Supplemental Material: Super-Resolution Neural Operator

This document provides additional details and results.

A. Dynamic basis update

In this section, we provide a principled discussion on the quasi-optimal property of the Galerkin-type attention mechanism, which tells that in each attention layer, SRNO can achieve the approximation capability that the Petrov-Galerkin projection can offer. Although some theorems have been proved in [1], we provide a systematic and complete discussion on the dynamic basis update processes in the query, test and value approximation spaces. The background knowledge about the Galerkin projection can be found in [4]

Let $g_\theta(\cdot) : \mathbb{R}^{n\times d} \to \mathbb{Q}_h, \{z_t : D_t \to \mathbb{R}^d\} \mapsto \{z_{t+1} : D_{t+1} \to \mathbb{R}^d\}$ be a learnable map that is the composition of the Galerkin-type attention operator and the FFN $\mathcal{O}$ used in SRNO, where $\mathbb{Q}_h \subset \mathcal{H}(\Omega_h) \subset \mathcal{H}$ is the current value space spanned by the column vectors of $Q = \mathbf{z}W_q$. Similarly, we can define the approximation spaces $\mathbb{V}_h$ and $\mathbb{K}_h$. Suppose f_h is the best approximation of image function f in current value space $\mathbb{Q}_h$, *i.e.*, $f_h = \arg\min_{q\in\mathbb{Q}_h}\|f - q\|_{\mathcal{H}}$, and then the approximation error between $g_\theta(\cdot)$ and f can be described as:

$$\|f - g_\theta(\mathbf{z})\|_{\mathcal{H}} \le \|f_h - g_\theta(\mathbf{z})\|_{\mathcal{H}} + \|f - f_h\|_{\mathcal{H}}. \tag{1}$$

Let $B(\cdot,\cdot) : \mathbb{V}_h \times \mathbb{Q}_h \to \mathbb{R}$ be a continuous bilinear form. Here we define, for $(u, v) \in \mathbb{Q}_h \times \mathbb{V}_h$, $B(u, v) := h^2\sum_{i=1}^n u(x_i)v(y_i)$. Under the settings of Galerkin-type attention [1], for any given $q \in \mathbb{Q}_h$, we have

$$\max_{v\in\mathbb{V}_h} \frac{|B(v, q)|}{\|v\|_{\mathcal{H}}} \ge c\|q\|_{\mathcal{H}}, \tag{2}$$

i.e., B is coercive on the current key space $\mathbb{V}_h$ with constant c. (1) can be reformulated as

$$\|f-g_\theta(\mathbf{z})\|_{\mathcal{H}} \le c^{-1}\max_{v\in\mathbb{V}_h}\frac{|B(v, f_h - g_\theta(\mathbf{z}))|}{\|v\|_{\mathcal{H}}}+\|f-f_h\|_{\mathcal{H}}. \tag{3}$$

Our purpose is to minimize (3) by optimizing the trainable parameters θ,

$$\min_\theta\max_{v\in\mathbb{V}_h}\frac{|B(v, f_h - g_\theta(\mathbf{z}))|}{\|v\|_{\mathcal{H}}} \le \min_{q\in\mathbb{Q}_h}\max_{v\in\mathbb{V}_h}\frac{|B(v, f_h - q)|}{\|v\|_{\mathcal{H}}}. \tag{4}$$

(3) and (4) show the approximation capacity of a Galerkin-type attention used in SRNO as the kernel integral operator. In SRNO, we are actually optimizing the basis functions of the current value space $\mathbb{Q}_h$ to approximate the best f_h. By the Riesz representation theorem [3], there exists a value-to-key linear map $\Pi : \mathbb{Q}_h \to \mathbb{V}_h$ such that $B(v, f_h) = \langle v, \Pi f_h\rangle$. In order to reveal the interactions among the bases of the three approximation spaces, we introduce the second bilinear form $A(\cdot,\cdot) : \mathbb{K}_h \times \mathbb{V}_h \to \mathbb{R}$ to substitute the inner product $\langle v, \Pi f_h\rangle$. In practice, the FFN $\mathcal{O}$, as a universal approximator in g_θ, helps the bilinear form $A(\cdot,\cdot)$ to approximate the inner product $\langle v, \Pi f_h\rangle$. We thus define the following problem to approximate the right hand side of (4) ($j = 1, \dots, d$):

$$\min_{q\in\mathbb{Q}_h}\max_{v\in\mathbb{V}_h}\frac{\|A(k_j, v) - B(q, v)\|}{\|v\|_{\mathcal{H}}}, \tag{5}$$

which involves solving the following operator equation system (finding $z_j \in \mathbb{Q}_h$ and $w \in \mathbb{V}_h$):

$$\begin{aligned}\langle w, v\rangle + B(v, z_j) = A(k_j, v), \quad \forall v \in \mathbb{V}_h,\\ B(w, q) = 0, \quad \forall q \in \mathbb{Q}_h,\end{aligned} \tag{6}$$

which is further equivalent to solve the following linear system:

$$\begin{pmatrix} M & B^T\\ B & 0\end{pmatrix}\begin{pmatrix}\boldsymbol{\mu}\\ \boldsymbol{\lambda}\end{pmatrix} = \begin{pmatrix}(V^TK)_j\\ 0\end{pmatrix}, \tag{7}$$

where $\mathbb{Q}_h$, $\mathbb{V}_h$ is formd by sets of basis $\{q_j(\cdot)\}_{j=1}^r$, $\{v_j(\cdot)\}_{j=1}^d$, respectively. $B \in \mathbb{R}^{r\times d}$, $M \in \mathbb{R}^{d\times d}$, and $(V^TK)_j \in \mathbb{R}^{d\times 1}$. $\boldsymbol{\mu} := \mu(w) = (\mu_{v_1}(w), \dots, \mu_{v_d}(w))^T$ is the vector representation for $w(\cdot) = \sum_{j=1}^d \mu_{v_j}(w)v_j(\cdot)$. Similiar to $\boldsymbol{\lambda}$ for z_j. It is straightforward to verify that $h^2(V^TK)_{ij} = A(k_j, v_i)$, $B_{ij} = B(v_j, q_i)$, $B = h^2(QU)^TV, M_{ij} = \langle v_i, v_j\rangle$. And then we can get:

$$\boldsymbol{\lambda} = (BM^{-1}B^T)^{-1}BM^{-1}(V^TK)_j, \tag{8}$$

if $rank(Q) = r \le rank(V) = d$, which is verified by our experiments. We multiply a permutation matrix $U \in \mathbb{R}^{d\times d}$ to Q, such that QU's first r columns form the value vector $(q_j(x_1), \dots, q_j(x_n))^T$ as the bases $\{q_j(\cdot)\}_{j=1}^r$ of $\mathbb{Q}_h$. Then

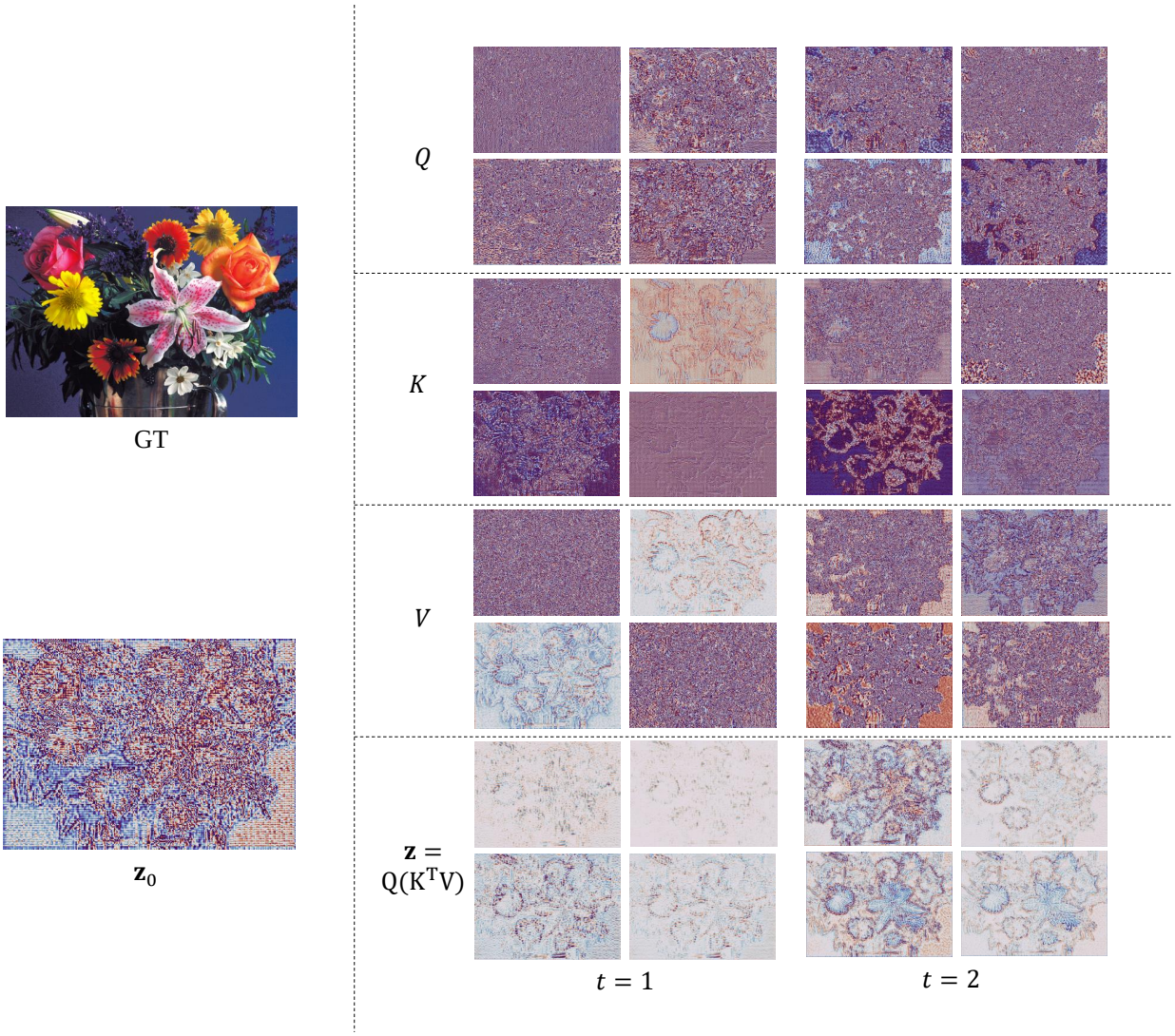

Figure 1. **Dynamic basis update**. The annotation $t = k$ refers to the iterative layer number. We display four basis function evaluation vectors (columns) of the matrices Q, K, and V, respectively, as well as the synthesized the latent representations $\mathbf{z}$

we multiply the permuted basis matrix QU with $\begin{pmatrix}\boldsymbol{\lambda} & 0\end{pmatrix} \in \mathbb{R}^d$, yielding

$$\begin{aligned} z_j &= h^2(QU)W(V^TK)_j, \\ W &= \Lambda \begin{pmatrix} B \\ 0 \end{pmatrix} M^{-1}, \end{aligned} \tag{9}$$

where $\Lambda = diag((BM^{-1}B^T), 0)$. The layer normalization scheme for V, K is used to mimick the matrix W.

The dynamic basis update rule, minimizer to (5), can be defined as:

$$z_j(\cdot) := \sum_{l=1}^{d} A(\widetilde{k}_j, \widetilde{v}_l) q_l(\cdot), \quad j = 1, \ldots, d, \tag{10}$$

where $\{\widetilde{k}_j\}, \{\widetilde{v}_l\}$ are the column vectors of layer normnalized matrices $\widetilde{K}, \widetilde{V}$.

The point-wise FFN $\mathcal{O} : \mathbb{R}^{d_z} \to \mathbb{R}^{d_z}$ introduces nonlinearties on one hand, and the positions concatenated in $\mathbf{z}$ enhance the bases on the other. In this way, the basis functions not only approximate the functions in the current value space, but also are being constantly enriched. Note that, in practice, we swap the matrix K, V to make this process closer to self-attention in [7]. In summary, our iterative process consist of two step: 1) the linear attention $Q(\widetilde{K}^T\widetilde{V})$ minimizes the (5) in the current value space; 2) the point-wise FFN $\mathcal{O}$ and position information for the latent representation $\mathbf{z}$ enrich the basis functions. The dynamic basis updating phenomenon is demonstrated in Fig.1. We observe that the basis function in the second layer appears to be more structured than in the previous layer, which verifies

the validity of our method.

B. Network Architecture

The Network architecture is shown in Fig.5. The input LR image f_{h_c} undergoes three phases to output the HR image f_{h_f} with the specified resolution: (a) Lifting the LR pixel values $a(\mathbf{x})$ on the set of coordinates $\mathbf{x} = \{x_i\}_{i=1}^{n_{h_f}}$ to a higher dimensional feature space by a CNN-based encoder E_ψ, constructing the latent representation $\hat{a}(x)$, and linearly transforming into the first layer's input $z_0(\mathbf{x})$. (b) kernel integrals composed of T layers of Galerkin-type attention, and (c) finally project to the RGB space. As to the feature encoder E_ψ, we employ EDSR-baseline [6], or RDN [8], both of which drop their upsampling layers, and their output channel dimensions $d_e = 64$. We employ the multi-head attention scheme in [7] by dividing the queries, keys and values into n_{heads} parts with each of dimension d_z/n_{heads}. In our implementation, $d_z = 256, n_{heads} = 16$, yielding 16-dimensional output values. We only use two iterations ($T = 2$) in the kernel integral operator, which already outperforms previous works, while keeping the running time advantage. Note that we utilize 1×1 convolutions to replace all the linear layers in SRNO, since they have a GPU-friendly data structure.

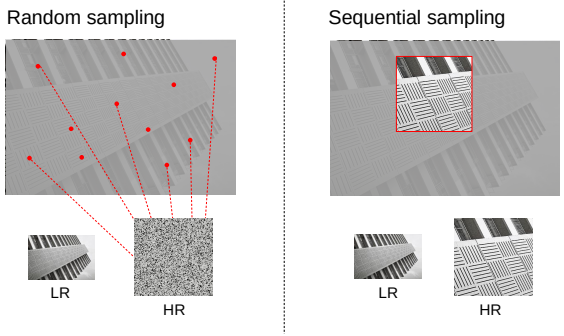

Figure 2. **Random vs. Sequential sampling**.

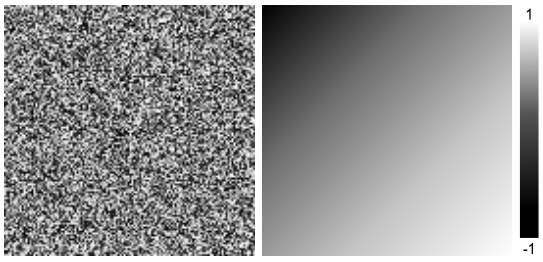

Figure 3. **Random v.s. sequential coordinates**.

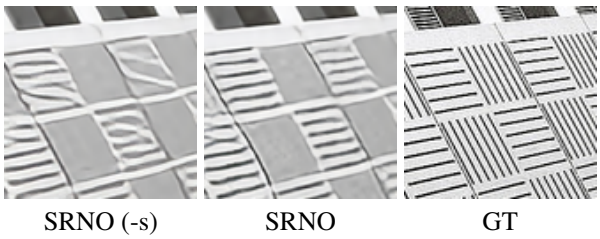

Figure 4. **Visual comparison on sampling methods**. Test on $\times 4$ scale. EDSR-baseline is used as the encoder.

C. Random vs. Sequential sampling

For a single batch, we crop B patches of sizes $\{128r^{(i)} \times 128r^{(i)}\}_{i=1}^{B}$ from the HR training images (one per each). The LR counterparts are downsampled using bicubic interpolation with the corresponding $r^{(i)}$. In order to keep the consistent dimensions of the LR patches, sharing a common supervisory HR signal in a single batch, we sample 128^2 HR pixels and calculate the corresponding fractional coordinates on the coarse grid associated with $r^{(i)}$. Figure 2 shows two different ways to sample function values. Experiments, in Tab.1, Tab.2 and Fig.4, verify that the random sampling method achieves better performance than the sequential sampling. These results show that using random sampling method can capture a more comprehensive representation for an image function, which is attributed to the fact that random coordinates, as demonstrated in Fig.3, contain some extra and useful high-frequency information for SR reconstruction.

D. Additional Results

We further compare our SRNO to LIIF, LTE on several images in Fig.6. It can be ovserved from the zoom-in regions that our SRNO consistently produces clearer and finer details than others.

Method	In-distribution			Out-of-distribution				
	×2	×3	×4	×6	×12	×18	×24	×30
Bicubic	31.01	28.22	26.66	24.82	22.27	21.00	20.19	19.59
EDSR-baseline-LTE [5]	34.72	31.02	29.04	26.81	23.78	22.23	21.24	20.53
EDSR-baseline-SRNO	**34.85**	**31.11**	**29.16**	**26.90**	**23.84**	**22.29**	**21.27**	**20.56**
EDSR-baseline-SRNO (-s)	34.79	31.07	29.09	26.84	23.80	22.26	21.24	20.54

Table 1. **Random vs. Sequential sampling of SRNO on DIV2K validation set (PSNR (dB))**. The best performance are bolded. All methods are trained with continuous random scales uniformly sampled in $\times 1$–$\times 4$. -s refers to using sequential sampling when training.

Dataset	Method	In-distribution			Out-of-distribution	
		×2	×3	×4	×6	×8
Set5	EDSR-baseline-LTE [5]	38.03	34.48	32.27	28.96	27.04
	EDSR-baseline-SRNO	**38.15**	**34.53**	**32.39**	**29.06**	**27.06**
	EDSR-baseline-SRNO (-s)	38.12	34.50	32.37	28.96	27.04
Set14	EDSR-baseline-LTE [5]	33.71	30.41	28.67	26.49	24.98
	EDSR-baseline-SRNO	**33.83**	**30.50**	**28.79**	**26.55**	**25.05**
	EDSR-baseline-SRNO (-s)	33.79	30.42	28.71	26.52	25.00
B100	EDSR-baseline-LTE [5]	32.22	29.15	27.63	25.87	24.83
	EDSR-baseline-SRNO	**32.28**	**29.20**	**27.68**	**25.91**	**24.88**
	EDSR-baseline-SRNO (-s)	32.25	29.18	27.65	25.90	24.85
Urban100	EDSR-baseline-LTE [5]	32.29	28.32	26.25	23.84	22.52
	EDSR-baseline-SRNO	**32.60**	**28.56**	**26.50**	**24.08**	**22.70**
	EDSR-baseline-SRNO (-s)	32.50	28.51	26.39	23.95	22.61

Table 2. **Random vs. Sequential sampling of SRNO on benchmark datasets (PSNR (dB))**. The best performances are in bold. All methods are trained with continuous random scales uniformly sampled in $\times 1$–$\times 4$. -s refers to using sequential sampling when training.

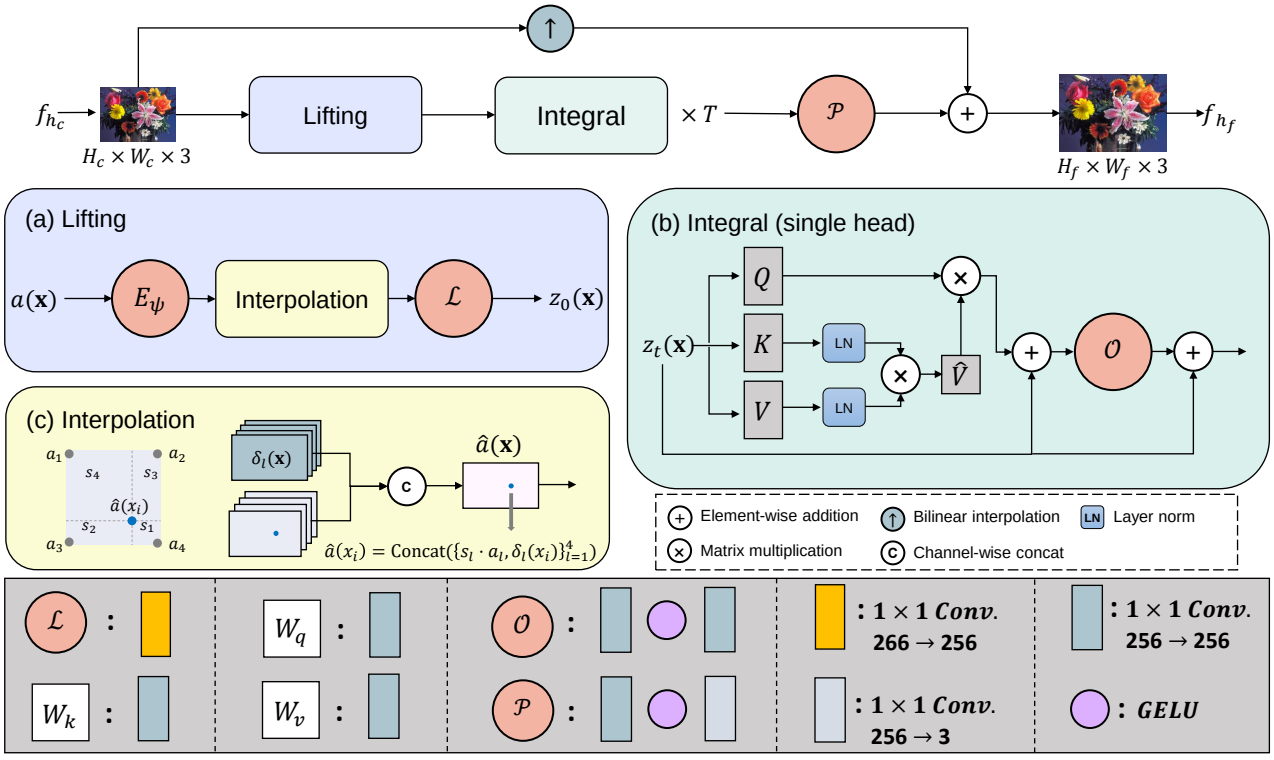

Figure 5. **Super-resolution neural operator (SRNO) architecture for continuous SR.** Encoder E_ψ's structure is omited.

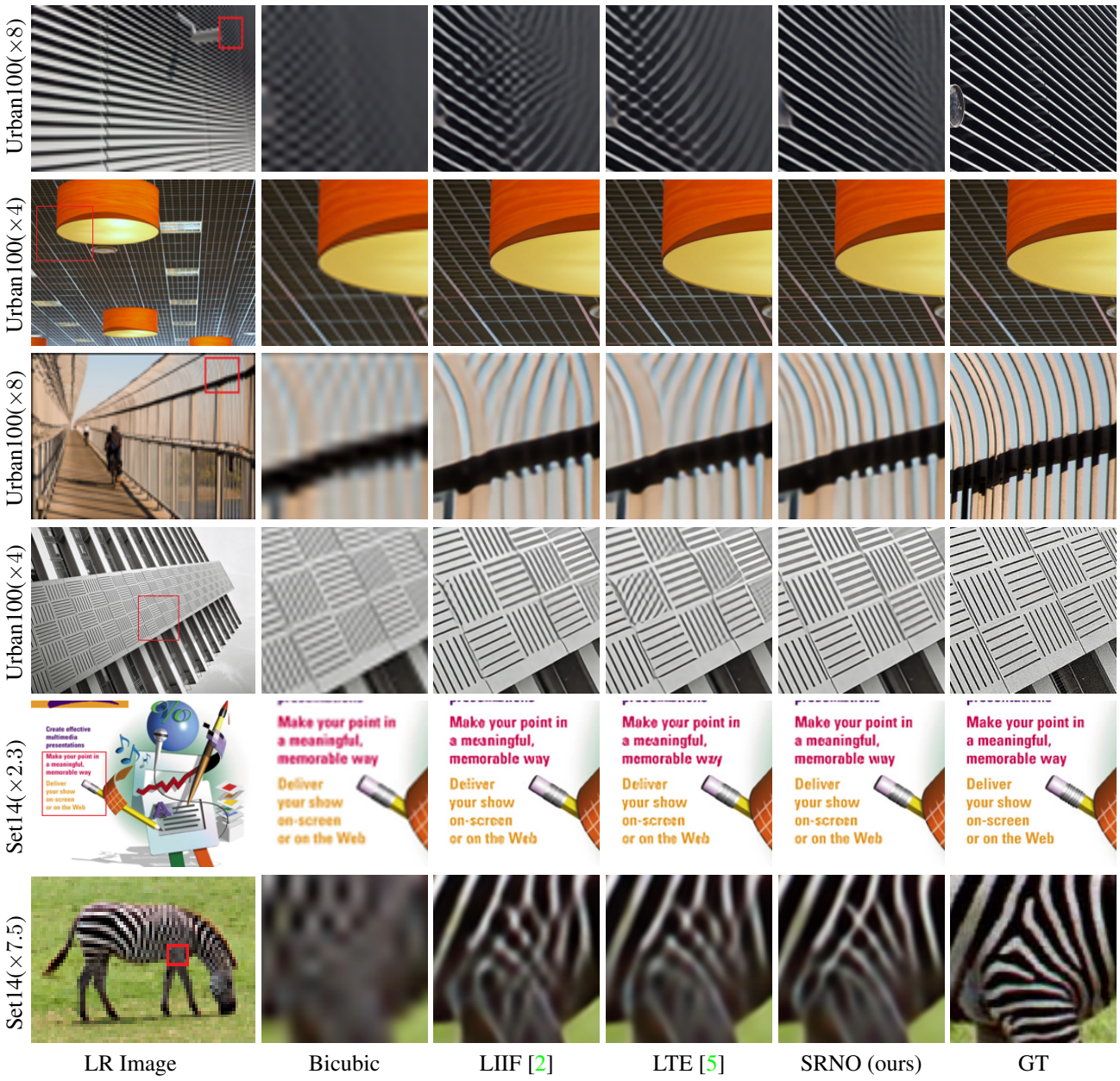

Figure 6. **Visual comparison on other zero-shot SR**. The boxes in the first column indicate the areas that the close-ups on the right display. All methods are trained with continuous random scales in $\times 1$–$\times 4$. RDN is used as the encoder for all methods.

References

[1] Shuhao Cao. Choose a transformer: Fourier or galerkin. *Advances in Neural Information Processing Systems*, 34:24924–24940, 2021. 1

[2] 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. 5

[3] Philippe G Ciarlet. *Linear and nonlinear functional analysis with applications*, volume 130. Siam, 2013. 1

[4] Alexandre Ern and Jean-Luc Guermond. *Theory and practice of finite elements*, volume 159. Springer, 2004. 1

[5] 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. 4, 5

[6] 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 conference on computer vision and pattern recognition workshops*, pages 136–144, 2017. 3

[7] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2, 3

[8] Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image super-resolution. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2472–2481, 2018. 3