gaussian splatting operation. After Gaussian splatting for detail preservation, a corresponding feature folding operation restores the original spatial dimensions. The other tensor stream is bicubically upsampled for efficiency. The augmented outputs from these two parallel streams are then fused and refined through the decoder, producing the high-resolution RGB output.

Experiments

Datasets

For our training, we use the DIV2K dataset (Agustsson and Timofte 2017). Sourced from the NTIRE challenge (Radu Timofte and Zhang. 2017), this dataset comprises 1000 diverse 2K-resolution images featuring a wide range of content, including individuals, urban scenes, flora, fauna and natural landscapes. Within this collection, we allocate 800 images for the training set, 100 images for validation. To evaluate the generalization performance of the model, we report the results on the DIV2K validation set with 100 images. Another four benchmark datasets are also utilized, namely General100 (Dong, Loy, and Tang 2016), BSD100 (Martin et al. 2001), Urban100 (Huang, Singh, and Ahuja 2015), and Manga109, (Matsui et al. 2016) which provided a comprehensive landscape for assessing cross-dataset robustness. Consistent with previous work (Chen, Liu, and Wang 2021; Hu et al. 2019; Yang et al. 2021), we utilized bicubic downsampling to synthesize the LR images.

Implementation Details

To simulate a continuous magnification process, the downsampling factor is randomly sampled from a uniform distribution, U(1, 4). To accommodate the GaussianSR framework, within each batch, the downsampling factor remains constant across different GPUs (if multi-GPU training is employed). The loss function used is the L_1 distance between the reconstructed image and the ground truth image. Following the settings in LIIF and its variants (Chen, Liu, and Wang 2021; Li et al. 2022; Yang et al. 2021), we randomly crop the LR images into 48×48 patches, collect 2304 random pixels on the HR images. The initial learning rate for all modules is set to 1e-4, and is halved every 200 epochs. The model is trained in parallel on 4 Tesla V100 GPUs with a mini-batch size of 16. The training process takes about 2000 epochs to converge.

Quantitative and Qualitative Results

To validate the efficacy of our GaussianSR, we conduct a comparative analysis against several advanced methods, including MetaSR (Hu et al. 2019), LIIF (Chen, Liu, and Wang 2021), and its variants (ITSRN (Yang et al. 2021), A-LIIF (Li et al. 2022), DIINN (Nguyen and Beksi 2023)). All models are re-trained under a unified framework for fairness. Due to the different framework of CiaoSR (Cao et al. 2023), it is excluded from our analysis. Results in Tables 1 and 2 show that our method achieves competitive performance across five datasets for various scaling factors, particularly excelling in non-integer scaling factors due to the flexible Gaussian representation. While our model performs similarly to LIIF and A-LIIF on lower-resolution datasets like General100 and BSD100, it significantly outperforms them on higher-resolution datasets such as Urban100 and Manga109, with especially notable results on Urban100.

Figure 4 offers a qualitative comparison with other arbitrary-scale SR methods. Our model excels at synthesizing SR images with sharper textures. For instance, in the second column of the first row and the second column of the second row, our method successfully recovers the texture of buildings despite significant texture degradation in the low-resolution images. In contrast, other methods tend to produce erroneous redundant artifacts that detract from the visual quality of the image due to limitations in local integration and insufficient feature extraction. More visual comparisons can be found in the supplementary materials.

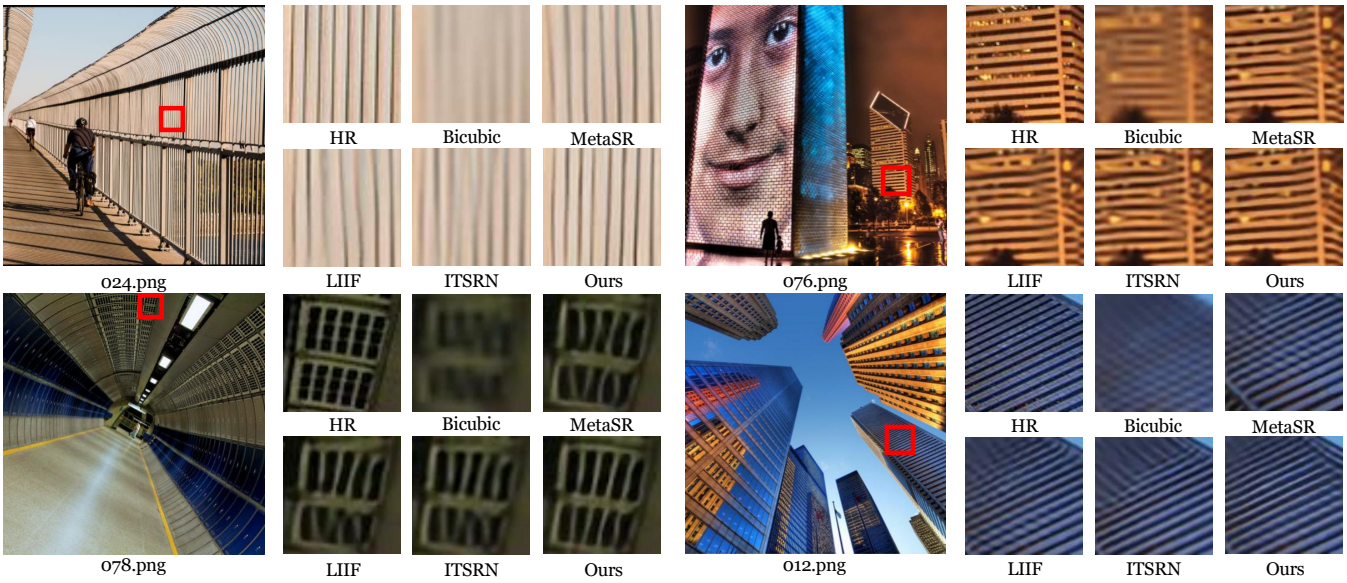

Figure 4: **Qualitative comparison for ×4 SR to other ASSR methods on Urban100 (Huang, Singh, and Ahuja 2015) dataset**. EDSR-baseline (Lim et al. 2017) is used as an encoder for all methods. For all the shown examples, our method significantly outperforms other methods, particularly in the image rich in repeated textures and structures.

Methods	General100			BSD100			Urban100			Manga109			DIV2K100		
	×2	×3	×4	×2	×3	×4	×2	×3	×4	×2	×3	×4	×2	×3	×4
Bicubic	32.14	28.56	26.58	28.25	25.96	24.69	25.68	23.07	21.77	29.98	25.68	23.52	31.45	28.42	26.81
EDSR-baseline	38.21	33.93	31.42	32.16	29.09	27.57	31.98	28.15	26.04	38.54	33.45	30.35	34.55	30.90	28.94
EDSR-baseline-MetaSR	38.22	33.93	31.40	32.16	29.09	27.55	32.08	28.12	25.95	38.53	33.51	30.37	34.64	30.93	28.92
EDSR-baseline-LIIF	38.25	33.97	31.53	32.17	29.10	27.60	32.15	28.22	26.15	38.63	33.47	30.54	34.67	30.96	29.00
EDSR-baseline-ITSRN	38.25	33.95	31.48	32.18	29.10	27.58	32.13	28.14	26.06	38.58	33.47	30.47	34.67	30.93	28.97
EDSR-baseline-ALIIF	38.21	33.95	31.48	32.18	29.11	27.60	32.09	28.19	26.14	38.53	33.42	30.47	34.65	30.95	28.99
EDSR-baseline-DIINN†	-	-	-	30.69	27.73	26.22	30.29	26.46	24.49	-	-	-	34.63	30.93	28.98
EDSR-baseline-GaussianSR	**38.31**	**34.02**	**31.55**	**32.20**	**29.13**	**27.61**	**32.25**	**28.28**	**26.19**	**38.64**	**33.57**	**30.54**	**34.71**	**31.00**	**29.03**

Table 1: Quantitative comparison for integer-scale super-resolution on other benchmarks (PSNR (dB)). Throughout the following tables, the best results for each case are in **bold**. † As the DIINN method has not been open-sourced, we directly use the results reported in their article.

Methods	General100			BSD100			Urban100			Manga109			DIV2K100		
	×1.5	×2.4	×3.6	×1.5	×2.4	×3.6	×1.5	×2.4	×3.6	×1.5	×2.4	×3.6	×1.5	×2.4	×3.6
Bicubic	34.89	30.12	27.17	30.78	27.09	25.11	27.92	24.25	22.21	33.12	27.50	24.15	34.00	29.75	27.31
EDSR-baseline†	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
EDSR-baseline-MetaSR	42.16	36.14	32.30	35.69	30.60	28.08	36.05	30.07	26.74	42.67	36.18	31.56	38.57	32.78	29.62
EDSR-baseline-LIIF	42.20	36.18	32.37	35.69	30.62	28.11	36.13	30.16	26.89	42.67	36.19	31.59	38.57	32.82	29.67
EDSR-baseline-ITSRN	42.24	36.18	32.34	35.70	30.62	28.10	36.14	30.12	26.81	42.68	36.18	31.55	38.61	32.80	29.64
EDSR-baseline-ALIIF	42.14	36.16	32.33	35.67	30.60	28.10	36.02	30.07	26.84	42.56	36.11	31.53	38.54	32.79	29.65
EDSR-baseline-GaussianSR	**42.24**	**36.23**	**32.40**	**35.73**	**30.64**	**28.13**	**36.27**	**30.23**	**26.94**	**42.72**	**36.25**	**31.60**	**38.64**	**32.85**	**29.70**

Table 2: Quantitative comparison for arbitrary noninteger-scale super-resolution on other benchmarks (PSNR (dB)). † EDSR-baseline (Lim et al. 2017) cannot handle noninteger-scale super-resolution.

In Table 3, we present a comparative analysis of our method against LIIF and ITSRN at early stage (epoch 100). Our method consistently outperforms other methods on Urban100 and BSD100, underscoring its superior performance even before full training is completed. This shows that our approach is more effective in leveraging a limited amount of training samples to achieve better results. Notably, we ensure that LIIF and ITSRN are trained under the same framework for a fair comparison. In addition, we report the inference time of GaussianSR and LIIF with the same input and output sizes in Table 4, providing a comprehensive evaluation of their efficiency.

Target Resolution	(128×128)	(256×256)	(384×384)
Methods	Runtime (ms) ↓		
LIIF	14.52	43.18	**86.43**
ITSRN	21.55	58.01	120.69
GaussianSR	**12.56**	**41.70**	106.72

Table 3: Latency comparison in milliseconds. The input is a single RGB image of size 64×64. We report the average runtime over 100 runs on a Tesla V100 GPU.

Methods	Urban100			BSD100		
	×2	×3	×4	×2	×3	×4
LIIF	31.35	27.62	25.65	31.98	28.95	27.44
ITSRN	31.45	27.54	25.56	32.03	28.94	27.43
GaussianSR	**31.63**	**27.79**	**25.78**	**32.06**	**29.00**	**27.50**

Table 4: Comparative analysis of early stage performance between methods on Urban100 and BSD100.

Ablation Study

In this section, we conduct an ablation study to investigate the effects of different settings in our architecture. Table 5 and Table 6 present the results of various hyperparameter settings for channel decoupling and Gaussian bank.

Channel decoupling. The default configuration involves 8 channels undergoing 2DGS upsampling, while the remaining channels are upsampled using bicubic interpolation. This setup is designed to achieve memory-efficient inference. We compare this configuration with settings where all channels were upsampled using bicubic interpolation and where 16 channels were upsampled using 2DGS. When increasing the channel size from 8 to 16, the model begins to overfit to the training data, capturing irrelevant patterns instead of meaningful, generalizable features. The two streams are designed to complement each other: the Gaussian stream captures high-frequency details, while the bicubic stream provides a stable approximation of overall structure. Allocating too many channels to the Gaussian stream can overshadow the bicubic stream, disrupting the balance between them.

Gaussian bank. GaussianSR consists of 100 Gaussian fields initialized in a stepwise manner. We conduct ablation experiments by varying the number of Gaussian fields to 100, 400, and 900. Interestingly, the configuration with 100 Gaussians produces better results. We explore the frequency of selection for each Gaussian, showing that fewer than 80 Gaussian kernels were frequently selected. Thus, increasing the number of Gaussian fields is detrimental to training. Figure 5 visualizes the parameters within the Gaussian bank, showing that the majority of sigma values are significantly greater than zero at the end of training. This indicates that each pixel is represented by an ellipse, validating our approach of replacing points with fields.

Setting	DIV2K100		
	×2	×3	×4
Bicubic-64	34.4128	30.8311	28.8788
Bicubic-56, Gaussian-8	**34.7134**	**31.0010**	**29.0281**
Bicubic-48, Gaussian-16	34.6990	30.9853	29.0157

Table 5: Ablation study on channel proportion of 2DGS in Dual-Stream Decoupling on DIV2K100 (PSNR (dB)).

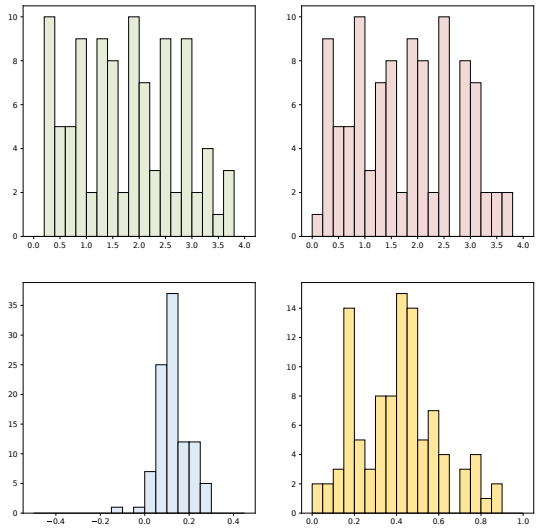

Figure 5: **Frequency distributions of Σ_x, Σ_y, ρ, and opacity in GaussianSR (top-down, left-right)**. Most of Σ values are significantly greater than zero, demonstrating that each pixel is represented by a Gaussian field instead of a point.

Classes of Gaussians	Scale Factor		
	×2	×3	×4
64	34.7070	30.9923	29.0196
100	**34.7134**	**31.0010**	**29.0281**
400	34.7041	30.9914	29.0174
900	34.7072	30.9906	20.0229

Table 6: Ablation study on the number of Gaussian ellipses in Gaussian Bank (PSNR (dB)).

Conclusion

The proposed GaussianSR represents a paradigm shift in image super-resolution by introducing a continuous feature field characterized by a Gaussian distribution, departing from the traditional discrete feature storage methodology. This approach enables natural implementation of arbitrary-scale upsampling, as any point within the field can be explicitly determined according to the Gaussian distribution. By redefining pixel representations as continuous Gaussian fields, GaussianSR offers a more natural and fluid representation with fewer parameters. Furthermore, a classifier is trained to assign optimal Gaussian kernels to each pixel, tailoring the model to diverse input characteristics. Experimental results show that GaussianSR enhances super-resolution performance and reduces computational overhead, demonstrating the potential of integrating Gaussian expressions in computer vision tasks typically dominated by INR-based methodologies. This framework not only advances arbitrary-scale super-resolution but also suggests new directions for conceptualizing and processing visual information.

Acknowledgments

This work was partly supported by the National Natural Science Foundation of China (Nos.62171251&62311530100) and the Special Foundations for the Development of Strategic Emerging Industries of Shenzhen (No.KJZD20231023094700001).

References

Agustsson, E.; and Timofte, R. 2017. NTIRE 2017 Challenge on Single Image Super-Resolution: Dataset and Study. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1122–1131.

Cao, J.; Wang, Q.; Xian, Y.; Li, Y.; Ni, B.; Pi, Z.; Zhang, K.; Zhang, Y.; Timofte, R.; and Van Gool, L. 2023. CiaoSR: Continuous Implicit Attention-in-Attention Network for Arbitrary-Scale Image Super-Resolution. In *IEEE Conference on Computer Vision and Pattern Recognition*.

Chen, B.; Kwiatkowski, R.; Vondrick, C.; and Lipson, H. 2021a. Full-Body Visual Self-Modeling of Robot Morphologies. arXiv:2111.06389.

Chen, B.; Li, G.; Wu, R.; Zhang, X.; Chen, J.; Zhang, J.; and Zhang, L. 2024. Adversarial Diffusion Compression for Real-World Image Super-Resolution. arXiv:2411.13383.

Chen, B.; and Zhang, J. 2024. Practical Compact Deep Compressed Sensing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1–17.

Chen, H.; He, B.; Wang, H.; Ren, Y.; Lim, S.-N.; and Shrivastava, A. 2021b. NeRV: Neural Representations for Videos. In *NeurIPS*.

Chen, Y.; Liu, S.; and Wang, X. 2021. Learning Continuous Image Representation with Local Implicit Image Function. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 8624–8634.

Chen, Z.; and Zhang, H. 2019. Learning Implicit Fields for Generative Shape Modeling. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 5932–5941.

Dong, C.; Loy, C. C.; and Tang, X. 2016. Accelerating the Super-Resolution Convolutional Neural Network. In Leibe, B.; Matas, J.; Sebe, N.; and Welling, M., eds., *Computer Vision – ECCV 2016*, 391–407. Cham: Springer International Publishing. ISBN 978-3-319-46475-6.

Hu, X.; Mu, H.; Zhang, X.; Wang, Z.; Tan, T.; and Sun, J. 2019. Meta-SR: A Magnification-Arbitrary Network for Super-Resolution. In *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1575–1584.

Huang, J.-B.; Singh, A.; and Ahuja, N. 2015. Single image super-resolution from transformed self-exemplars. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 5197–5206.

Keetha, N.; Karhade, J.; Jatavallabhula, K. M.; Yang, G.; Scherer, S.; Ramanan, D.; and Luiten, J. 2024. SplaTAM: Splat, Track & Map 3D Gaussians for Dense RGB-D SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Kerbl, B.; Kopanas, G.; Leimkühler, T.; and Drettakis, G. 2023. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 42(4).

Lee, J.; and Jin, K. H. 2022. Local Texture Estimator for Implicit Representation Function. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 1919–1928.

Li, G.; Chen, B.; Zhao, C.; Zhang, L.; and Zhang, J. 2024. OSMamba: Omnidirectional Spectral Mamba with Dual-Domain Prior Generator for Exposure Correction. *arXiv preprint arXiv:2411.15255*.

Li, H.; Dai, T.; Li, Y.; Zou, X.; and Xia, S.-T. 2022. Adaptive Local Implicit Image Function for Arbitrary-Scale Super-Resolution. In *2022 IEEE International Conference on Image Processing (ICIP)*, 4033–4037.

Li, Y.; Li, S.; Sitzmann, V.; Agrawal, P.; and Torralba, A. 2021. 3D Neural Scene Representations for Visuomotor Control. In *Conference on Robot Learning*.

Lim, B.; Son, S.; Kim, H.; Nah, S.; and Lee, K. M. 2017. Enhanced Deep Residual Networks for Single Image Super-Resolution. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1132–1140.

Liu, L.; Gu, J.; Lin, K. Z.; Chua, T.-S.; and Theobalt, C. 2020. Neural Sparse Voxel Fields. *NeurIPS*.

Liu, S.; Chen, B.; Zou, W.; Sha, H.; Feng, X.; Han, S.; Li, X.; Yao, X.; Zhang, J.; and Zhang, Y. 2024. Compressive confocal microscopy imaging at the single-photon level with ultra-low sampling ratios. *Communications Engineering*, 3(1): 88.

Martin, D.; Fowlkes, C.; Tal, D.; and Malik, J. 2001. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*, volume 2, 416–423 vol.2.

Matsui, Y.; Ito, K.; Aramaki, Y.; Fujimoto, A.; Ogawa, T.; Yamasaki, T.; and Aizawa, K. 2016. Sketch-based manga retrieval using manga109 dataset. *Multimedia Tools and Applications*, 76(20): 21811–21838.

Matsuki, H.; Murai, R.; Kelly, P. H. J.; and Davison, A. J. 2024. Gaussian Splatting SLAM. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*.

Mildenhall, B.; Srinivasan, P. P.; Tancik, M.; Barron, J. T.; Ramamoorthi, R.; and Ng, R. 2020. NeRF: Representing Scenes as Neural Radiance Fields for View Synthesis. In Vedaldi, A.; Bischof, H.; Brox, T.; and Frahm, J.-M., eds., *Computer Vision – ECCV 2020*, 405–421. Cham: Springer International Publishing. ISBN 978-3-030-58452-8.

Nguyen, Q. H.; and Beksi, W. J. 2023. Single Image Super-Resolution via a Dual Interactive Implicit Neural Network. In *2023 IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 4925–4934.

Park, J. J.; Florence, P.; Straub, J.; Newcombe, R.; and Lovegrove, S. 2019. DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation. In *2019*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 165–174.

Park, K.; Sinha, U.; Barron, J. T.; Bouaziz, S.; Goldman, D. B.; Seitz, S. M.; and Martin-Brualla, R. 2021a. Nerfies: Deformable Neural Radiance Fields. *ICCV*.

Park, K.; Sinha, U.; Hedman, P.; Barron, J. T.; Bouaziz, S.; Goldman, D. B.; Martin-Brualla, R.; and Seitz, S. M. 2021b. HyperNeRF: A Higher-Dimensional Representation for Topologically Varying Neural Radiance Fields. *ACM Trans. Graph.*, 40(6).

Radu Timofte, L. V. G. M. Y., Eirikur Agustsson; and Zhang., L. 2017. NTIRE 2017 Challenge on Single Image Super-Resolution: Methods and Results. In *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 1110–1121.

Saito, S.; Huang, Z.; Natsume, R.; Morishima, S.; Li, H.; and Kanazawa, A. 2019. PIFu: Pixel-Aligned Implicit Function for High-Resolution Clothed Human Digitization. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2304–2314.

Shi, W.; Caballero, J.; Huszár, F.; Totz, J.; Aitken, A. P.; Bishop, R.; Rueckert, D.; and Wang, Z. 2016. Real-Time Single Image and Video Super-Resolution Using an Efficient Sub-Pixel Convolutional Neural Network. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 1874–1883.

Sitzmann, V.; Zollhöfer, M.; and Wetzstein, G. 2019. Scene Representation Networks: Continuous 3D-Structure-Aware Neural Scene Representations. In *Advances in Neural Information Processing Systems*.

Wu, H.; Ni, N.; and Zhang, L. 2023. Learning Dynamic Scale Awareness and Global Implicit Functions for Continuous-Scale Super-Resolution of Remote Sensing Images. *IEEE Transactions on Geoscience and Remote Sensing*, 61: 1–15.

Xu, X.; Wang, Z.; and Shi, H. 2022. UltraSR: Spatial Encoding is a Missing Key for Implicit Image Function-based Arbitrary-Scale Super-Resolution. arXiv:2103.12716.

Yan, C.; Qu, D.; Xu, D.; Zhao, B.; Wang, Z.; Wang, D.; and Li, X. 2024. GS-SLAM: Dense Visual SLAM with 3D Gaussian Splatting. In *CVPR*.

Yang, J.; Shen, S.; Yue, H.; and Li, K. 2021. Implicit Transformer Network for Screen Content Image Continuous Super-Resolution. In Beygelzimer, A.; Dauphin, Y.; Liang, P.; and Vaughan, J. W., eds., *Advances in Neural Information Processing Systems*.

Yu, A.; Ye, V.; Tancik, M.; and Kanazawa, A. 2021. pixelNeRF: Neural Radiance Fields from One or Few Images. In *CVPR*.

Zhang, Y.; Tian, Y.; Kong, Y.; Zhong, B.; and Fu, Y. 2018. Residual Dense Network for Image Super-Resolution. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2472–2481.

Zhang, Y.; van Rozendaal, T.; Brehmer, J.; Nagel, M.; and Cohen, T. 2022. Implicit Neural Video Compression. In *ICLR Workshop on Deep Generative Models for Highly Structured Data*.