Latent Modulated Function for Computational Optimal Continuous Image Representation

Zongyao He[1]   Zhi Jin[1,2,*]

[1]School of Intelligent Systems Engineering, Shenzhen Campus of Sun Yat-sen University
[2]Guangdong Provincial Key Laboratory of Fire Science and Technology

hezy28@mail2.sysu.edu.cn, jinzh26@mail.sysu.edu.cn

Abstract

The recent work Local Implicit Image Function (LIIF) and subsequent Implicit Neural Representation (INR) based works have achieved remarkable success in Arbitrary-Scale Super-Resolution (ASSR) by using MLP to decode Low-Resolution (LR) features. However, these continuous image representations typically implement decoding in High-Resolution (HR) High-Dimensional (HD) space, leading to a quadratic increase in computational cost and seriously hindering the practical applications of ASSR. To tackle this problem, we propose a novel Latent Modulated Function (LMF), which decouples the HR-HD decoding process into shared latent decoding in LR-HD space and independent rendering in HR Low-Dimensional (LD) space, thereby realizing the first computational optimal paradigm of continuous image representation. Specifically, LMF utilizes an HD MLP in latent space to generate latent modulations of each LR feature vector. This enables a modulated LD MLP in render space to quickly adapt to any input feature vector and perform rendering at arbitrary resolution. Furthermore, we leverage the positive correlation between modulation intensity and input image complexity to design a Controllable Multi-Scale Rendering (CMSR) algorithm, offering the flexibility to adjust the decoding efficiency based on the rendering precision. Extensive experiments demonstrate that converting existing INR-based ASSR methods to LMF can reduce the computational cost by up to 99.9%, accelerate inference by up to $57\times$, and save up to 76% of parameters, while maintaining competitive performance. The code is available at https://github.com/HeZongyao/LMF.

1. Introduction

Image data has long been stored and presented as discrete 2D arrays with fixed resolutions, while the visual world we perceive is continuous and precise. As a result, there is a
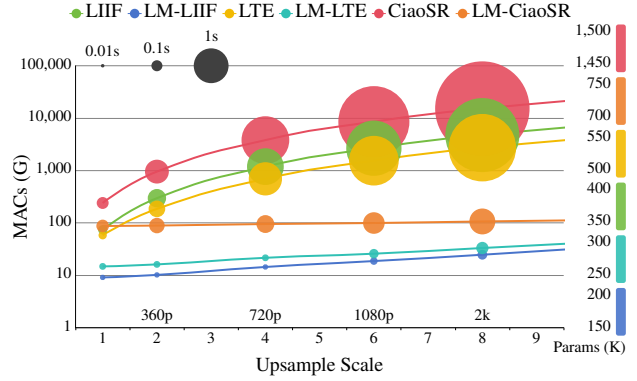
*Corresponding author

Figure 1. Efficiency comparisons (320×180 input) for ASSR. LMF-based ASSR methods significantly reduce computational cost (MACs), runtime (circle sizes), and parameters (colors).

growing demand for arbitrary scale image resizing in various real-world scenarios. For instance, users often require continuous resizing of images during capture, viewing, and editing. Recent developments in Single Image Super-Resolution (SISR) [7, 9, 12, 16, 20, 21, 23–25, 36, 45–47] have showcased significant advancements in reconstructing High-Resolution (HR) images from Low-Resolution (LR) counterparts. However, traditional interpolation methods are still the preferred choice for handling real-world image resizing. This is because SISR methods typically learn upsampling functions for a particular scale factor, requiring training different models for different scales. In contrast, Arbitrary-Scale Super-Resolution (ASSR) utilizes a single model to handle arbitrary scales, thus presenting significant potential in practical image-resizing applications.

Building on the success of Implicit Neural Representation (INR), Local Implicit Image Function (LIIF) [11] is a recent ASSR approach that learns to continuously represent 2D images through an implicit function. LIIF encodes an image as latent codes and employs a decoding MLP to predict pixel values based on coordinates and their neighboring latent codes. The use of continuous coordinates enables LIIF and subsequent INR-based ASSR works
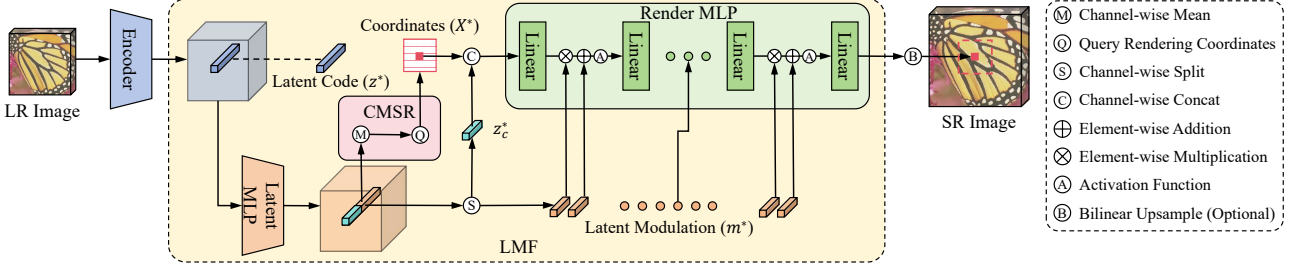
Figure 2. The framework of our LMF-based continuous image representation. Given a latent code generated by the encoder, an HD latent MLP generates the latent modulation to adjust the features of an LD render MLP, thereby achieving efficient arbitrary-resolution rendering.

[5, 8, 14, 15, 22, 34, 37–41] to represent images at any resolution, achieving robust performance in up to ×30 SR task. However, existing INR-based ASSR methods adhere to an intuitive ASSR paradigm that uses a complex decoding function in HR High-Dimensional (HD) space to independently predict the pixel value of each HR coordinate. As a result, their computational cost and runtime increase quadratically with the scale factor, contradicting the goal of real-world image-resizing applications (refer to Figure 1). Upon closer examination, the significant computational redundancy of HR-HD decoding can be divided into three aspects. Firstly, there is inherent similarity within the HR region corresponding to an LR latent code, suggesting the need for merging shared computations within this region before independently predicting each HR pixel value. Secondly, this paradigm ignores the varying complexity of different LR latent codes and applies the same computational process to predict all the HR pixel values. Lastly, this paradigm couples the HD MLP parameters and HR computations, while such a large number of parameters are not always needed for each prediction.

To tackle this problem, we propose a novel Latent Modulated Function (LMF) to realize a computational optimal paradigm of continuous image representation. By leveraging latent modulation, LMF successfully disentangles the decoding process from HR-HD space into latent (LR-HD) and render (HR-LD) spaces. As shown in Figure 2, LMF first utilizes a latent MLP to generate the latent modulation of a given latent code. When querying the coordinates near the latent code, the latent modulation is applied to adjust each hidden linear layer in the render MLP. This modulation enables LMF to minimize the dimension of the render MLP without sacrificing performance. Furthermore, we observe a positive correlation between the latent modulation intensity and the input image complexity. Building on this insight, we propose a Controllable Multi-Scale Rendering (CMSR) algorithm. CMSR not only allows the computational cost of rendering to be proportional to the input image complexity rather than the output resolution, but also provides the flexibility to balance precision and efficiency during testing. Computational optimal and fast ASSR holds great promise for future research and downstream appli-

cations. Experiments demonstrate that transforming existing INR-based ASSR methods into LMF reduces computational costs by 90.4% to 99.9%, accelerates inference by 2.4× to 56.9×, and saves 45.1% to 76.0% of parameters, while maintaining competitive PSNR performance.

Our main contributions are summarized as follows:

- We propose a Latent Modulated Function (LMF) that realizes computational optimal continuous image representation, which decouples the HR-HD decoding into decoding in latent (LR-HD) and render (HR-LD) spaces.
- Based on the correlation between modulation intensity and input image complexity, we propose a Controllable Multi-Scale Rendering (CMSR) algorithm to effortlessly balance the rendering efficiency and precision at test time.
- Extensive experiments demonstrate that seamlessly converting existing INR-based ASSR methods to LMF can achieve substantial reductions in computational cost, runtime, and parameters, while preserving performance.

2. Related Work

2.1. Implicit Neural Representation

INR has demonstrated remarkable success for continuous signal representation across various tasks and modalities, including signal reconstruction [10, 27, 31, 32], image and video compression [6, 13, 35], and image generation [33, 43]. The highlight of INR research is NeRF [28] and subsequent works [2, 3, 30, 42], which learn MLPs to infer the RGB and density of any given coordinates and view directions in a 3D scene. However, a major challenge faced by INR is the extensive training process required to fit a signal from scratch. To address this, MINER [31] and KiloNeRF [30] propose the use of multiple small MLPs to independently learn different segments of a signal. Another solution is hyper-base network design [10, 13, 27, 33], where a pre-trained hyper network generates INR for any input signal during testing. To represent complex signals, hyper networks often generate modulations for base networks.

2.2. Single-Image Super-Resolution

SISR as a classic low-level vision task has been studied for decades. In recent years, this field witnessed a variety of

CNN-based networks, such as SRCNN [12], LapSRN [20], SRResNet [21], EDSR [25], MemNet [36], RCAN [46], RDN [47], and SRFBN [23]. Lately, image processing transformers, such as IPT [7], SwinIR [24], and HAT [9], have been proposed, which surpass CNN-based methods in SR performance using a large dataset. Although SISR methods have achieved remarkable performance, most of them are designed for a specific scale.

ASSR methods are designed to utilize a single network for all scales, showcasing significant advancements in practicality and convenience over previous works. Hu *et al.* proposed MetaSR [17] to generate an upsampling layer for ASSR using a meta-network. However, MetaSR exhibits poor generalization for large-scale SR. Building upon the success of INR, a novel continuous image representation was proposed in LIIF [11], where each pixel value is inferred from an MLP using its coordinate and neighboring 2D features. Subsequently, a variety of INR-based ASSR methods have been proposed [5, 8, 14, 15, 22, 34, 37–41]. Notably, Lee *et al.* proposed LTE [22], which introduces a dominant-frequency estimator to continuously capture fine details. Cao *et al.* introduced CiaoSR [5], a continuous implicit attention-in-attention network that learns ensemble weights for nearby local features, achieving State-Of-The-Art (SOTA) performance. Despite the robust performance of INR-based ASSR methods for scales up to ×30, they adhere to the paradigm of decoding in HR-HD space, leading to prohibitively high computational cost and runtime.

3. Methods

3.1. Preliminary and Motivation

In continuous image representations, an image I_{in} is represented as 2D feature maps F in latent space $\mathbb{R}^{H_{in} \times W_{in} \times D_F}$. A latent code $z^* \in \mathbb{R}^{D_F}$ represents a feature vector derived from the feature maps, which is used to decode its nearest coordinates in the continuous spatial domain $X \subset \mathbb{R}^2$. Since the encoder can be any feature extractor from existing SISR methods, the essence of continuous image representations is the decoding function, which arbitrarily upsamples LR-HD features to HR-LD pixel values. A simple and intuitive paradigm is to learn a decoding function in HR-HD space that independently decodes the pixel value of each HR coordinate. This vanilla decoding function f_θ parameterized as an HD MLP θ can be formulated as:

$$I(x_{i,j}^*) = f_\theta(z^*, x_{i,j}^*), \qquad (1)$$

where $x_{i,j}^* \in X^*$ refers to any nearest coordinate of z^*, and $I(x_{i,j}^*)$ represents the predicted pixel value at $x_{i,j}^*$.

However, as mentioned earlier in the Introduction section, this vanilla paradigm suffers from three aspects of computational redundancy. A computational optimal decoding function should allow computations proportional to

the input image complexity rather than the output resolution. With this in mind, we propose to decouple the HR-HD decoding process into shared decoding in latent space and independent rendering in render space, along with a rendering algorithm adaptive to the input image content.

3.2. Latent Modulated Function

To eliminate the computational redundancy in HR-HD decoding, we first propose a two-stage decoding process comprising a latent MLP θ_l and a render MLP θ_r. The latent MLP handles the shared decoding process within the HR region of a latent code z^*, while the render MLP independently predicts the pixel value of each HR coordinate. The LR-HD output $f_{\theta_l}(z^*)$ from the latent MLP is utilized to assist the render MLP in rendering. However, directly using this output $f_{\theta_l}(z^*)$ as the input of the render MLP is akin to using the latent code itself, as demonstrated in Eq. 1, resulting in no efficiency improvement yet:

$$I(x_{i,j}^*) = f_{\theta_r}(f_{\theta_l}(z^*), x_{i,j}^*). \qquad (2)$$

Furthermore, to render arbitrary pixels accurately, it is necessary for the render MLP to operate in HR space. Therefore, the most viable approach to minimize rendering computations is to reduce the dimension or depth of the render MLP. However, an HD render MLP is still needed in Eq. 2 to fully receive and process the output $f_{\theta_l}(z^*)$, which contradicts our intention of utilizing an LD render MLP. To overcome these limitations, modulations become a suitable solution to connect the latent and render space. Modulations have been used in INR [13, 27, 33] and are often associated with the hyper-base network design. A hyper network converts an input image to modulations, which are then utilized by a small base network to map coordinates to a continuous image. However, these modulations are shared by the input image and work at the image level, resulting in unsatisfactory performance and challenging training. Hence, we introduce a latent modulation $m^* \in \mathbb{R}^{D_M}$, which is generated by the latent MLP using the latent code z^*. The latent modulation consists of the FiLM layers [29], where each layer consists of a scale modulation and a shift modulation:

$$m^* = [\alpha_1^*, \beta_1^*, \dots, \alpha_K^*, \beta_K^*] = Split(f_{\theta_l}(z^*)), \quad (3)$$

where α^* represents the element-wise scale modulation, β^* represents the element-wise shift modulation, and K represents the depth of hidden layers in the render MLP.

For each HR coordinate, the latent modulation m^* of the latent code z^* nearest to this coordinate is applied to the render MLP, and is applied to the hidden features $h^* \in \mathbb{R}^{D_H}$ after each hidden linear layer:

$$h_{k+1}^* = f_{\theta_r^{k+1}}(\sigma((1 + \alpha_k^*) \odot h_k^* + \beta_k^*)), \qquad (4)$$

where h_k^* represents the hidden features after the k-th hidden linear layer, and σ represents the activation function.
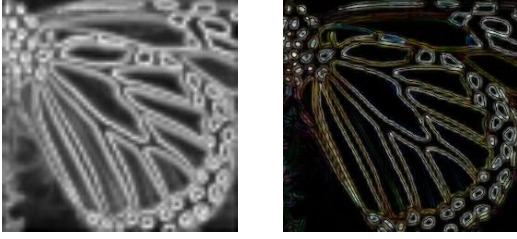
Figure 3. Visualization ($\times 4$ SR) of the normalized mean of shift modulation (left), and the normalized residual between the LM-LIIF predicted image and the bilinear upsampled image (right).

Importantly, the proposed latent modulation effectively solves the incompatibility between LR-HD latent codes and HR-LD rendering and significantly disentangles the computations from parameters. For a latent modulation with 192 dimensions, we can accurately render images using a 7-layer render MLP with only 16 dimensions. This LD render MLP only learns the general rendering rules of natural images, while the latent modulation optimizes its rendering process based on the characteristics of each input latent code. As a result, a tiny render MLP can quickly adapt to any input latent code and accomplish efficient rendering at arbitrary resolution. A remaining concern is that the latent codes in existing INR-based ASSR methods often possess high dimensions, resulting in unnecessary computations when directly input into the render MLP. Therefore, we compress the HD latent code to an LD one z_c^* using the latent MLP, extending Eq. 3 to the first stage in Eq. 5. Building on the latent modulation, we propose a computational optimal paradigm of continuous image representation, called Latent Modulated Function (LMF):

$$
\begin{aligned}
\text{Stage 1: } & [m^*, z_c^*] = Split(f_{\theta_l}(z^*)), \\
\text{Stage 2: } & I(x_{i,j}^*) = f_{\theta_r, m^*}(z_c^*, x_{i,j}^*).
\end{aligned} \tag{5}
$$

LMF is a universal paradigm for efficient ASSR and is compatible with existing INR-based ASSR methods. By effortlessly relocating the time-consuming operations and large MLPs to latent space, LMF allows a minimal render MLP to perform the same level of ASSR as the original ASSR method. In addition, LMF effectively decouples the computational expensive techniques of feature unfolding and local ensemble [11]. Specifically, LMF only utilizes feature unfolding for the latent MLP, where nearby 3×3 feature vectors are concatenated as inputs. LMF performs local ensemble exclusively for the render MLP, which merges the predictions from the top-left, top-right, bottom-left, and bottom-right latent codes to achieve continuous predictions.

3.3. Controllable Multi-Scale Rendering

The HR regions of two latent codes often have different signal complexity, allowing for the use of different decoding functions. However, the challenge lies in obtaining the signal complexity of each latent code. The modulation scheme
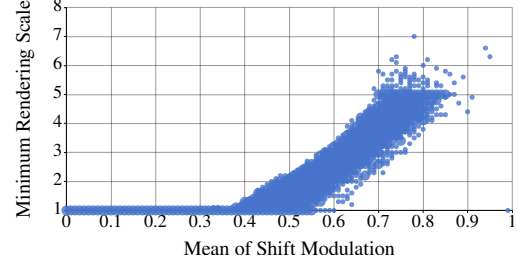

Figure 4. The positive correlation between the means of shift modulation and the minimum rendering scale factors in LM-LIIF.

in Eq. 4 suggests that the intensity of latent modulation is proportional to the signal complexity of its latent code. A lower intensity implies that the signal can be sufficiently rendered by the render MLP itself. In Figure 3, we display the normalized mean of shift modulation (left), and the normalized residual between the SR result of LMF-based LIIF and the bilinear upsampled result (right). It is evident that the means of modulation closely align with the signal complexity of the predicted image. Regions with higher means of modulation consistently correspond to more complex textures and edges. Hence, we employ the mean of shift modulation as the intensity of latent modulation.

In LMF, we define the signal complexity of the latent code z^* as the minimum scale factor s^* needed to fully represent its signal. In simple terms, any rendering with a scale factor $s, s > s^*$ will yield the same result as rendering with s^* followed by upsample interpolation (*i.e.* bilinear) with scale $\frac{s}{s^*}$. Therefore, we only need to render z^* with scale $min(s^*, s)$ when upsampling an image with a scale s. To obtain the minimum scale factor of each mean of shift modulation, in Figure 4, we sample 100 means of shift modulation uniformly from the range of $[0, 1]$ and measure their minimum scale factors on the LR images in the training set. Figure 4 clearly shows the positive correlation between the means of shift modulation and the minimum scale factors. However, since they do not have a one-to-one correspondence, we create a Scale2Mods table by sampling common scale factors and recording their minimum and maximum means of shift modulation.

Based on the Scale2Mods table, we propose a Controllable Multi-Scale Rendering (CMSR) algorithm to decode latent codes with their minimum scale factors. Firstly, we initialize s_{-1} as 1, $I_{s_{-1}}$ as the LR image. For each sampled scale $s_i \in [s_0, s_1, \ldots, s]$, we first upsample the last result $I_{s_{i-1}}$ using bilinear interpolation with $\frac{s_i}{s_{i-1}}$ to obtain I_{s_i}. We then query the Scale2Mods table to obtain the range of modulation means $[m_{min}^{s_i}, m_{max}^{s_i}]$ for s_i. For s_i, the render MLP only needs to render the latent codes whose corresponding modulation means falling within this range. We repeat the above process iteratively until we obtain the final SR result or until all latent codes have been decoded. For precision controllable rendering, when creating

Table 1. Quantitative comparison (PSNR (dB)/MACs) for ASSR on DIV2K validation set. The best results are highlighted in **bold**.

Method	Params	In-scale			Out-of-scale		
		$\times 2$	$\times 3$	$\times 4$	$\times 6$	$\times 12$	$\times 18$
Bicubic	-	31.01/-	28.22/-	26.66/-	24.82/-	22.27/-	21.00/-
EDSR-b [25]	-	34.55/-	30.90/-	28.94/-	-	-	-
LIIF [11]	355K	**34.67**/3.567T	**30.96**/3.567T	**29.00**/3.567T	**26.75**/3.567T	23.71/3.567T	22.17/3.567T
LM-LIIF	**170K**	34.65/**125.32G**	30.95/**65.37G**	28.99/**44.38G**	**26.75**/**25.30G**	**23.72**/**6.61G**	**22.19**/**2.95G**
LTE [22]	506K	**34.72**/2.228T	**31.02**/2.123T	**29.04**/2.086T	**26.81**/2.059T	**23.78**/2.044T	**22.23**/2.026T
LM-LTE	**278K**	34.71/**198.81G**	30.99/**100.75G**	29.03/**66.43G**	26.78/**34.96G**	23.74/**9.07G**	22.21/**4.10G**
CiaoSR-1.4M [5]	1.429M	**34.91**/11.460T	**31.15**/11.443T	**29.23**/11.437T	**26.95**/11.433T	**23.88**/11.430T	**22.32**/11.348T
LM-CiaoSR	**753K**	34.81/**1.071T**	31.10/**501.36G**	29.12/**293.01G**	26.86/**135.66G**	23.80/**42.60G**	22.26/**23.90G**

Table 2. Quantitative comparison (PSNR (dB)) for in-scale SR on test sets. The best results are highlighted in **bold**.

Method	Set5			Set14			BSD100			Urban100			DIV2K100		
	$\times 2$	$\times 3$	$\times 4$	$\times 2$	$\times 3$	$\times 4$	$\times 2$	$\times 3$	$\times 4$	$\times 2$	$\times 3$	$\times 4$	$\times 2$	$\times 3$	$\times 4$
RDN [47]	38.24	34.71	32.47	34.01	30.57	28.81	32.34	29.26	27.72	32.89	28.80	26.61	34.94	31.22	29.19
LIIF [11]	38.17	34.68	**32.50**	33.97	30.53	28.80	**32.32**	29.26	27.74	**32.87**	28.82	**26.68**	34.99	31.26	**29.27**
LM-LIIF	**38.18**	**34.72**	**32.50**	**34.06**	30.55	28.83	**32.32**	29.28	27.75	32.82	28.81	26.66	34.96	31.25	**29.27**
LTE [22]	**38.23**	34.72	32.61	34.09	**30.58**	28.88	32.36	29.30	27.77	33.04	28.97	26.81	35.04	31.32	29.33
LM-LTE	**38.23**	34.76	32.53	**34.11**	30.56	28.86	**32.37**	29.31	27.78	33.03	28.96	26.80	35.03	31.31	29.32
CiaoSR-1.4M [5]	**38.29**	34.85	32.66	**34.22**	30.65	28.93	**32.41**	29.34	27.83	33.30	29.17	27.11	35.15	31.42	29.45
LM-CiaoSR	38.24	34.81	32.59	34.11	30.62	28.90	32.39	29.32	27.80	33.13	29.13	26.95	35.08	31.36	29.37
SwinIR [24]	38.35	34.89	32.72	34.14	30.77	28.94	32.44	29.37	27.83	33.40	29.29	27.07	35.26	31.50	29.48
LIIF [11]	38.28	34.87	32.73	34.14	30.75	28.98	32.39	29.34	27.84	33.36	29.33	27.15	35.17	31.46	29.46
LM-LIIF	**38.31**	34.88	32.78	**34.31**	30.79	29.02	**32.43**	29.37	27.85	**33.46**	29.43	27.21	35.21	31.48	**29.49**
LTE [22]	**38.33**	34.89	32.81	34.25	30.80	29.06	32.44	**29.39**	27.86	33.50	29.41	**27.24**	35.24	31.50	29.51
LM-LTE	38.32	34.88	32.77	**34.28**	30.79	29.01	**32.46**	**29.39**	27.87	**33.52**	29.44	**27.24**	35.24	31.50	29.50
CiaoSR-3.1M [5]	**38.38**	34.91	32.84	**34.33**	30.82	29.08	32.47	29.42	27.90	33.65	29.52	27.42	35.29	31.55	29.59
LM-CiaoSR	38.37	**34.94**	32.87	**34.33**	30.83	29.08	**32.48**	29.43	27.90	33.65	29.58	27.39	35.30	31.55	29.56

the Scale2Mods table, we set an MSE threshold t and define the signal complexity of a latent code as the minimum scale factor needed to render its signal within this error t. By effortlessly adjusting the MSE threshold during testing, CMSR is capable of rendering with varying precision and speed, while adapting naturally to the image content.

4. Experiments

4.1. Experimental Setups

Dataset. Following the setups in previous ASSR methods [5, 11, 22], we employ the DIV2K dataset [1] as the training set. In each training iteration, we uniformly sample a downsampling scale s from a continuous range of $[\times 1, \times 4]$. HR patches with spatial size of $48s \times 48s$ are cropped and bicubic downsampled to LR patches with spatial size of 48×48. To supervise the training process, we randomly sample 48×48 pixels from each HR patch. For evaluation, we present results on the DIV2K validation set (DIV2K 100) along with four widely-used benchmark test sets: Set5 [4], Set14 [44], BSD100 [26], and Urban100 [18].

Implementation Details. We transform three previous ASSR methods, namely LIIF [11], LTE [22], and CiaoSR [5], into our proposed LMF. LM-LIIF employs a 2-layer latent MLP with 208 dimensions, and a 7-layer render MLP with 16 dimensions. As for LM-LTE, we employ a 2-layer latent MLP with 256 dimensions, a 9-layer render MLP with 16 dimensions, and a local texture estimator [22] with 128 dimensions. LM-CiaoSR employs four 2-layer latent MLPs with 256 dimensions to generate key, value, and query information, followed by a 9-layer render MLP with 16 dimensions. The detailed architectures of LMF-based methods are provided in the supplementary material.

We utilize EDSR-baseline (EDSR-b) [25], RDN [47], and SwinIR [24] without their upsampling modules as encoders. For both EDSR-b and RDN encoders, we conduct training for 1000 epochs with a batch size of 16. The learning rate is initialized as 1×10^{-4} and decayed by a factor of 0.5 every 200 epochs. As for the SwinIR encoder, we conduct training for 1000 epochs with a batch size of 32. The learning rate is initialized as 2×10^{-4} and decayed by a factor of 0.5 at $[500, 800, 900, 950]$ epochs. For optimization, we utilize the L1 pixel loss as the loss function and employ the Adam [19] optimizer with $\beta_1 = 0.9$, $\beta_2 = 0.999$. Our training details align with previous ASSR works [5, 22].

Evaluation Settings. In line with previous methods [5, 11,

Table 3. Quantitative comparison (PSNR (dB)) for out-of-scale SR on test sets. The best results are highlighted in **bold**. The first 6 rows use RDN as the encoder, and the last 6 rows use SwinIR as the encoder.

Method	Set5			Set14			BSD100			Urban100			DIV2K100		
	×6	×8	×12	×6	×8	×12	×6	×8	×12	×6	×8	×12	×6	×12	×18
LIIF [11]	29.15	27.14	**24.86**	**26.64**	**25.15**	23.24	**25.98**	24.91	23.57	**24.20**	**22.79**	**21.15**	26.99	23.89	22.34
LM-LIIF	**29.18**	**27.21**	24.71	**26.66**	**25.17**	**23.31**	25.98	**24.93**	**23.58**	24.14	22.77	21.15	26.99	**23.91**	**22.36**
LTE [22]	**29.32**	**27.26**	24.79	**26.71**	25.16	23.31	**26.01**	**24.95**	23.60	**24.28**	**22.88**	21.22	**27.04**	23.95	**22.40**
LM-LTE	29.21	27.22	**24.82**	26.69	25.15	**23.32**	**26.01**	**24.95**	**23.61**	24.26	22.86	**21.24**	27.03	**23.95**	22.39
CiaoSR-1.4M [5]	**29.46**	**27.36**	**24.92**	**26.79**	**25.28**	**23.37**	**26.07**	**25.00**	**23.64**	**24.58**	**23.13**	**21.42**	**27.16**	**24.06**	**22.48**
LM-CiaoSR	29.24	27.26	24.80	26.72	25.18	23.30	26.03	24.81	23.60	24.40	22.98	21.33	27.07	23.98	22.42
LIIF [11]	**29.46**	**27.36**	**24.99**	**26.82**	25.34	23.39	26.07	25.01	**23.65**	24.59	23.14	21.43	27.15	24.02	22.43
LM-LIIF	**29.46**	**27.36**	24.98	**26.82**	**25.37**	**23.43**	**26.08**	**25.03**	**23.65**	**24.61**	**23.15**	**21.49**	**27.18**	**24.06**	**22.49**
LTE [22]	29.50	27.35	**25.07**	**26.86**	**25.42**	**23.44**	26.09	**25.03**	**23.66**	24.62	**23.17**	**21.50**	**27.20**	**24.09**	**22.50**
LM-LTE	**29.52**	**27.39**	25.01	26.83	25.38	23.41	**26.10**	**25.03**	**23.66**	**24.63**	**23.17**	**21.50**	27.19	24.08	**22.50**
CiaoSR-3.1M [5]	**29.62**	**27.45**	24.96	**26.88**	**25.42**	23.38	**26.13**	**25.07**	**23.68**	**24.84**	**23.34**	**21.60**	**27.28**	**24.15**	**22.54**
LM-CiaoSR	29.52	27.42	**25.02**	26.84	25.41	**23.45**	26.12	25.06	**23.68**	24.74	23.26	21.58	27.24	24.12	22.53

Table 4. Efficiency comparison (MACs/GPU runtime) for ASSR. The best results are highlighted in **bold**.

Decoder	Params	GPU mem.	180p			270p			360p		
			720p	1080p	2k	720p	1080p	2k	720p	1080p	2k
LIIF [11]	355K	7.7GB	1.160T/ 1.1148s	2.609T/ 2.5230s	4.638T/ 4.5232s	1.160T/ 1.1195s	2.609T/ 2.5347s	4.638T/ 4.5544s	1.160T/ 1.1272s	2.609T/ 2.5456s	4.638T/ 4.5589s
LM-LIIF	**170K**	**5.3GB**	**14.43G/ 0.0246s**	**18.72G/ 0.0474s**	**24.72G/ 0.0795s**	**25.39G/ 0.0363s**	**32.46G/ 0.0593s**	**37.50G/ 0.0919s**	**40.74G/ 0.0541s**	**47.81G/ 0.0770s**	**57.71G/ 0.1107s**
LTE [22]	506K	8.4GB	694.34G/ 0.9135s	1.506T/ 2.0366s	2.666T/ 3.7578s	714.11G/ 0.9191s	1.526T/ 2.0497s	2.685T/ 3.7694s	741.80G/ 0.9258s	1.553T/ 2.0637s	2.712T/ 3.7791s
LM-LTE	**278K**	**4.7GB**	**21.60G/ 0.0382s**	**25.87G/ 0.0750s**	**33.21G/ 0.1254s**	**39.53G/ 0.0545s**	**48.59G/ 0.0905s**	**52.88G/ 0.1414s**	**64.63G/ 0.0774s**	**73.69G/ 0.1123s**	**86.38G/ 0.1640s**
CiaoSR-1.4M [5]	1.429M	**8.8GB**	3.718T/ 1.8820s	8.362T/ 4.1718s	14.864T/ 7.3619s	3.721T/ 1.9838s	8.365T/ 4.2999s	14.867T/ 7.5611s	3.725T/ 2.1355s	8.370T/ 4.5064s	14.871T/ 7.8366s
LM-CiaoSR	**753K**	8.9GB	**95.23G/ 0.2637s**	**99.57G/ 0.3849s**	**106.68G/ 0.5546s**	**204.09G/ 0.5263s**	**214.32G/ 0.6476s**	**217.83G/ 0.8160s**	**356.46G/ 0.8844s**	**366.69G/ 1.0013s**	**381.02G/ 1.1778s**

22], we employ PSNR as the objective criterion to assess quality performance. To evaluate efficiency, we measure the computational cost using Multiply-Accumulate Operations (MACs) and quantify the model speed using the runtime on an RTX 2080 Ti GPU. Additionally, we provide the number of parameters and GPU memory usage. Notably, the MACs, runtime, GPU memory, and parameters are evaluated solely on the decoder unless explicitly stated.

4.2. ASSR Comparison

Quantitative Results. In Table 1, we compare our LMF-based methods with reference ASSR methods, namely LIIF [11], LTE [22], and CiaoSR [5], on the DIV2K validation set using EDSR-b as the encoder. We include the results of bicubic interpolation and EDSR-b with its initial upsampling module for reference. The results demonstrate that LMF-based methods achieve a significant reduction in computational cost while delivering competitive PSNR performance compared to the original methods. Compared to LIIF, LM-LIIF achieves a reduction of 96.6% to 99.9% in computational cost across all scales. Similarly, LM-LTE and LM-CiaoSR provide 91.3% to 99.8% and 90.4% to

99.8% reduction in computational cost compared to LTE and CiaoSR, respectively. However, LM-CiaoSR exhibits slightly lower PSNR than CiaoSR due to relocating the expensive attention-in-attention modules from HR-HD space to latent space for the minimal computational cost.

In Table 2 and 3, we present the comprehensive PSNR evaluation for ASSR using RDN and SwinIR as the encoders. Notably, CiaoSR utilizes a 1.429M decoder for EDSR-b and RDN, and a 3.065M decoder for SwinIR, while our LM-CiaoSR employs the same decoder for all encoders. The results demonstrate the competitive PSNR performance of our LMF-based methods, especially LM-LIIF and LM-LTE, compared to the reference methods on benchmark test sets. Notably, LMF-based methods exhibit superior PSNR performance gains on the SwinIR encoder. The results presented in Table 1, 2, and 3 collectively validate the universal applicability of our LMF across different encoders, INR-based ASSR methods, test sets, and scales.

In Table 4, we present comparisons of MACs, runtime, GPU memory, and parameters for ASSR across multiple input and output resolutions. We utilize bicubic downsampled images from the DIV2K validation set as inputs for
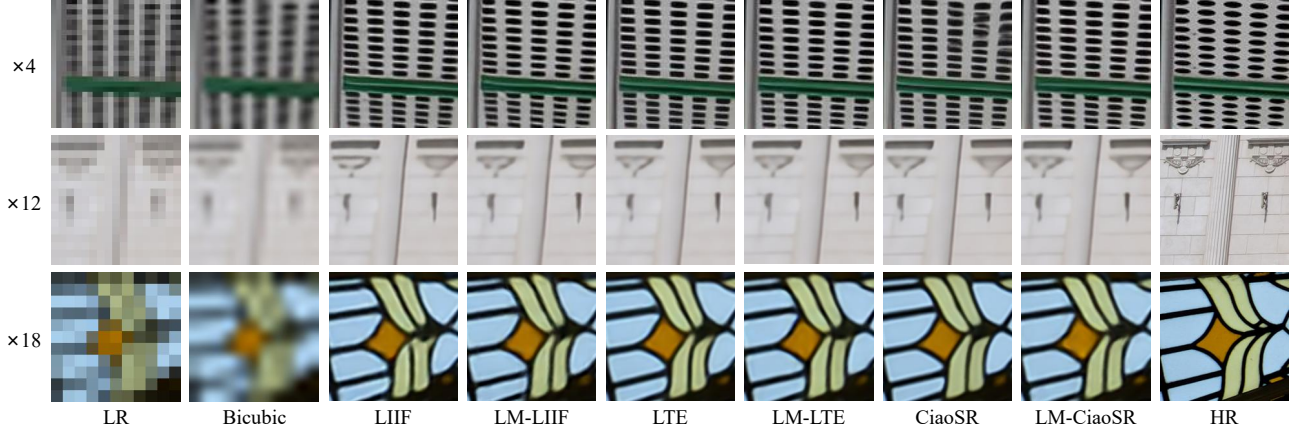
Figure 5. Qualitative comparison for ASSR. All of the ASSR methods use SwinIR as the encoder.

Table 5. Ablation study (PSNR(dB)/MACs on DIV2K validation set) on latent modulations and latent codes, using EDSR-b as the encoder.

Decoder	Latent code	Latent modulation	Latent MLP	Render MLP	Params	In-scale		Out-of-scale	
						×2	×4	×8	×16
LIIF	576	×	×	5 × 256	355K	**34.67**/3.567T	**29.00**/3.567T	**25.40**/3.567T	22.61/3.567T
LIIF (tiny)	576	×	×	9 × 16	**12K**	33.61/117.44G	28.18/117.44G	24.78/117.44G	22.16/117.44G
LIIF (C2F)	192	×	2 × 192	7 × 16	202K	34.11/221.54G	28.63/114.62G	25.11/73.04G	22.38/58.91G
LM-LIIF	64	Scale	2 × 192	7 × 16	136K	34.61/**110.94G**	28.96/46.87G	25.38/30.79G	22.61/26.58G
LM-LIIF	64	Shift	2 × 192	7 × 16	136K	34.59/**110.94G**	28.94/46.87G	25.36/30.79G	22.59/26.58G
LM-LIIF	64	Scale & shift	2 × 192	7 × 16	155K	34.66/123.10G	**29.00**/49.91G	**25.40**/31.55G	**22.63**/26.77G
LM-LIIF	16	Scale & shift	2 × 208	7 × 16	170K	34.65/125.32G	28.99/**44.38G**	**25.40**/24.10G	22.62/**18.90G**

each resolution, and all results exclude the encoders. As shown in the table, our LMF-based methods exhibit a significant reduction in computational cost, ranging from 90.7% to 99.5%, and a substantial reduction in runtime, ranging from 58.6% to 98.2%, compared to the reference methods. Compared to the reference methods, LMF-based methods also exhibit a reduction in parameters, ranging from 45.1% to 52.1%, which is attributed to the decoupling of computations from parameters, demonstrating the significance of our elegant design over merely stacking parameters.

Qualitative Results. In Figure 5, we present a qualitative comparison for ASSR, using SwinIR as the encoder. The visual results demonstrate that our LMF-based methods achieve competitive visual quality across all scales compared to the reference methods. Remarkably, LMF-based methods achieve continuous and precise results while significantly reducing computational cost and runtime. This confirms that LMF effectively addresses the challenge of upsampling LR-HD features to HR-LD pixel values, eliminating the need for an HR-HD decoding function. More visual results are provided in the supplementary materials.

5. Ablation Study

In this section, we investigate the effect of each component in LMF. Both PSNR and MACs results on the DIV2K vali-

dation set are provided for a comprehensive evaluation, and all of the models use EDSR-b as the encoder. Notably, the CMSR algorithm is not used in Table 5 and Table 6.

Latent Modulation. In Table 5, we evaluate the impact of latent modulation. Firstly, the comparison between LIIF and LIIF (tiny) reveals a significant decline in PSNR performance when shrinking the decoder from a 5-layer MLP with 256 dimensions to a 9-layer MLP with 16 dimensions. This performance drop is expected due to the substantial reduction in network parameters. However, our LM-LIIF model, which utilizes a 7-layer render MLP with 16 dimensions, outperforms LIIF in terms of PSNR, while consuming computational cost even lower than LIIF (tiny). Furthermore, we introduce a coarse-to-fine variant of LIIF, where both stages employ the same structure as LM-LIIF. Unlike LM-LIIF, LIIF (C2F) fails to achieve competitive PSNR performance compared to LIIF, highlighting the critical role of latent modulation in reducing the dimensions of the render MLP. These results also validate that HR-HD computations are redundant for decoding LR-HD features.

In Table 5, we also compare the performance of different types of latent modulations, namely the scale modulation, the shift modulation, and the combination of scale and shift modulations. The results demonstrate that LM-LIIF achieves the highest PSNR performance when both scale and shift modulations are employed, and the computational

Table 6. Ablation study (PSNR(dB)/MACs on DIV2K validation set) on the latent MLP and the render MLP, using EDSR-b as the encoder, and LM-LIIF as the decoder.

Latent MLP	Render MLP	In-scale		Out-of-scale	
		×2	×4	×8	×16
2×256	2×128	34.66/ 236.91G	28.97/ 131.21G	25.37/ 104.57G	22.60/ 97.18G
2×256	3×64	34.67/ 232.18G	**29.00**/ 126.48G	**25.41**/ 99.85G	**22.63**/ 92.50G
2×256	5×32	**34.68**/ 197.37G	**29.00**/ 91.67G	25.41/ 65.11G	22.63/ 58.04G
4×256	9×16	34.64/ 258.37G	28.98/ 87.79G	25.40/ 45.04G	22.62/ 34.13G
2×256	9×16	34.66/ 171.85G	28.99/ 66.16G	25.40/ 39.65G	22.62/ 32.79G
2×192	7×16	34.66/ 123.10G	**29.00**/ 49.91G	25.40/ 31.55G	**22.63**/ 26.77G
2×128	5×16	34.63/ **79.76G**	28.98/ **35.02G**	25.39/ **23.79G**	22.62/ **20.83G**

cost remains nearly identical. Finally, we compare using the latent code (64 dimensions) from the encoder and the compressed latent code (16 dimensions) from the latent MLP as the input of the render MLP. The results indicate that using the compressed latent code can further reduce the computational cost for large scales. Based on these findings, we incorporate both scale and shift modulations as well as the compressed latent code in our final LMF models.

Network Structure. We evaluated different depths and dimensions of the latent MLP and the render MLP in Table 6. Notably, the compressed latent code is not used to ensure better alignment of depths and dimensions between the two MLPs. For the latent MLP, we observe that decreasing the depth of the latent MLP from 4 to 2 yields nearly the same PSNR performance, which suggests that the latent MLP does not require a deep architecture or a large number of parameters. However, when reducing the dimension of the latent MLP from 192 to 128, the PSNR performance begins to decrease. This observation confirms that HD computations are crucial for a powerful latent MLP.

In Table 6, we investigate the depths and dimensions of the render MLP by fixing the dimensions of the latent MLP and the latent modulation at 256. The results demonstrate that we can effortlessly reduce the dimension of the render MLP from 128 to 16, resulting in an impressive up to 66.26% reduction in computational cost, without compromising performance. When the latent modulation possesses a fixed dimension, it is consistently viable to utilize a lower dimensional rendering process, thereby minimizing the computational burden. All things considered, our final LM-LIIF model employs a 2-layer latent MLP with 208 dimensions (16 dimensions for the compressed latent code) and a 7-layer render MLP with 16 dimensions, optimizing computational efficiency without sacrificing performance.

Controllable Multi-Scale Rendering. In Table 7, we evaluate the MSE thresholds when creating the Scale2Mod ta-

Table 7. Ablation study (PSNR(dB)/MACs of rendering/MACs of decoder on DIV2K validation set) on MSE thresholds of CMSR, using EDSR-b as the encoder, and LM-LIIF as the decoder.

MSE threshold	In-scale		Out-of-scale	
	×2	×4	×8	×16
×	**34.65**/17.41G/ 125.32G	**28.99**/17.41G/ 44.38G	**25.40**/17.37G/ 24.10G	**22.62**/17.23G/ 18.90G
2×10^{-5}	**34.65**/16.30G/ 124.22G	**28.99**/13.67G/ 40.62G	**25.40**/12.41G/ 19.14G	**22.62**/3.48G/ 5.15G
1×10^{-4}	34.62/13.63G/ 121.55G	28.98/11.71G/ 38.69G	25.39/8.35G/ 15.08G	**22.62**/2.50G/ 4.17G
5×10^{-4}	34.42/**10.35G**/ 118.27G	28.93/**8.94G**/ 35.92G	25.38/**6.08G**/ 12.81G	22.61/**2.29G**/ 3.96G

ble for CMSR. Compared to the case without CMSR, a Scale2Mod table created with an MSE threshold of 2×10^{-5} achieves the same PSNR performance while reducing computational cost by 20.58% and 72.75% for scale ×8 and ×16. With lower precision requirements, *i.e.*, MSE thresholds of 1×10^{-4} and 5×10^{-4}, CMSR effectively reduces the computational cost even further while maintaining maximum PSNR performance. This adaptability is advantageous for real-world applications as the precision can be adjusted during testing without the need for model retraining. However, CMSR exclusively reduces the computational cost of the render MLP, which limits the computational cost reduction for small scales. Hence, in the Experiments section, we only employ CMSR with the MSE threshold 2×10^{-5} for out-of-distribution scales. More details and results regarding CMSR are provided in the supplementary material.

6. Conclusion

In this paper, we propose a Latent Modulated Function (LMF), which realizes a computational optimal paradigm of continuous image representation. LMF utilizes an MLP in latent space to generate the latent modulation of each latent code. This enables a modulated MLP in render space to perform effective rendering at any resolution. Moreover, based on the correlation between modulation intensity and signal complexity, we propose a Controllable Multi-Scale Rendering (CMSR) algorithm, offering the flexibility to balance the rendering precision and efficiency during testing. Experiments demonstrate that converting INR-based ASSR methods to LMF leads to substantial reductions in computational cost, runtime, and parameters, while preserving performance. Better forms of latent modulations and more efficient decoder architectures will be explored in future work.

7. Acknowledgement

References

[1] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 126–135, 2017. 5

[2] Jonathan T Barron, Ben Mildenhall, Matthew Tancik, Peter Hedman, Ricardo Martin-Brualla, and Pratul P Srinivasan. Mip-nerf: A multiscale representation for anti-aliasing neural radiance fields. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 5855–5864, 2021. 2

[3] Jonathan T Barron, Ben Mildenhall, Dor Verbin, Pratul P Srinivasan, and Peter Hedman. Mip-nerf 360: Unbounded anti-aliased neural radiance fields. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5470–5479, 2022. 2

[4] Marco Bevilacqua, Aline Roumy, Christine Guillemot, and Marie Line Alberi-Morel. Low-complexity single-image super-resolution based on nonnegative neighbor embedding. *British machine vision conference*, 2012. 5

[5] Jiezhang Cao, Qin Wang, Yongqin Xian, Yawei Li, Bingbing Ni, Zhiming Pi, Kai Zhang, Yulun Zhang, Radu Timofte, and Luc Van Gool. Ciaosr: Continuous implicit attention-in-attention network for arbitrary-scale image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023. 2, 3, 5, 6

[6] Hao Chen, Bo He, Hanyu Wang, Yixuan Ren, Ser Nam Lim, and Abhinav Shrivastava. Nerv: Neural representations for videos. *Advances in Neural Information Processing Systems*, 34:21557–21568, 2021. 2

[7] Hanting Chen, Yunhe Wang, Tianyu Guo, Chang Xu, Yiping Deng, Zhenhua Liu, Siwei Ma, Chunjing Xu, Chao Xu, and Wen Gao. Pre-trained image processing transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12299–12310, 2021. 1, 3

[8] Hao-Wei Chen, Yu-Syuan Xu, Min-Fong Hong, Yi-Min Tsai, Hsien-Kai Kuo, and Chun-Yi Lee. Cascaded local implicit transformer for arbitrary-scale super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 18257–18267, 2023. 2, 3

[9] Xiangyu Chen, Xintao Wang, Jiantao Zhou, Yu Qiao, and Chao Dong. Activating more pixels in image super-resolution transformer. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22367–22377, 2023. 1, 3

[10] Yinbo Chen and Xiaolong Wang. Transformers as meta-learners for implicit neural representations. In *European Conference on Computer Vision*, pages 170–187. Springer, 2022. 2

[11] Yinbo Chen, Sifei Liu, and Xiaolong Wang. Learning continuous image representation with local implicit image function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8628–8638, 2021. 1, 3, 4, 5, 6

[12] Chao Dong, Chen Change Loy, Kaiming He, and Xiaoou Tang. Learning a deep convolutional network for image super-resolution. In *European Conference on Computer Vision*, pages 184–199, 2014. 1, 3

[13] Emilien Dupont, Hrushikesh Loya, Milad Alizadeh, Adam Goliński, Yee Whye Teh, and Arnaud Doucet. Coin++: Neural compression across modalities. *arXiv preprint arXiv:2201.12904*, 2022. 2, 3

[14] Sicheng Gao, Xuhui Liu, Bohan Zeng, Sheng Xu, Yanjing Li, Xiaoyan Luo, Jianzhuang Liu, Xiantong Zhen, and Baochang Zhang. Implicit diffusion models for continuous super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10021–10030, 2023. 2, 3

[15] Zongyao He and Zhi Jin. Dynamic implicit image function for efficient arbitrary-scale image representation. *arXiv preprint arXiv:2306.12321*, 2023. 2, 3

[16] Zongyao He, Zhi Jin, and Yao Zhao. Srdrl: A blind super-resolution framework with degradation reconstruction loss. *IEEE Transactions on Multimedia*, 24:2877–2889, 2021. 1

[17] Xuecai Hu, Haoyuan Mu, Xiangyu Zhang, Zilei Wang, Tieniu Tan, and Jian Sun. Meta-sr: A magnification-arbitrary network for super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1575–1584, 2019. 3

[18] Jia-Bin Huang, Abhishek Singh, and Narendra Ahuja. Single image super-resolution from transformed self-exemplars. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5197–5206, 2015. 5

[19] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015. 5

[20] Wei-Sheng Lai, Jia-Bin Huang, Narendra Ahuja, and Ming-Hsuan Yang. Deep laplacian pyramid networks for fast and accurate super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 624–632, 2017. 1, 3

[21] Christian Ledig, Lucas Theis, Ferenc Huszár, Jose Caballero, Andrew Cunningham, Alejandro Acosta, Andrew Aitken, Alykhan Tejani, Johannes Totz, Zehan Wang, et al. Photo-realistic single image super-resolution using a generative adversarial network. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4681–4690, 2017. 1, 3

[22] Jaewon Lee and Kyong Hwan Jin. Local texture estimator for implicit representation function. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1929–1938, 2022. 2, 3, 5, 6

[23] Zhen Li, Jinglei Yang, Zheng Liu, Xiaomin Yang, Gwanggil Jeon, and Wei Wu. Feedback network for image super-resolution. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3867–3876, 2019. 1, 3

[24] Jingyun Liang, Jiezhang Cao, Guolei Sun, Kai Zhang, Luc Van Gool, and Radu Timofte. Swinir: Image restoration using swin transformer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 1833–1844, 2021. 3, 5

[25] Bee Lim, Sanghyun Son, Heewon Kim, Seungjun Nah, and Kyoung Mu Lee. Enhanced deep residual networks for single image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops*, pages 136–144, 2017. 1, 3, 5

[26] David Martin, Charless Fowlkes, Doron Tal, and Jitendra Malik. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 416–423, 2001. 5

[27] Ishit Mehta, Michaël Gharbi, Connelly Barnes, Eli Shechtman, Ravi Ramamoorthi, and Manmohan Chandraker. Modulated periodic activations for generalizable local functional representations. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14214–14223, 2021. 2, 3

[28] Ben Mildenhall, Pratul P Srinivasan, Matthew Tancik, Jonathan T Barron, Ravi Ramamoorthi, and Ren Ng. Nerf: Representing scenes as neural radiance fields for view synthesis. *Communications of the ACM*, 65(1):99–106, 2021. 2

[29] Ethan Perez, Florian Strub, Harm De Vries, Vincent Dumoulin, and Aaron Courville. Film: Visual reasoning with a general conditioning layer. In *Proceedings of the AAAI conference on artificial intelligence*, 2018. 3

[30] Christian Reiser, Songyou Peng, Yiyi Liao, and Andreas Geiger. Kilonerf: Speeding up neural radiance fields with thousands of tiny mlps. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 14335–14345, 2021. 2

[31] Vishwanath Saragadam, Jasper Tan, Guha Balakrishnan, Richard G Baraniuk, and Ashok Veeraraghavan. Miner: Multiscale implicit neural representation. In *European Conference on Computer Vision*, pages 318–333. Springer, 2022. 2

[32] Vincent Sitzmann, Julien Martel, Alexander Bergman, David Lindell, and Gordon Wetzstein. Implicit neural representations with periodic activation functions. *Advances in Neural Information Processing Systems*, 33:7462–7473, 2020. 2

[33] Ivan Skorokhodov, Savva Ignatyev, and Mohamed Elhoseiny. Adversarial generation of continuous images. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 10753–10764, 2021. 2, 3

[34] Gaochao Song, Qian Sun, Luo Zhang, Ran Su, Jianfeng Shi, and Ying He. Ope-sr: Orthogonal position encoding for designing a parameter-free upsampling module in arbitrary-scale image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 10009–10020, 2023. 2, 3

[35] Yannick Strümpler, Janis Postels, Ren Yang, Luc Van Gool, and Federico Tombari. Implicit neural representations for image compression. In *European Conference on Computer Vision*, pages 74–91. Springer, 2022. 2

[36] Ying Tai, Jian Yang, Xiaoming Liu, and Chunyan Xu. Memnet: A persistent memory network for image restoration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4539–4547, 2017. 1, 3

[37] Longguang Wang, Yingqian Wang, Zaiping Lin, Jungang Yang, Wei An, and Yulan Guo. Learning a single network for scale-arbitrary super-resolution. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4801–4810, 2021. 2, 3

[38] Xiaohang Wang, Xuanhong Chen, Bingbing Ni, Hang Wang, Zhengyan Tong, and Yutian Liu. Deep arbitrary-scale image super-resolution via scale-equivariance pursuit. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1786–1795, 2023.

[39] Xingqian Xu, Zhangyang Wang, and Humphrey Shi. Ultrasr: Spatial encoding is a missing key for implicit image function-based arbitrary-scale super-resolution. *arXiv preprint arXiv:2103.12716*, 2021.

[40] Jingyu Yang, Sheng Shen, Huanjing Yue, and Kun Li. Implicit transformer network for screen content image continuous super-resolution. *Advances in Neural Information Processing Systems*, 34:13304–13315, 2021.

[41] Jie-En Yao, Li-Yuan Tsao, Yi-Chen Lo, Roy Tseng, Chia-Che Chang, and Chun-Yi Lee. Local implicit normalizing flow for arbitrary-scale image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1776–1785, 2023. 2, 3

[42] Alex Yu, Vickie Ye, Matthew Tancik, and Angjoo Kanazawa. pixelnerf: Neural radiance fields from one or few images. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4578–4587, 2021. 2

[43] Sihyun Yu, Jihoon Tack, Sangwoo Mo, Hyunsu Kim, Junho Kim, Jung-Woo Ha, and Jinwoo Shin. Generating videos with dynamics-aware implicit generative adversarial networks. *arXiv preprint arXiv:2202.10571*, 2022. 2

[44] Roman Zeyde, Michael Elad, and Matan Protter. On single image scale-up using sparse-representations. In *Curves and Surfaces: 7th International Conference, Avignon, France, June 24-30, 2010, Revised Selected Papers 7*, pages 711–730. Springer, 2012. 5

[45] Huanrong Zhang, Jie Xiao, and Zhi Jin. Multi-scale image super-resolution via a single extendable deep network. *IEEE Journal of Selected Topics in Signal Processing*, 15(2):253–263, 2020. 1

[46] Yulun Zhang, Kunpeng Li, Kai Li, Lichen Wang, Bineng Zhong, and Yun Fu. Image super-resolution using very deep residual channel attention networks. In *European Conference on Computer Vision*, pages 286–301, 2018. 3

[47] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2472–2481, 2018. 1, 3, 5