Low-Resolution Editing is All You Need for High-Resolution Editing

Supplementary Material

A. Pseudo-code of ScaleEdit

We provide an abstract overview of the proposed method by outlining its core procedure in Algorithm 1.

Algorithm 1 ScaleEdit

Inputs: Conditioning images $I_{\text{src}}^{\text{high}}$, $I_{\text{src}}^{\text{low}}$, $I_{\text{ref}}^{\text{low}}$, hyperparameter τ
// 1. Forward process
for $i \leftarrow 1, \cdots, NM$ **do**
  Get $\{\mathbf{x}_t^{\text{high}}[i]\}_{t=0}^{T}, \{\mathbf{x}_t^{\text{low}}[i]\}_{t=0}^{T}, \{\mathbf{y}_t^{\text{low}}[i]\}_{t=0}^{T}$
  $\tilde{\mathbf{x}}_T[i] \leftarrow \mathbf{x}_T^{\text{low}}[i]$, $\tilde{\mathbf{y}}_T[i] \leftarrow \mathbf{y}_T^{\text{low}}[i]$
end for
// 2. Transfer function optimization
for $t \leftarrow T, \cdots, 1, \quad i \leftarrow 1, \cdots, NM$ **do**
  if $t \leq \tau$ **then**
    Optimize ϕ_θ using Eq. (6)
    $\Delta\mathbf{h}_t[i] \leftarrow \phi_\theta(\mathbf{h}_t[i], t)$
  else
    $\Delta\mathbf{h}_t[i] \leftarrow \mathbf{0}$
  end if
  Get $\tilde{\mathbf{x}}_{t-1}[i]$ using Eq. (7)
end for
// 3. Detail injection with synchronization
for $t \leftarrow T, \cdots, 1$ **do**
  Get $\tilde{\mathbf{y}}_{t-1}[i]$ using Eq. (9) for $1 \leq i \leq NM$
  if $t \leq \tau$ **then**
    Get $\tilde{\mathbf{y}}_t^{\text{rsp}}[i]$ using Eq. (13) for $1 \leq i \leq NM$
    Get $\tilde{\mathbf{y}}_{t-1}[i]$ using Eq. (14) for $1 \leq i \leq NM$
  end if
end for
$I_{\text{ref}}^{\text{high}} \leftarrow \text{Decode}(\text{Composite}(\{\tilde{\mathbf{y}}_0[i]\}_{i=1}^{NM}))$
Output: Target image $I_{\text{ref}}^{\text{high}}$

B. Detailed explanation on synchronization

To synchronize adjacent patches, we introduce an auxiliary latent $\tilde{\mathbf{A}}_t[i, i+1]$, constructed by spatially blending the bottom half of $\tilde{\mathbf{y}}_t^{\text{rsp}}[i]$ and the upper half of $\tilde{\mathbf{y}}_t^{\text{rsp}}[i+1]$ at timestep t. Since each patch is denoised independently, their latent trajectories may diverge, often producing visible discontinuities along their boundary. However, $\tilde{\mathbf{A}}_t[i, i+1]$ contains the boundary region, and therefore its Tweedie estimate may naturally captures a smoother transition between the two patches. By blending the Tweedie estimate of an auxiliary patch with those of the original patches, the boundary between neighboring patches becomes more coherent, reducing artifacts and enhancing spatial continuity.

For the spatial blending, we first apply Eq. (2) to obtain the original patches' Tweedie estimates:

$$\hat{\mathbf{x}}_{t\to0}^{\text{aux}}[i, i+1] = \hat{\mathbf{x}}_0(\tilde{\mathbf{A}}_t[i, i+1], t) \tag{15}$$

$$\hat{\mathbf{x}}_{t\to0}[i] = \hat{\mathbf{x}}_0(\tilde{\mathbf{y}}_t^{\text{rsp}}[i], t). \tag{16}$$

Then we define $L(\hat{\mathbf{x}}_{t\to0}[i], \hat{\mathbf{x}}_{t\to0}^{\text{aux}}[i, i+1])$, a blended Tweedie estimate that is used to swap the $\hat{\mathbf{x}}_0$ term in Eq. (1) during the reverse process of $\tilde{\mathbf{y}}_t^{\text{rsp}}[i]$. Let $\mathcal{T}_1$ be the vertical translation operator defined in $[0, H_p] \times [0, W_p]$ as follows:

$$\mathcal{T}_1 f(u, v) = \begin{cases} 0, & 0 \leq v < H_p/2, \\ f(u, v - H_p/2), & H_p/2 \leq v \leq H_p, \end{cases} \tag{17}$$

where W_p and H_p denote the patch width and height, respectively. Thus, $\mathcal{T}_1\hat{\mathbf{x}}_{t\to0}^{\text{aux}}[i, i+1]$ repositions $\hat{\mathbf{x}}_{t\to0}^{\text{aux}}[i, i+1]$ so that its top edge aligns with the midpoint of $\hat{\mathbf{x}}_{t\to0}[i]$. Then we define a spatial weight mask $\mathbf{M}_1(u, v, t)$ for smooth transition across the overlap:

$$\mathbf{M}_1(u, v, t) = \begin{cases} 0, & 0 \leq v < H_p/2, \\ \frac{v - H_p/2}{H_p/2} \cdot \left(1 - \frac{t}{\tau}\right), & H_p/2 \leq v \leq H_p. \end{cases} \tag{18}$$

The blended Tweedie estimate is then computed as

$$\begin{aligned} &L(\hat{\mathbf{x}}_{t\to0}[i], \hat{\mathbf{x}}_{t\to0}^{\text{aux}}[i, i+1]) \\ &= (1 - \mathbf{M}_1) \odot \hat{\mathbf{x}}_{t\to0}[i] + \mathbf{M}_1 \odot \mathcal{T}_1\hat{\mathbf{x}}_{t\to0}^{\text{aux}}[i, i+1]. \end{aligned} \tag{19}$$

For the rest of the paragraph, we detail the case of $\tilde{\mathbf{y}}_t^{\text{rsp}}[i+1]$. Let $\mathcal{T}_2$ and $\mathbf{M}_2(u, v, t)$ defined in the vertically-flipped manner compared to $\mathcal{T}_1$ and $\mathbf{M}_1(u, v, t)$:

$$\mathcal{T}_2 f(u, v) = \begin{cases} f(u, v + H_p/2), & 0 \leq v \leq H_p/2 \\ 0, & H_p/2 < v < H_p, \end{cases} \tag{20}$$

$$\mathbf{M}_2(u, v, t) = \begin{cases} \frac{H_p/2 - v}{H_p/2} \cdot \left(1 - \frac{t}{\tau}\right), & 0 \leq v \leq H_p/2 \\ 0, & H_p/2 < v < H_p. \end{cases} \tag{21}$$

Subsequently, the blended Tweedie estimate is calculated as:

$$\begin{aligned} &L(\hat{\mathbf{x}}_{t\to0}[i+1], \hat{\mathbf{x}}_{t\to0}^{\text{aux}}[i, i+1]) \\ &= (1 - \mathbf{M}_2) \odot \hat{\mathbf{x}}_{t\to0}[i+1] + \mathbf{M}_2 \odot \mathcal{T}_2\hat{\mathbf{x}}_{t\to0}^{\text{aux}}[i, i+1]. \end{aligned} \tag{22}$$

C. Implementation details

Latent decoding. After denoising $N \times M$ latent patches, we first merge them into a single latent tensor. The merged latent is then passed through the pretrained VAE decoder [1] to obtain the final high-resolution image $I_{\text{ref}}^{\text{high}}$.

Adaptation to Stable Diffusion. For Stable Diffusion [8] v2.1-base, the upsampling path of the U-Net [9] consists of four upsampling blocks, each composed of three sub-blocks. We insert a 1×1 convolution layer into the last sub-block of each upsampling block, resulting in four additional 1×1 convolution layers in total.

Adaptation to FLUX. For FLUX.1-dev [2], we attach the detail enhancement module to the linear layer located immediately after the last single-stream block, which is followed by a Layer Normalization operation. Since FLUX is based on flow matching [3, 4], we apply ScaleEdit with several additional modifications. The forward and reverse processes follow deterministic ODE dynamics, defined as:

$$\mathbf{x}_{t+1}^{\text{fwd}} = \mathbf{x}_t^{\text{fwd}} + (\sigma_{t+1} - \sigma_t) \cdot \mathbf{v}_\theta(\mathbf{x}_t^{\text{fwd}}, t), \tag{23}$$

$$\mathbf{x}_{t-1}^{\text{rev}} = \mathbf{x}_t^{\text{rev}} + (\sigma_{t-1} - \sigma_t) \cdot \mathbf{v}_\theta(\mathbf{x}_t^{\text{rev}}, t), \tag{24}$$

where $\mathbf{v}_\theta(\cdot, \cdot)$ denotes the pretrained vector field prediction network. The clean data sample is estimated as:

$$\hat{\mathbf{x}}_0(\mathbf{x}_t, t) = \mathbf{x}_t - \sigma_t \mathbf{v}_\theta(\mathbf{x}_t, t), \tag{25}$$

which corresponds to the Tweedie estimate in diffusion models. In addition, the reverse process using the blended latent differs from that of diffusion models. We calculate the blended latent $L(\hat{\mathbf{x}}_{t\to0}[i], \hat{\mathbf{x}}^{\text{aux}}_{t\to0}[i, i+1])$ using the same blending operation described in Sec. 4.5.1 of the main paper, which corresponds to the blended Tweedie estimate in diffusion models. We then compute the modified vector field using the blended latent as follows:

$$\mathbf{v}'(t)[i] = \frac{\tilde{\mathbf{y}}_t^{\text{rsp}}[i] - L(\hat{\mathbf{x}}_{t\to0}[i], \hat{\mathbf{x}}^{\text{aux}}_{t\to0}[i, i+1])}{\sigma_t}, \tag{26}$$

Finally, we perform the reverse process using the modified vector field:

$$\tilde{\mathbf{y}}_{t-1}[i] = \tilde{\mathbf{y}}_t^{\text{rsp}}[i] + (\sigma_{t-1} - \sigma_t) \cdot \mathbf{v}'(t)[i]. \tag{27}$$

In practice, to achieve accurate reconstruction, we cache the output of each attention layer in the forward process and reuse it in the reverse process.

D. Additional ablation studies

Impact of hyperparameter τ. As described in Algorithm 1, ScaleEdit utilizes a hyperparameter τ. To investigate its effect, we conduct an ablation study by varying τ in $15, 25, 35$ and evaluate the results using the same metrics as the main experiment on a set of 60 (image, instruction) pairs. As shown in Table 2, the default setting $\tau = 15$ achieves the best performance across all configurations.

Table 2. Ablation study on τ values.

τ	HaarPSI ↑	M-MSE ↓	M-SSIM ↑	M-PSNR ↑	LPIPS ↓
15	0.335	0.042	0.573	17.430	0.472
25	0.326	0.043	0.561	17.097	0.471
35	0.308	0.044	0.537	16.308	0.496

Ablation on the design choice of transfer function. We first analyze the effect of layers by varying the index of the sub-block used within each block of the U-Net [9] that are used for the detail transfer module. Specifically, for each stage (down/middle/up), we select N-th sub-block from all blocks in that stage. Variant of ϕ_θ is also evaluated by replacing the convolution with constant mapping. Evaluation is conducted using a total of 60 (image, instruction) pairs. Tab. 3 shows our design (Up #3) performs best; while maintaining the overall perceptual similarity (measured by LPIPS [11]), it also preserves the intended editing effects (quantified by CLIP-Sim [7] between target image and target prompt). Alternatives still outperform most baselines but fall short. Since the last (3rd) sub-blocks in up stage operate at high-resolution, they are most effective at transferring fine-grained details needed for high-resolution generation.

Table 3. Ablation study on design of detail transfer module.

Method	Down #1	Down #2	Middle	Up #1	Up #2	**Up #3**	Constant
LPIPS ↓	0.1718	0.1746	0.1913	**0.1648**	0.1663	**0.1648**	0.1781
CLIP-Sim. ↑	0.2676	0.2677	0.2674	0.2681	0.2681	**0.2684**	0.2683

Ablation on synchronization strategy. In Figure 8, we illustrate the effect of the proposed synchronization method. As shown, naive patch sampling without synchronization brings noticeable edge artifacts along patch boundaries, whereas our method effectively mitigates these artifacts.

E. Receptive field of generative models

In this work, we leverage a low-resolution image generation model to perform high-resolution image editing in a patch-wise manner. Although FreeScale [6] modifies internal components of a low-resolution diffusion model (*e.g.* via dilated convolutions) to generate high-resolution outputs, such architectural changes do not guarantee a strong high-resolution image prior, as the model is not trained on high-resolution image datasets.

In contrast, our patch-wise approach is well-suited for extracting detailed information. Each patch matches the resolution of the low-resolution model's receptive field (*e.g.* 512×512 for the pretrained Stable Diffusion [8]), enabling faithful detail reconstruction without modifying the underlying generator. The main challenge in the modified framework lies in generating details that the model was not originally trained to produce. For these reasons, we adopt a patch-wise strategy grounded in low-resolution image priors instead of modifying diffusion architectures to synthesize high-resolution content directly.

F. Additional results

We show the additional results of ScaleEdit in Figure 7 and 9. Figure 7 shows that our method is even applicable to transformer-based [10] FLUX model [2], demonstrating the robustness and generalizability of our method across backbone architectures. Then, we show the additional 1K- and 2K-editing results obtained with the pretrained Stable Diffusion [8] in Figure 9. We emphasize that the proposed method effectively transfers the fine-grained details of the source image into the target image.

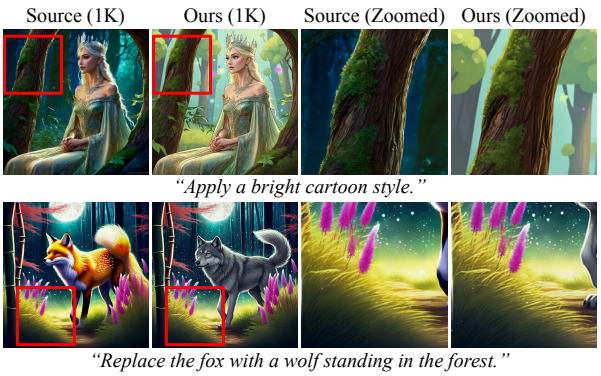

Figure 7. **[Best visualized when magnified.]** Qualitative results of our method combined with the pretrained FLUX.1-dev [2] model.

G. Discussion on computational cost

We evaluate the computational efficiency of our method against baselines in terms of runtime and GPU memory usage using a single NVIDIA A6000 GPU on 1K-editing scenario. While baselines in half-precision typically require 0.74–88.06 seconds and 8.40–36.00 GB of VRAM, our method currently operates in full-precision to ensure numerical stability during optimization, requiring 20.29 GB. Although our method currently has a higher overhead, we anticipate that further refinement of the implementation will enable comparable efficiency in half-precision without compromising performance.

We also note that our framework is compatible with Null-text Inversion (NTI) [5] for accurate reconstruction. The runtime without and with NTI are 234.77 and 635.51 seconds, respectively; we clarify that latter case's overhead is mainly due to NTI's iterative optimization rather than our core components. Ultimately, these results represent a justifiable trade-off for the state-of-the-art performance our method achieves in Table 1 and Figure 4, consistently outperforming baselines in high-resolution image editing. Note that for the evaluation results reported in the main paper, all methods were executed in full-precision whenever possible, unless this led to out-of-memory errors.

H. Limitations

Our method may produce suboptimal results due to the limited performance of the pretrained generative models [2, 8]. Furthermore, since the proposed method relies on the low-resolution reference image $I_{\text{ref}}^{\text{low}}$ generated by existing low-resolution image-to-image translation methods, some artifacts introduced by these editing methods can propagate to the final high-resolution output $I_{\text{ref}}^{\text{high}}$, potentially leading to visible artifacts.

I. Societal impacts

The proposed method may generate some harmful images due to the imperfections of the underlying pretrained generative models [2, 8]. Rare cases of undesired or inappropriate results may arise, especially when the generative prior itself produces such outputs under challenging conditions.

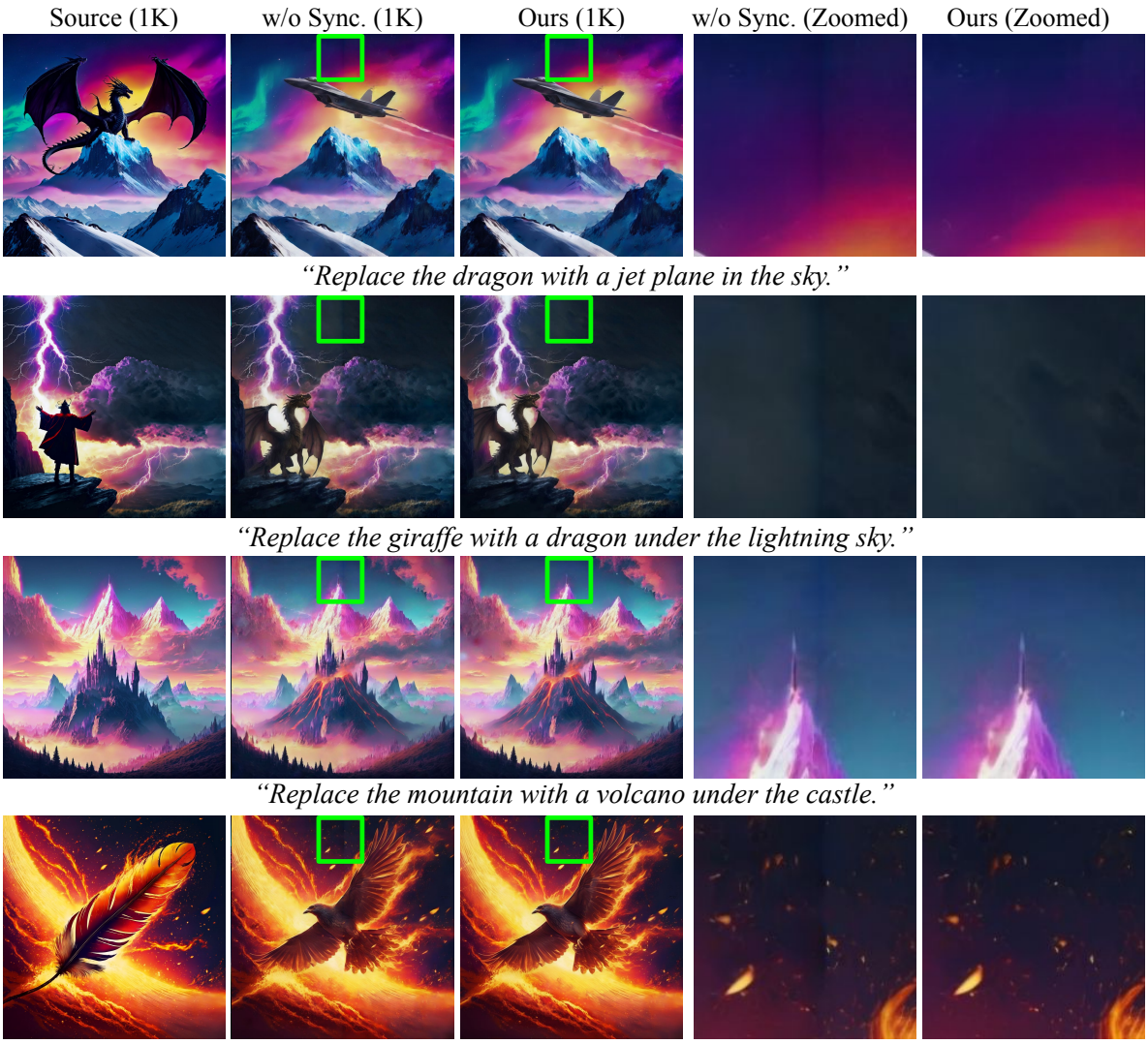

Figure 8. **[Best visualized when magnified.]** Effectiveness of the proposed synchronization strategy. While images sampled without synchronization incorporates a significant artifact on the patch boundaries, our method effectively alleviates the edge artifacts.

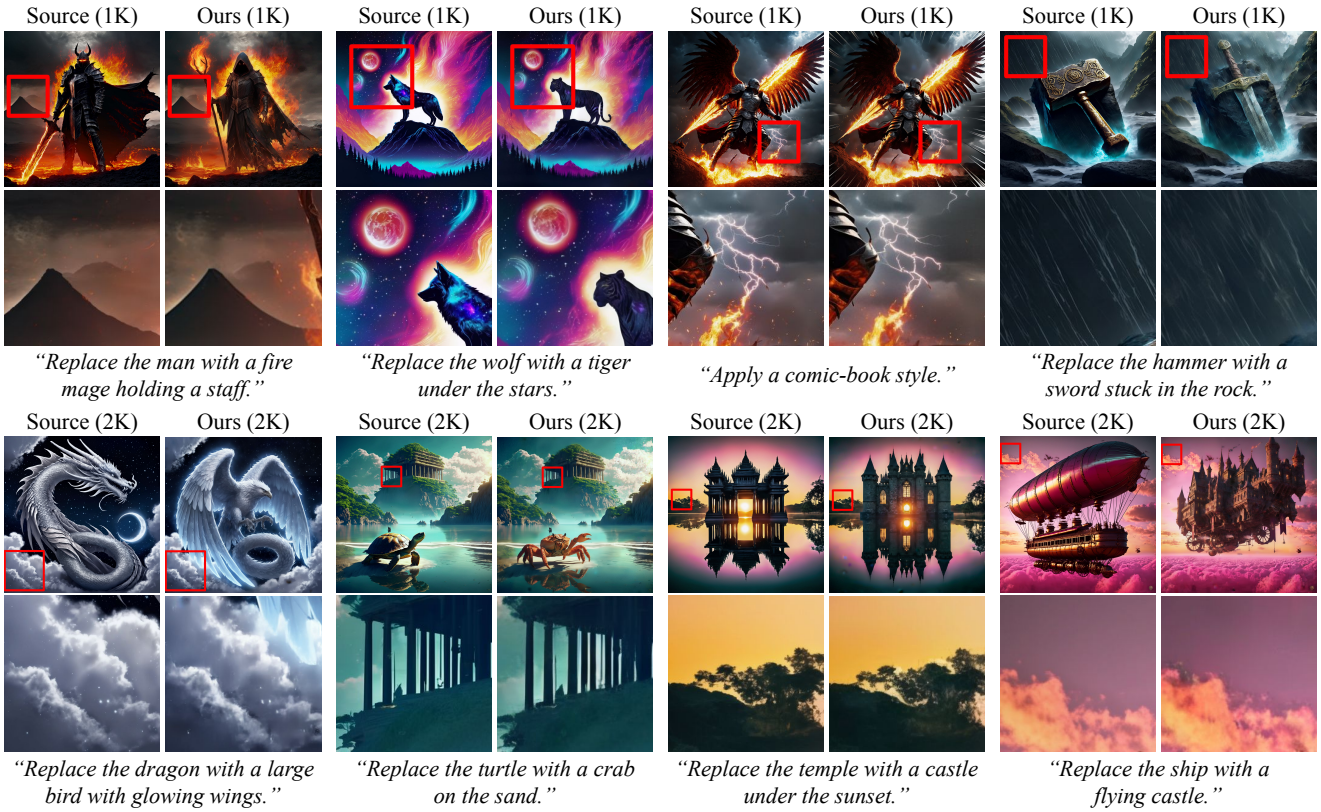

Figure 9. **[Best visualized when magnified.]** We visualize the additional results of 1K-editing and 2K-editing. By conditioning on low-resolution reference images, our method is successfully synthesizes high-resolution edited images.

References

[1] Diederik P Kingma and Max Welling. Auto-encoding variational bayes. *arXiv:1312.6114*, 2013. 1

[2] Black Forest Labs. Flux. https://github.com/black-forest-labs/flux, 2024. 2, 3

[3] Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *ICLR*, 2023. 2

[4] Xingchao Liu, Chengyue Gong, and Qiang Liu. Flow straight and fast: Learning to generate and transfer data with rectified flow. *ICLR*, 2023. 2

[5] Ron Mokady, Amir Hertz, Kfir Aberman, Yael Pritch, and Daniel Cohen-Or. Null-text inversion for editing real images using guided diffusion models. In *CVPR*, 2023. 3

[6] Haonan Qiu, Shiwei Zhang, Yujie Wei, Ruihang Chu, Hangjie Yuan, Xiang Wang, Yingya Zhang, and Ziwei Liu. Freescale: Unleashing the resolution of diffusion models via tuning-free scale fusion. In *ICCV*, 2025. 2

[7] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *ICML*, 2021. 2

[8] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *CVPR*, 2022. 2, 3

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, 2015. 2

[10] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *NIPS*, 2017. 3

[11] Richard Zhang, Phillip Isola, Alexei A Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *CVPR*, 2018. 2