Zoom-to-Inpaint: Image Inpainting with High Frequency Details

Soo Ye Kim[1,2†] Kfir Aberman[2] Nori Kanazawa[2] Rahul Garg[2] Neal Wadhwa[2]
Huiwen Chang[2] Nikhil Karnad[2] Munchurl Kim[1] Orly Liba[2]

[1]KAIST
Daejeon, Republic of Korea

[2]Google Research
Mountain View CA, USA

Abstract

Although deep learning has enabled a huge leap forward in image inpainting, current methods are often unable to synthesize realistic high-frequency details. In this paper, we propose applying super resolution to coarsely reconstructed outputs, refining them at high resolution, and then downscaling the output to the original resolution. By introducing high-resolution images to the refinement network, our framework is able to reconstruct finer details that are usually smoothed out due to spectral bias – the tendency of neural networks to reconstruct low frequencies better than high frequencies. To assist training the refinement network on large upscaled holes, we propose a progressive learning technique in which the size of the missing regions increases as training progresses. Our zoom-in, refine and zoom-out strategy, combined with high-resolution supervision and progressive learning, constitutes a framework-agnostic approach for enhancing high-frequency details that can be applied to other inpainting methods. We provide qualitative and quantitative evaluations along with an ablation analysis to show the effectiveness of our approach, which outperforms state-of-the-art inpainting methods.

1. Introduction

Image inpainting is a long-standing problem in computer vision and has many graphics applications. The goal of the problem is to fill in missing regions in a masked image, such that the output is a natural completion of the captured scene with plausible semantics, and realistic details and textures. The latter can be achieved with traditional inpainting methods that copy patches of valid pixels, e.g., PatchMatch [3], thus preserving the textural statistics of the surrounding regions. Nevertheless, the inpainted results often lack semantic context and do not blend well with the rest of the image.

With the advent of deep learning, inpainting neural networks are commonly trained in a self-supervised fashion, by generating random masks and applying them to the full image to produce masked images that are used as the network's input. These networks are able to produce semantically plausible results thanks to abundant training data. However, the results often do not have realistic details and textures, largely due to the finding of a *spectral bias* [36] in neural networks. That is, high frequency details are diffi-

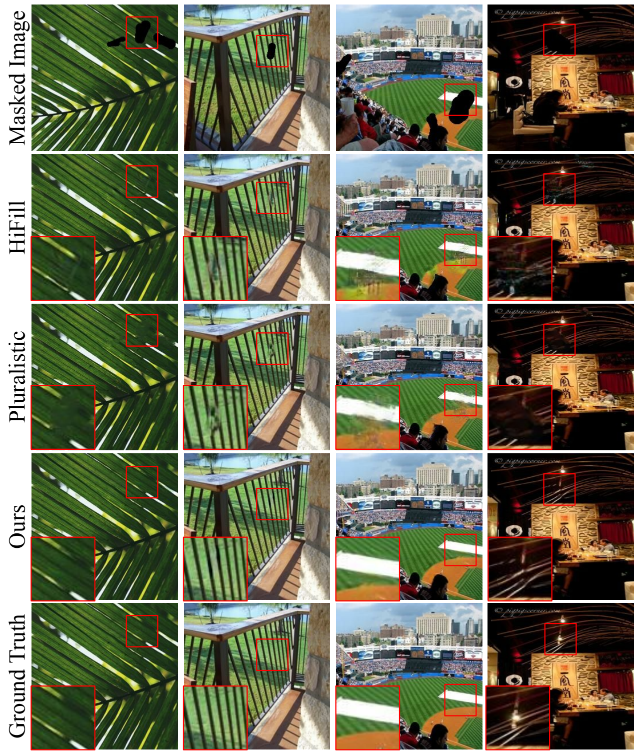


Figure 1: Qualitative comparison to recent inpainting methods, HiFill [48] and Pluralistic [53]. Our method correctly reconstructs high frequency details, e.g, fine textures and narrow structures, and preserves the continuity and the orientation of edges.

†This work was done during an internship at Google Research.

cult to learn as neural networks are biased towards learning low frequency components. This is especially problematic when training neural networks for image restoration tasks such as image inpainting, since high frequency details must be generated for realistic results.

Recent neural network architectures for image inpainting consist of a coarse network that first generates a coarsely filled-in result, and a refinement network that corrects and refines the coarse output for better quality [48, 49, 50]. In this paper, with the goal of generating more realistic high frequency details, we propose to refine after *zooming in*, therefore refining the image at a resolution higher than the target resolution. This approach effectively reduces the spectral bias at the desired resolution and injects more high frequency details into the resulting image. We show that adding a simple bicubic upsampling component between the coarse and refinement networks improves inpainting results, and using a super-resolution (SR) network to upscale the intermediate result improves the results even further. Conventional convolutional neural networks (CNNs) for image and video restoration tend to reduce the size of the image (as in multi-scale architectures [20, 30]), or feature map (eg. U-Net [38]), to increase the receptive field of the network. However, the benefits of going *above* the target resolution have not been previously explored.

Increasing the image resolution, that is *zooming in*, allows the refinement network to correct local irregularities at a finer level and to learn from high-resolution (HR) labels. However, the HR refinement network can be more difficult to train than low-resolution (LR) refinement network due to a larger mask and more missing pixels in the HR input. We overcome this difficulty by a *progressive learning strategy*, where the framework is first trained on smaller masks before training on larger masks that may span the entire image. Moreover, to further enhance the high frequency details, we use an additional gradient loss [12] that minimizes the image gradients of the difference between the prediction and the ground truth. We believe that these fundamental strategies can benefit any existing inpainting network.

In summary, our contributions are three-fold:

- We propose a novel inpainting framework that includes an SR network to *zoom in* allowing refinement at HR, training with HR labels, and enhances the generation of high-frequency details in the final inpainted output.
- We propose a progressive learning strategy for inpainting to aid convergence with larger masks.
- We explore the usage of a gradient loss for inpainting to further improve textural details.

2. Related Work

2.1. Image Inpainting

Traditional image inpainting methods can be largely classified into three types: (i) Propagation-based approaches that gradually fill in the missing regions from known pixel values at hole boundaries [5, 43], (ii) Markov Random Field (MRF) approaches optimizing discrete MRFs [22, 34], or (iii) patch-based approaches that search for plausible patches outside of the hole to be pasted into the missing region [3, 4] similar to texture synthesis algorithms [10, 11]. These types of approaches exploit information already present in the input image.

Deep-learning-based inpainting methods leverage information external to any one specific image by learning global semantics from an abundant corpus of training data. An early CNN-based method for inpainting was the Context Encoder [33], where the authors proposed using an L2 loss with a global generative adversarial network (GAN) loss for improved perceptual quality. GANs [14] are especially suitable for image inpainting because they are able to synthesize realistic images [6, 18, 35, 52]. To consider local details as well as global semantics, Demir *et al.* [9] proposed using a PatchGAN [17] along with the global GAN. Our inpainting framework also employs a PatchGAN discriminator for enhanced local details.

When generating each missing pixel, CNNs with stacked convolution layers are limited by the local receptive field of the convolution operation, whereas previous patch-based methods are able to copy from any part in the surrounding known regions. Thus, Yu *et al.* [49] devised a contextual attention (CA) module that copies patches from the surrounding regions into the missing region, weighted by the computed similarity. We add the CA module in the bottleneck of our refinement network to get the best of both worlds – ability of GANs in synthesizing novel structures and details, as well as the ability to copy patches anywhere from the image without restrictions in the receptive field, like in [49, 50].

Typically, input images for inpainting consist of some input pixels that are valid, or known, while others are invalid, or unknown/missing. Liu *et al.* [25] addressed this dichotomy using partial convolution, where only the valid pixels are taken into consideration during convolution by using a predefined mask. Yu *et al.* [50] proposed gated convolutions, where the masks are also learned, while Xie *et al.* [46] proposed using learnable bidirectional attention maps. We employ gated convolutions [50] in our framework to handle the valid/invalid pixels.

CNN-based inpainting methods commonly use a two-stage approach, where the first network generates a coarse output and the second network refines this output [48, 49, 50]. Some methods [24, 31, 47] divide the stages as (i) edge completion and (ii) image completion, or in StructureFlow

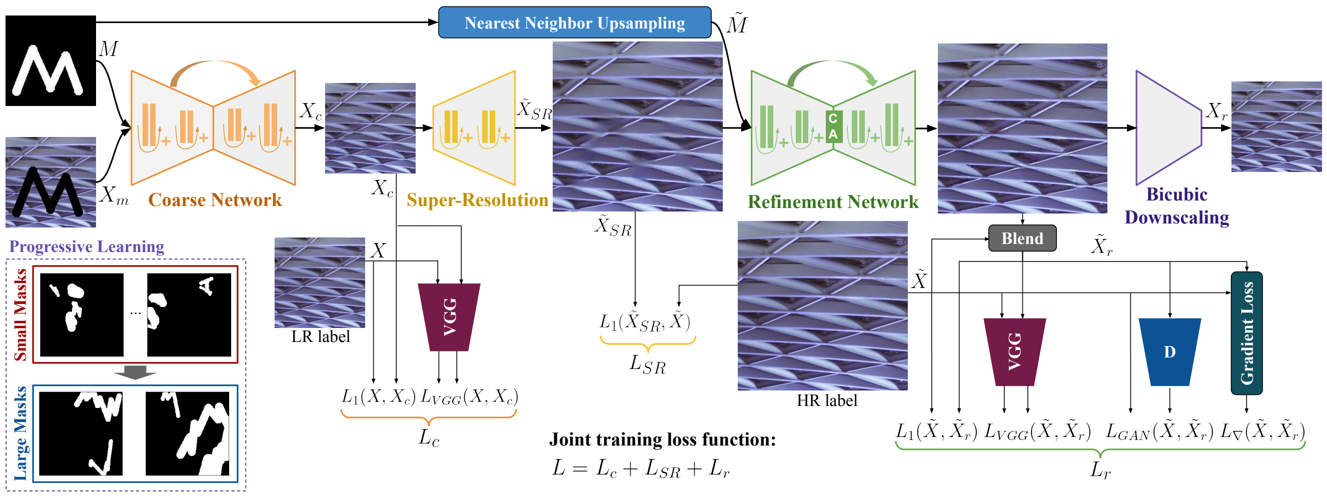


Figure 2: Our proposed inpainting framework containing three main components: a coarse network, SR network, and HR refinement network. The framework progressively learns to inpaint missing regions, from smaller masks to larger masks, and the refinement network is trained with a gradient loss to enhance the generation of high frequency details.

[37], as (i) structure generation and (ii) texture generation. In our *zoom-to-inpaint* model, we increase the resolution of the image between the two stages such that the coarse output is refined at HR for enhanced high frequency details. Upsampling can be achieved by simple bicubic interpolation, and we show that using an SR network, trained end-to-end with the coarse and refinement inpainting networks, produces even better results. Recently, inpainting methods that can perform well on HR input images were proposed [48, 51]. Due to computational complexity, the HR input is downscaled to a fixed resolution for inpainting, and then increased back to HR guided by the original HR image, using residual aggregation [48] or guided upsampling [51]. The main difference between these models and our *zoom-to-inpaint* model is that we go *beyond* the input resolution for better refinement of high-frequency details and then come back to the original or the desired resolution.

2.2. Progressive Learning

Learning to solve a difficult task from scratch can be challenging for neural networks. Hence, fine-tuning [15] or transfer learning is a commonly applied technique, where prior knowledge is transferred from pretrained networks to the subsequent training stages. Progressive neural networks [39] expanded on this idea and added lateral connections to previously learned features. Karras *et al.* [18] proposed to progressively train a GAN to synthesize low resolution (LR) images first, then to generate HR images by incrementally adding on layers to stabilize and speed up the training process. For image inpainting, filling in missing regions can be increasingly difficult as they become larger. We propose to incrementally increase the size of the regions to be inpainted as training progresses.

2.3. Gradient Loss

Utilizing the image gradients as a prior [42] or in the loss function [12, 23, 27] has been widely used for image SR [27, 42] and depth estimation [12, 23] to increase sharpness in the reconstructed images. In image inpainting, a traditional inpainting method [43] used the gradients of neighboring pixels to estimate the missing values in the inpainted region. Liu *et al.* [26] proposed a loss function to enforce the continuity of gradients in the reconstructed region and its neighboring regions. In our method, we propose to minimize the image gradients of the difference between the prediction and the ground truth, similar to [12] used for depth estimation, but explore the idea for image inpainting to further enhance the high frequency details in the result.

3. Proposed Method

We propose an inpainting method that enhances high-frequency details in the final output by (i) upsampling the result of a coarse inpainting network using an SR network and refining at HR, and (ii) employing a gradient loss. For better convergence, the framework is trained *progressively*, first on smaller masks then on larger masks. We give an overview of the proposed framework in Section 3.1 and then elaborate on progressive learning for inpainting in Section 3.2.

3.1. Framework Overview

Our inpainting framework consists of three trainable networks connected sequentially: a coarse inpainting network, an SR network, and an HR refinement network. The SR network and the HR refinement network are trained with HR (original) labels, $\tilde{X}$, whereas the coarse network is trained with the bicubic-downscaled versions, X. Each network is first pre-trained separately, and then all networks are trained jointly. Please refer to the Appendix for the detailed architectures of each network.

3.1.1 Coarse Network

The coarse network, f_c, aims to characterize the LR variations in the image across its entire field-of-view, coarsely filling in the missing regions in the input masked image, X_m, given by,

$$X_m = (1 - M) \odot X, \tag{1}$$

where $M \in \mathbb{R}^{H \times W \times 3}$ is a binary mask where invalid pixels, i.e., pixels to be inpainted, are 1 and valid pixels are 0, with repeated values across the channel dimension, $X \in \mathbb{R}^{H \times W \times 3}$ is the full image, and $\odot$ is element-wise multiplication. We employ an encoder-decoder-based CNN architecture with gated convolutions similar to [50], additionally with residual blocks and batch normalization. The network output is masked with the input, yielding X_c as,

$$X_c = M \odot f_c(X_m, M, \Theta_c) + X_m, \tag{2}$$

so that the network does not attempt to reconstruct the valid regions. We train it by minimizing the loss function L_c, consisting of an L1 loss, to enforce pixel-wise similarity, and the VGG loss, to enforce similarity in the feature domain, given as,

$$L_c = \|X_c - X\|_1 + \lambda_c^{\phi} \cdot \|\phi_{1,4}(X_c) - \phi_{1,4}(X)\|_1, \tag{3}$$

where $\phi_{i,j}$ is the i-th convolution layer at the j-th block in VGG19 [41], and λ_c^{ϕ} is a constant.

3.1.2 Super-Resolution Network

We use an SR network that *zooms in* the coarse output X_c by scale factor $s > 1$. The SR network should be simple yet effective, to reasonably upscale X_c to $\tilde{X}_{SR} \in \mathbb{R}^{sH \times sW \times 3}$. Hence, our SR network architecture is designed as a cascade of four residual blocks with a pixel shuffle layer [40] at the end. The detailed architecture is provided in the Appendix. Contrary to the coarse network output, we do not mask $\tilde{X}_{SR}$ since the refinement network in the following stage can propagate high-resolution patches from valid regions to the inpainted region. Therefore, we train the SR network by directly minimizing $L_{SR} = \|\tilde{X}_{SR} - \tilde{X}\|_1$, where $\tilde{X} \in \mathbb{R}^{sH \times sW \times 3}$ is the full HR image.

3.1.3 High Resolution Refinement Network

Different from common refinement schemes in previous inpainting frameworks, our proposed refinement is achieved by *zooming in*, refining, then *zooming out* back to the input resolution, in order to benefit from the supervision of HR labels during refinement and aid the learning of high frequency components. As long as the refinement network is trained on HR labels so that local irregularities generated by the coarse network can be corrected with the magnification, the *zoom* can even be achieved by bicubic upsampling. Using this strategy of forcing the refinement network to learn from HR labels, we push the CNN to explicitly learn the high frequency details.

Specifically, given $\tilde{X}_{SR}$ and $\tilde{M}$ as input, where $\tilde{M} \in \mathbb{R}^{sH \times sW \times 3}$ is M upscaled by nearest neighbor upsampling, the HR refinement network, f_r, generates the refined image $f_r(\tilde{X}_{SR}, \tilde{M}, \Theta_r) \in \mathbb{R}^{sH \times sW \times 3}$. Then, $\tilde{X}_r$, which is used for optimizing the training losses in the refinement network, is obtained by blending the network output with the label:

$$\tilde{X}_r = \tilde{M} \odot f_r(\tilde{X}_{SR}, \tilde{M}, \Theta_r) + (1 - \tilde{M}) \odot \tilde{X}. \tag{4}$$

By masking with the original label and not the input, $\tilde{X}_{SR}$, no loss occurs outside the inpainted regions. Because the loss is zero in the valid regions, the network does not spend its capacity on reconstructing these regions, which will be replaced by the input image, X_m, after downscaling, in the final output.

The architecture of the refinement network is similar to the coarse network and is encoder-decoder-based, except that we add a contextual attention (CA) module [49] to its bottleneck. For training, we minimize a loss function, L_r, that consists of an L1 loss, VGG loss, hinge GAN loss, L_h, and gradient loss, L_∇, given by,

$$\begin{aligned} L_r = \|\tilde{X}_r - \tilde{X}\|_1 &+ \lambda_r^{\phi} \cdot \|\phi_{i,j}(\tilde{X}_r) - \phi_{i,j}(\tilde{X})\|_1 \\ &+ \lambda_r^{GAN} \cdot L_h(\tilde{X}_r) + \lambda_r^{\nabla} \cdot L_\nabla. \end{aligned} \tag{5}$$

L1 loss and VGG loss defined in Equation 3 are used to regularize the output in the pixel domain as well as in the feature domain. A PatchGAN [17] approach is adopted for good perceptual results, with spectral normalization [29] similar to [50] for stable training of GANs. We additionally propose to use a gradient loss, L_∇, to further encourage the generation of high frequency details. Inspired by [12], L_∇ is given as,

$$L_\nabla = \frac{1}{2}(\|(\tilde{X}_r - \tilde{X})_{\nabla_x}\|_2^2 + \|(\tilde{X}_r - \tilde{X})_{\nabla_y}\|_2^2), \tag{6}$$

where ∇_x and ∇_y are horizontal and vertical image gradients, respectively, obtained by 1-tap filters, $[-1, 1]$ and $[-1, 1]^{\mathrm{T}}$.

Mask	Number of vertices	Length of piece-wise stroke	Thickness of stroke	Angle a for each vertex	Every other angle for each vertex	Invalid pixel ratio (mean / max from 10K random masks)
Small	$\mathcal{U}\{1,12\}$	$\mathcal{U}\{1,d/12\}$	$\mathcal{U}\{5,30\}$	$\mathcal{U}\{0,2\pi\}$	$a+\mathcal{U}\{3\pi/4,5\pi/4\}$	4.3% / 15.75%
Large	$\mathcal{U}\{4,12\}$	$\mathcal{N}\{d/8,d/16\}$	$\mathcal{U}\{12,40\}$	$2\pi/5+\mathcal{U}\{-2\pi/15,2\pi/15\}$	$2\pi-a$	15.12% / 50.24%

Table 1: Random distributions of mask parameters for small and large masks. d denotes the diagonal length of the image.

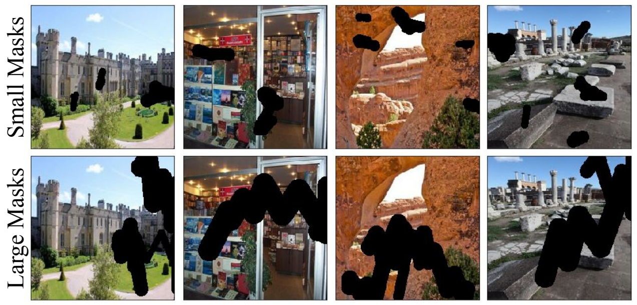


Figure 3: Example masks used during progressive learning.

The refined HR output, $f_r(\tilde{X}_{SR}, \tilde{M}, \Theta_r)$ is downscaled back (*zoomed out*), by scale factor s, to the original resolution by bicubic downsampling after the refinement, and blended with the input, X_m. The final output $X_r \in \mathbb{R}^{H\times W\times 3}$ is then given as,

$$X_r = M \odot f_r(\tilde{X}_{SR}, \tilde{M}, \Theta_r)\downarrow_s + (1-M)\odot X_m, \quad (7)$$

We train the entire framework jointly with the total loss $L = L_c + L_{SR} + L_r$. Please refer to Section 4.1 for details on hyperparameters.

3.2. Progressive Learning for Image Inpainting

In image inpainting, the training time until convergence tends to increase proportionally as invalid regions in masks become larger and increasingly difficult to fill in [25]. This is problematic if we want to refine at HR with the image enlarged by scale factor s, since the number of missing pixels would increase by s^2. Thus, we propose a progressive training strategy for inpainting, where we train the network in N steps, from smaller masks to larger masks. We set $N = 2$ in our experiments and train first with smaller random masks shown in the top row of Figure 3, then with larger random masks used in [49] shown in the bottom row of Figure 3. For generating the small masks, we modify the random mask generation algorithm in [49] as summarized in Table 1. This progressive learning strategy helps the framework to converge faster with the large masks.

4. Experiment Results and Evaluation

In this section, we evaluate our results, compare them to other inpainting methods, and demonstrate the effectiveness of the components in our framework via ablation studies.

4.1. Implementation Details

Training configuration. For training our *zoom-to-inpaint* model, we first pretrain the coarse and refinement networks on Places2 [54] with 256 × 256 images for 80K iterations using small masks. For both networks, we only use the L1 loss and the VGG loss, with $\lambda^{\phi} = 0.01$. For SR network, we use an upsampling scale $s = 2$ and pretrain the network on DIV2K [2] for 400K iterations with randomly cropped 64×64 patches. Then, we jointly train the entire framework on high resolution DIV2K [2] using the proposed progressive learning strategy, i.e., 80K iterations with small masks, and then another 1.2M iterations with large masks. We randomly crop 512×512 patches from DIV2K and use them as HR labels ($\tilde{X}$), and bicubic-downsample them to 256×256 to generate LR labels (X). We use the following loss coefficients: $\lambda_c^{\phi} = 0.01$, $\lambda_r^{\phi} = 10^{-5}$, $\lambda_r^{GAN} = 0.5$ and $\lambda_r^{\nabla} = 1$. Our implementation is in Tensorflow [1] and trained on 8 NVIDIA V100 GPUs using Adam optimizer [21] with a mini-batch size of 16 and a learning rate of 10^{-5}. Please refer to the Appendix for the details of our model architecture and a complexity analysis.

Test dataset. For the test dataset, we applied the small and large random masks [50] as shown in Figure 3 on 200 images from the validation and test sets of Places2 (100 images each), and on 100 images in the validation set of DIV2K. For DIV2K, 256×256 patches were center-cropped from the full images. Although large masks [50] were used for benchmarking in previous papers, we have observed that the visual reconstruction quality of our method changes depending on the size of the inpainted region and therefore, we evaluate on both small and large masks.

Inpainting methods. We compare our method with recent state-of-the-art inpainting approaches: DeepFillv2 [50], EdgeConnect [31], Pluralistic [53], and HiFill [48]. Similar to our method, [31, 50, 53] were trained on 256 × 256 images and therefore our test set can be used as is. However, since HiFill [48] was originally trained on 512 × 512 images, we bicubic-upscale the test images to 512×512 for input and then downscale the output back to 256 × 256 before computing the metrics. We used the publicly released weights trained on Places2 for all methods.

Method	Places2 (256 × 256)				DIV2K (256 × 256, center-cropped)				User Study
	PSNR ↑	SSIM ↑	MS-SSIM ↑	L1 Error ↓	PSNR ↑	SSIM ↑	MS-SSIM ↑	L1 Error ↓	Prefer Ours
Small masks									
HiFill [48]	31.12	0.9586	0.9742	0.00744	30.91	0.9633	0.9777	0.00700	75.49%
Pluralistic [53]	33.23	0.9670	0.9807	0.00558	32.73	0.9703	0.9820	0.00543	89.13%
DeepFill-v2 [50]	34.03	0.9719	0.9834	0.00485	33.11	0.9741	0.9852	0.00467	69.23%
EdgeConnect [31]	33.98	0.9718	0.9841	0.00388	33.11	0.9734	0.9845	0.00388	64.21%
Ours	**34.78**	**0.9755**	**0.9863**	**0.00357**	**34.08**	**0.9787**	**0.9886**	**0.00329**	-
Large masks									
HiFill [48]	24.94	0.8891	0.9134	0.02034	24.23	0.8739	0.8993	0.02302	63.38%
Pluralistic [53]	26.17	0.9022	0.9191	0.01784	25.62	0.8890	0.9071	0.01949	71.38%
DeepFill-v2 [50]	26.77	0.9158	0.9326	0.01536	26.07	0.9018	0.9229	0.01735	30.67%
EdgeConnect [31]	27.61	0.9166	0.9382	0.01328	26.87	0.9036	0.9291	0.01494	34.40%
Ours	**27.71**	**0.9202**	**0.9415**	**0.01314**	**27.07**	**0.9094**	**0.9346**	**0.01462**	-

Table 2: Quantitative comparison and user study results on small and large inpainted regions. Values in **bold** denote the best performance, and the results of the user study is indicated by the percentage (%) of users who selected our method over the compared method. More details of the user study including the number of counts is provided in the Appendix.

4.2. Quantitative Evaluation

For quantitative evaluation, we measure the PSNR, SSIM [44], MS-SSIM (multi-scale SSIM) [45] and L1 error of the output inpainted images. As shown in Table 2, our *zoom-to-inpaint* model outperforms all compared methods, on all metrics. We test on both small and large masks and our improvement is larger on the small masks. Specifically, on the Places2 dataset, while we observe an improvement of 0.1 dB in PSNR over the next best method for the large masks, we improve by 0.75 dB for the small masks. Similarly, for the DIV2K dataset, the improvement on the large masks is 0.2 dB compared to 0.97 dB on the small masks. We therefore conclude that the performance gain of our framework is more significant when the inpainted regions are small. This observation is also supported by the results of the user study described in the next section.

Ablations	PSNR ↑	SSIM ↑	MS-SSIM ↑	L1 Error ↓
Small masks				
No zoom	32.12	0.9714	0.9812	0.00441
Bicubic zoom	32.80	0.9753	0.9832	0.00391
SR zoom	33.40	0.9770	0.9853	0.00363
SR zoom+$L_\triangledown$	**34.08**	**0.9787**	**0.9886**	**0.00329**
Large masks				
No zoom	25.89	0.8977	0.9180	0.01747
Bicubic zoom	26.09	0.9016	0.9219	0.01657
SR zoom	26.95	0.9080	0.9307	0.01497
SR zoom+$L_\triangledown$	**27.07**	**0.9094**	**0.9346**	**0.01462**

Table 3: Quantitative comparison of our model (SR zoom+$L_\triangledown$) with its ablations.

4.3. Qualitative Evaluation

Qualitative comparison. We show qualitative comparisons in Figures 1 and 4. In Figure 1, our method accurately reconstructs edges and fine lines in the correct orientation while other methods find it difficult to preserve the continuity of the fine lines or fail to produce any edges at all in the missing region. Similarly, in Figure 4, our method accurately reconstructs high frequency details such as edges and fine texture. Please refer to the Appendix for additional results, including full images of the crops shown in Figure 4.

User study. We also conduct a user study to evaluate the preferences of users on the results produced by our approach compared to the other methods. We use the same test sets as in the quantitative evaluation, with the small and large masks datasets shown to 13 and 15 users respectively. Each user evaluates 300 pairs of inpainted images in a random order, where one image is generated by our method, and the other image is generated by a method among [31, 48, 50, 53]. Users are asked to select their preferred result based on the question: "Which of these images looks better?".

The percentage of users that prefer our method over the others is summarized in the last column of Table 2. Users more frequently prefer our method over all other methods for the small masks. Interestingly, for the larger masks, the users prefer [50] and [31] over our method even though our method has better quantitative metrics. We attribute this observation to the fact that examples with large masks contain sizeable missing regions and all methods generate severe artifacts, and users select the result with a more preferable artifact rather than a more realistic result. While our framework can generate realistic high frequency details for most

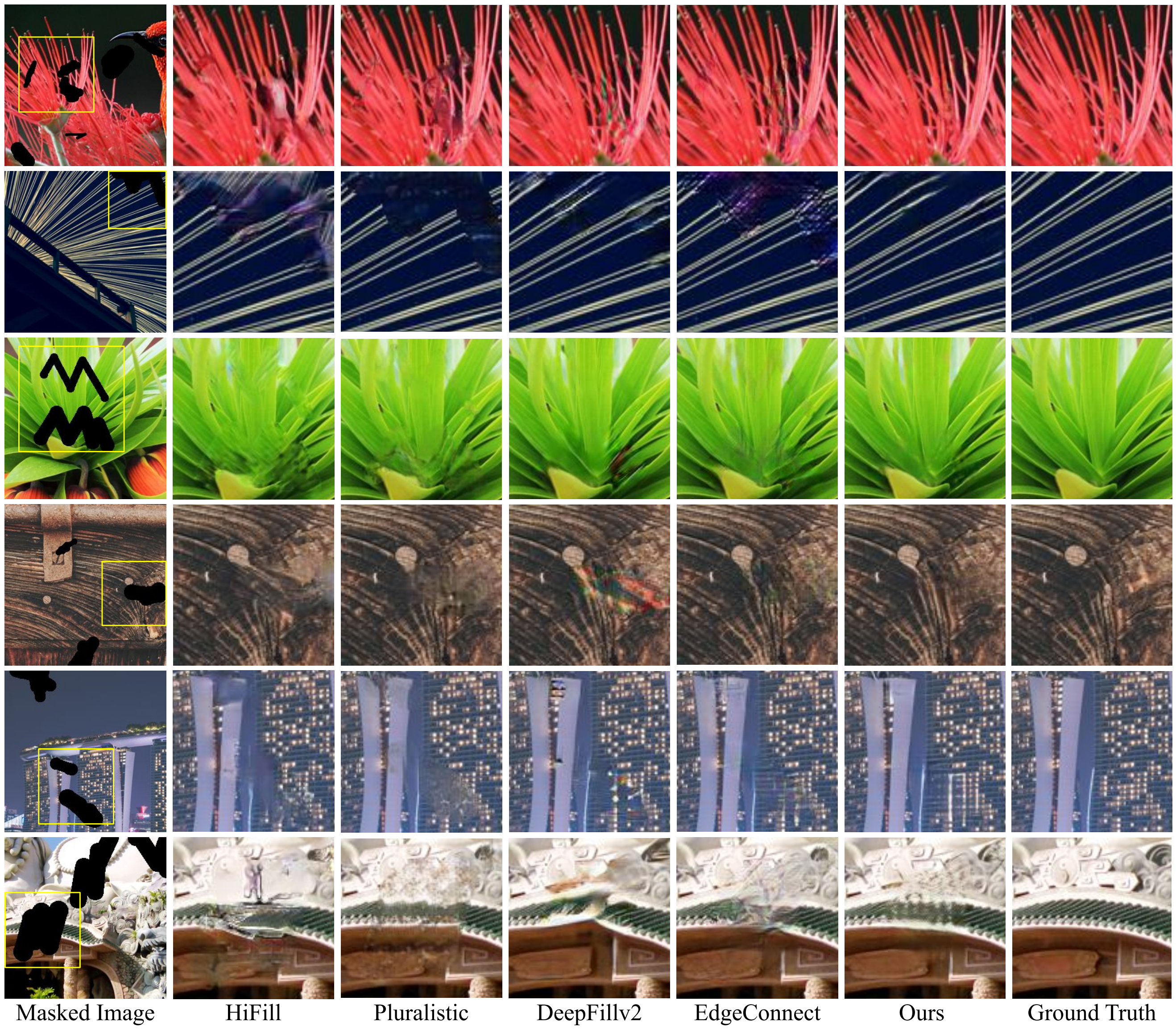


Figure 4: Qualitative comparison to other methods. When images with high frequency regions are fed into inpainting networks, HiFill [48] and Pluralistic [53] tend to generate blurry texture as seen in the examples in the 1st, 4th and 5th rows, and DeepFillv2 [50] and EdgeConnect [31] generate color artifacts as shown in the 3rd, 4th and 5th rows. Our method is able to accurately reconstruct high frequency details.

cases, when provided with a challenging input where all methods fail, it can produce an artifact with a higher frequency compared to the artifacts generated by [50] and [31]. In some examples, the artifacts contain a high frequency repetitive pattern that is displeasing to the users and we believe that these patterns cause users to prefer the other methods, which may have less objectionable artifacts. Examples of these images as well as a screenshot of the user study are provided in the Appendix.

4.4. Ablation Study

Ablation study on framework components. In order to analyze the contributions of the components in our inpainting framework, we compare against three ablations of our framework: (i) No zoom, (ii) Bicubic zoom and (iii) SR zoom. For (i), we replace the SR component (described in Section 3.1.2) with an identity transform that simply copies the coarse output without an upsampling component, so that the refinement is applied at the original resolution. For (ii), we replace the SR component with bicubic upsampling.

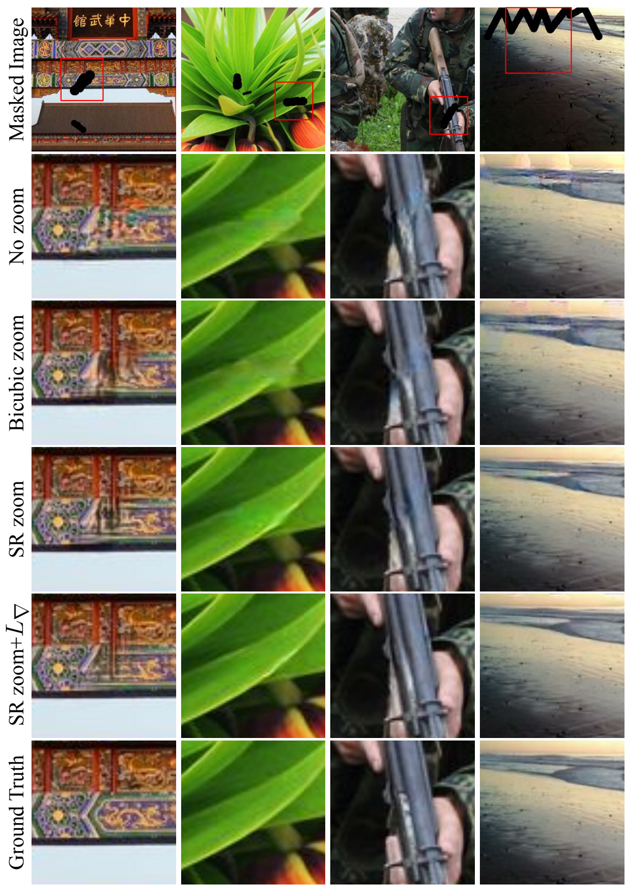


Figure 5: Visual comparison of results produced by our model and its ablations. HR refinement and the gradient loss improves the generation of high frequency details.

Both (i) and (ii) are trained without the gradient loss L_∇. For (iii), we add back the SR zoom but not the gradient loss. The results are summarized in Table 3 where SR Zoom+L_∇ corresponds to our full framework.

As shown in Table 3, zooming in with bicubic upsampling improves all metrics compared to doing refinement at the original resolution (*No zoom*) showing the benefit of refining at a higher resolution and training the refinement network with HR supervision. Adding the SR network further improves the accuracy by a large margin. Compared to bicubic zoom, the SR network generates sharper areas for the surrounding regions as well, that can then be propagated by the CA module to the inpainted region. Adding the gradient loss further improves the quantitative metrics for both small and large masks. Its benefit is more prominent for small masks, where the network is more likely to reconstruct the image more accurately, and thus, fine details contribute more to the evaluation metrics. Qualitative results are shown in Figure 5. Please refer to the Appendix for additional results on the ablation study.

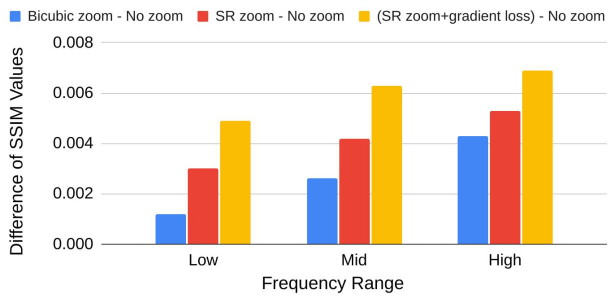


Figure 6: Frequency domain comparison of the ablation models using 2-level Laplacian pyramids [7], evaluated on small masks. Each component of our framework adds to improving the reconstruction of the inpainted regions, with high frequencies benefitting more than lower frequencies.

Frequency-domain analysis. We provide insights into the benefits of zooming in and refining at high resolution even though the final output is of lower resolution. Using frequency-domain analysis, we demonstrate that our strategy introduces desirable frequencies into the inpainted result that survive downsampling. Specifically, instead of directly computing metrics corresponding to a ground truth image and a prediction, we first construct two-level Laplacian pyramids [7] for both by using a traditional 5-tap Gaussian kernel, and report per-level metrics. This allows us to measure the accuracy in different frequency bands.

In Figure 6, we measure the per-level improvement of *Bicubic zoom*, *SR zoom* and *SR zoom*+L_∇ over the baseline *No zoom*. We use the SSIM metric, that is known to be sensitive to local structural changes, and use the small masks dataset to avoid the effect of artifacts generated with very large masks, as mentioned in Section 4.3.

While we see improvements in all frequency bands, we observe that the improvements are skewed towards higher frequency bands. This indicates that all components of our framework, i.e., refining at high resolution with HR labels, SR zoom and gradient loss improve the overall reconstruction, more so at higher frequencies. Please see the Appendix for additional results.

5. Conclusion

We propose a novel inpainting framework with HR refinement by inserting an SR network between coarse and refinement networks so that the refinement network refines at HR. By training the refinement network with HR labels, our model learns from high-frequency components present in the HR labels, reducing the spectral bias [36] at the desired resolution. Furthermore, with the addition of the gradient loss for inpainting, our *zoom-to-inpaint* model generates even more accurate texture and details. To improve training, we use a progressive learning strategy for inpainting that increases the area of the missing regions as the training progresses. The HR refinement, progressive learning and gradient loss can each or together be applied to any inpainting framework. Our method surpasses current inpainting methods quantitatively. Qualitatively, based on a user study, we find that there's a dependence on the size of the masks. Our approach outperforms all other methods for small masks but for large masks, where all methods tend to have more prominent artifacts, we trail behind two of the four evaluated methods, perhaps because our method tends to produce high frequency artifacts that are more objectionable. Our code will be made publicly available.

Acknowledgement We thank Jonathan T. Barron for insightful discussions and Yael Pritch and David Salesin for useful comments.

References

[1] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org. 5

[2] Eirikur Agustsson and Radu Timofte. Ntire 2017 challenge on single image super-resolution: Dataset and study. In *IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2017. 5

[3] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Transactions on Graphics*, 2009. 1, 2

[4] Connelly Barnes, Eli Shechtman, Dan B. Goldman, and Adam Finkelstein. The generalized patchmatch correspondence algorithm. In *European Conference on Computer Vision*, 2010. 2

[5] Marcelo Bertalmio, Guillermo Sapiro, Vincent Caselles, and Coloma Ballester. Image inpainting. In *SIGGRAPH*, 2000. 2

[6] Andrew Brock, Jeff Donahue, and Karen Simonyan. Large scale gan training for high fidelity natural image synthesis. In *International Conference on Learning Representations*, 2019. 2

[7] Peter Burt and Edward Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540, 1983. 8, 16

[8] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential linear units (elus). In *International Conference on Learning Representations*, 2016. 11

[9] Ugur Demir and Gozde Unal. Patch-based image inpainting with generative adversarial networks. In *arXiv:1803.07422*, 2018. 2

[10] Alexei A. Efros and William T. Freeman. Image quilting for texture synthesis and transfer. In *SIGGRAPH*, 2001. 2

[11] Alexei A. Efros and Thomas K. Leung. Texture synthesis by non-parametric sampling. In *IEEE International Conference on Computer Vision*, 1999. 2

[12] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *IEEE International Conference on Computer Vision*, 2015. 2, 3, 4

[13] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2011. 11

[14] Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. In *International Conference on Neural Information Processing Systems*, 2014. 2

[15] Geoffrey E. Hinton and Ruslan R. Salakhutdinov. Reducing the dimensionality of data with neural networks. *Science*, 2006. 3

[16] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 2015. 11

[17] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A. Efros. Image-to-image translation with conditional adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2, 4, 11

[18] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *International Conference on Learning Representations*, 2018. 2, 3

[19] Jiwon Kim, Jung Kwon Lee, and Kyoung Mu Lee. Accurate image super-resolution using very deep convolutional networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 11

[20] Soo Ye Kim, Jihyong Oh, and Munchurl Kim. Fisr: Deep joint frame interpolation and super-resolution with a multi-scale temporal loss. In *AAAI Conference on Artificial Intelligence*, 2020. 2

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

[22] Nikos Komodakis and Georgios Tziritas. Image completion using efficient belief propagation via priority scheduling and dynamic pruning. *IEEE Transactions on Image Processing*, 16(11):2649–2661, 2007. 2

[23] Zhengqi Li and Noah Snavely. Megadepth: Learning single-view depth prediction from internet photos. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 3

[24] Liang Liao, Ruimin Hu, Jing Xiao, and Zhongyuan Wang. Edge-aware context encoder for image inpainting. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, 2018. 2

[25] Guilin Liu, Fitsum A. Reda, Kevin J. Shih, Ting-Chun Wang, Andrew Tao, and Bryan Catanzaro. Image inpainting for irregular holes using partial convolutions. In *European Conference on Computer Vision*, 2018. 2, 5, 14

[26] Huaming Liu, Guanming Lu, Xuehui Bi, Jingjie Yan, and Weilan Wang. Image inpainting based on generative adversarial networks. In *IEEE International Conference on Natural Computation, Fuzzy Systems and Knowledge Discovery*, 2018. 3

[27] Cheng Ma, Yongming Rao, Yean Cheng, Ce Chen, Jiwen Lu, and Jie Zhou. Structure-preserving super resolution with gradient guidance. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 3

[28] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning*, 2013. 12

[29] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learining Representations*, 2018. 4, 12

[30] Seungjun Nah, Tae Hyun Kim, and Kyoung Mu Lee. Deep multi-scale convolutional neural network for dynamic scene deblurring. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2017. 2

[31] Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Qureshi, and Mehran Ebrahimi. Edgeconnect: Structure guided image inpainting using edge prediction. In *IEEE International Conference on Computer Vision Workshops*, 2019. 2, 5, 6, 7, 12, 13, 14, 15, 16, 17

[32] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. 11

[33] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A. Efros. Context encoders: Feature learning by inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 2

[34] Yael Pritch, Eitam Kav-Venaki, and Shmuel Peleg. Shift-map image editing. In *IEEE International Conference on Computer Vision*, 2009. 2

[35] Alec Radford, Luke Metz, and Soumith Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *International Conference on Learning Representations*, 2016. 2

[36] Nasim Rahaman, Aristide Baratin, Devansh Arpit, Felix Draxler, Min Lin, Fred A. Hamprecht, Yoshua Bengio, and Aaron Courville. On the spectral bias of neural networks. In *International Conference on Machine Learning*, 2019. 1, 8

[37] Yurui Ren, Xiaoming Yu, Ruonan Zhang, Thomas H. Li, Shan Liu, and Ge Li. Structureflow: Image inpainting via structure-aware appearance flow. In *IEEE International Conference on Computer Vision*, 2019. 3

[38] 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

[39] Andrei A. Rusu, Neil C. Rabinowitz, Guillaume Desjardins, Hubert Soyer, James Kirkpatrick, Koray Kavukcuoglu, Razvan Pascanu, and Raia Hadsell. Progressive neural networks. In *arXiv:1606.04671*, 2016. 3

[40] Wenzhe Shi, Jose Caballero, Ferenc Huszár, Johannes Totz, Andrew P. Aitken, Rob Bishop, Daniel Rueckert, and Zehan Wang. Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2016. 4, 11

[41] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015. 4

[42] Jian Sun, Zongben Xu, and Heung-Yeung Shum. Image super-resolution using gradient profile prior. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2008. 3

[43] Alexandru Telea. An image inpainting technique based on the fast marching method. *Journal of Graphics Tools*, 2004. 2, 3

[44] Zhou Wang, Alan C. Bovik, Hamid R. Sheikh, and Eero P. Simoncelli. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing*, 13(4):600–612, 2004. 6

[45] Zhou Wang, Eero P. Simoncelli, and Alan C. Bovik. Multiscale structural similarity for image quality assessment. *Asilomar Conference on Signals, Systems & Computers*, 2:1398–1402, 2003. 6

[46] Chaohao Xie, Shaohui Liu, Chao Li, Ming-Ming Cheng, Wangmeng Zuo, Xiao Liu, Shilei Wen, and Errui Ding. Image inpainting with learnable bidirectional attention maps. In *IEEE International Conference on Computer Vision*, 2019. 2

[47] Wei Xiong, Jiahui Yu, Zhe Lin, Jimei Yang, Xin Lu, Connelly Barnes, and Jiebo Luo. Foreground-aware image inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 2

[48] Zili Yi, Qiang Tang, Shekoofeh Azizi, Daesik Jang, and Zhan Xu. Contextual residual aggregation for ultra high-resolution image inpainting. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2020. 1, 2, 3, 5, 6, 7, 12, 13, 14, 15, 16, 17

[49] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2018. 2, 4, 5, 11

[50] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *IEEE International Conference on Computer Vision*, 2019. 2, 4, 5, 6, 7, 11, 12, 13, 14, 15, 16, 17

[51] Yu Zeng, Zhe Lin, Jimei Yang, Jianming Zhang, Eli Shechtman, and Huchuan Lu. High-resolution image inpainting with iterative confidence feedback and guided upsampling. In *European Conference on Computer Vision*, 2020. 3

[52] Han Zhang, Tao Xu, Hongsheng Li, Shaoting Zhang, Xiaogang Wang, Xiaolei Huang, and Dimitris Metaxas. Stackgan: Text to photo-realistic image synthesis with stacked genera-

tive adversarial networks. In *IEEE International Conference on Computer Vision*, 2017. 2

[53] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Pluralistic image completion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 1, 5, 6, 7, 12, 13, 14, 15, 16, 17

[54] Bolei Zhou, Agata Lapedriza, Aditya Khosla, Aude Oliva, and Antonio Torralba. Places: A 10 million image database for scene recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017. 5

Appendices

A. Detailed Network Architectures

In this section, we give details of the network architectures of the different components in our zoom-to-inpaint framework.

A.1. Coarse Network

The coarse network in our framework has an encoder-decoder-based architecture with residual connections at each level as shown in Figure 9. We find that the residual connections help to propagate the information at each level and help convergence in training. We employ gated convolutions with ELU activation [8] as in [50], and batch normalization (BatchNorm) [16] at every layer. For downscaling the feature resolution, we use max pooling with a 2×2 window, and for upscaling we use nearest neighbor upsampling by a factor of 2 with an additional convolution layer to avoid checkerboard artifacts [32]. At the bottleneck, the gated convolutions are applied with dilation rate set to 2, 4 and 8 like in [50], for an enlarged receptive field. All gated convolution filters are of size 3×3, and the number of output channels is denoted at the top of each level in Figure 9. The network output is computed by blending with the input as mentioned in Equation 2 in the main paper.

A.2. Super-Resolution Network

As shown in Figure 7, the super-resolution (SR) network in our framework has four cascaded residual blocks, each composed of two convolution layers with ReLU activation [13]. We employ pixel shuffle [40] at the end to increase the size of the effective receptive field. All output channels are 64 except for the layer before pixel shuffle, which is 256, and the last convolution layer, which is 3, for reconstructing RGB channels for the output. We add a global residual connection with bicubic upsampling as in [19] so that the network can focus on recovering the missing high frequency components rather than on low frequency components that are already present in the input. All convolution filters are of size 3×3.

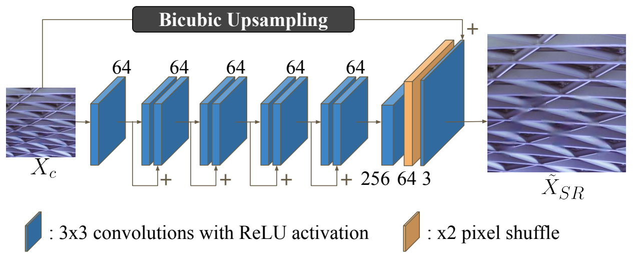


Figure 7: Architecture of the SR network in our zoom-to-inpaint framework.

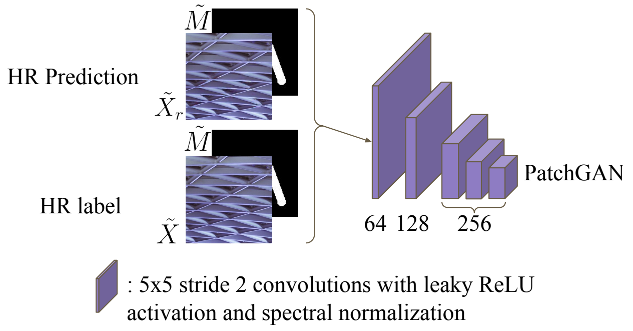


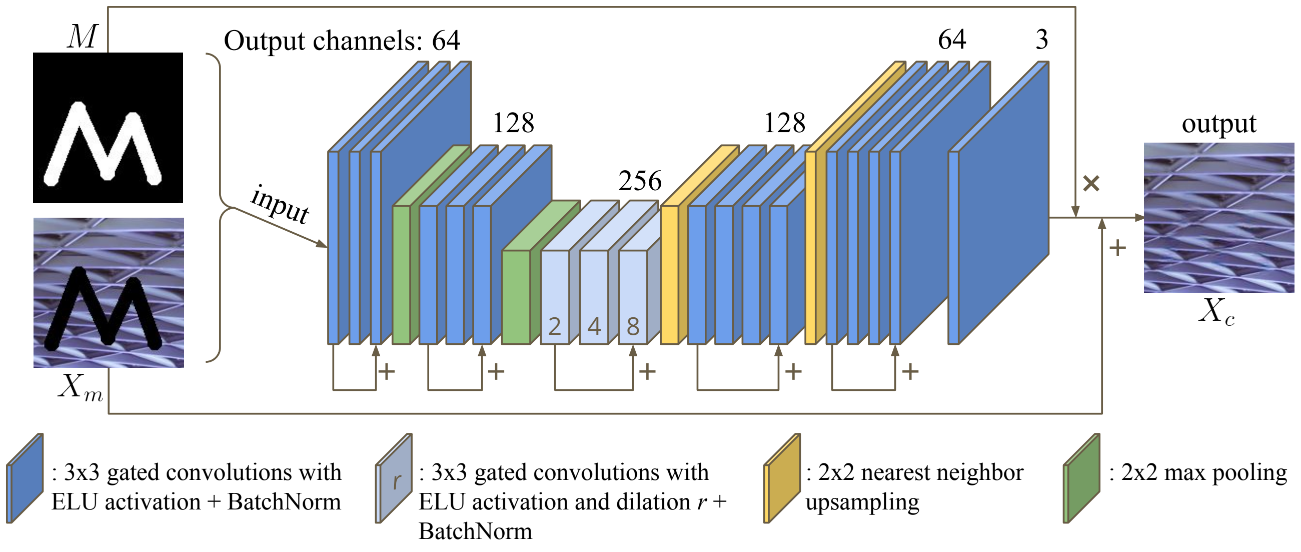
Figure 8: Architecture of the PatchGAN discriminator used during training.

A.3. Refinement Network

The architecture of our refinement network is shown in Figure 10. The refinement network is similar to the coarse network except that a contextual attention [49] module is added to the bottleneck. The contextual attention module copies patches from the surrounding known regions into the regions to be inpainted based on the computed similarity. For memory-wise efficiency, the attention map is computed on $\times 1/2$ downscaled feature maps of the bottleneck, and applied on the original resolution of the bottleneck features. The resulting feature maps after contextual attention are concatenated with the feature maps of the main pass at the bottleneck so that proceeding layers can merge both information sources as needed. In our complete zoom-to-inpaint framework, the refinement network works at high resolution (HR) of 512×512 for enhanced refinement with magnification (zoom). Note that the output of the refinement network is blended with the HR label when calculating the losses as mentioned in Equation 4 in the main paper.

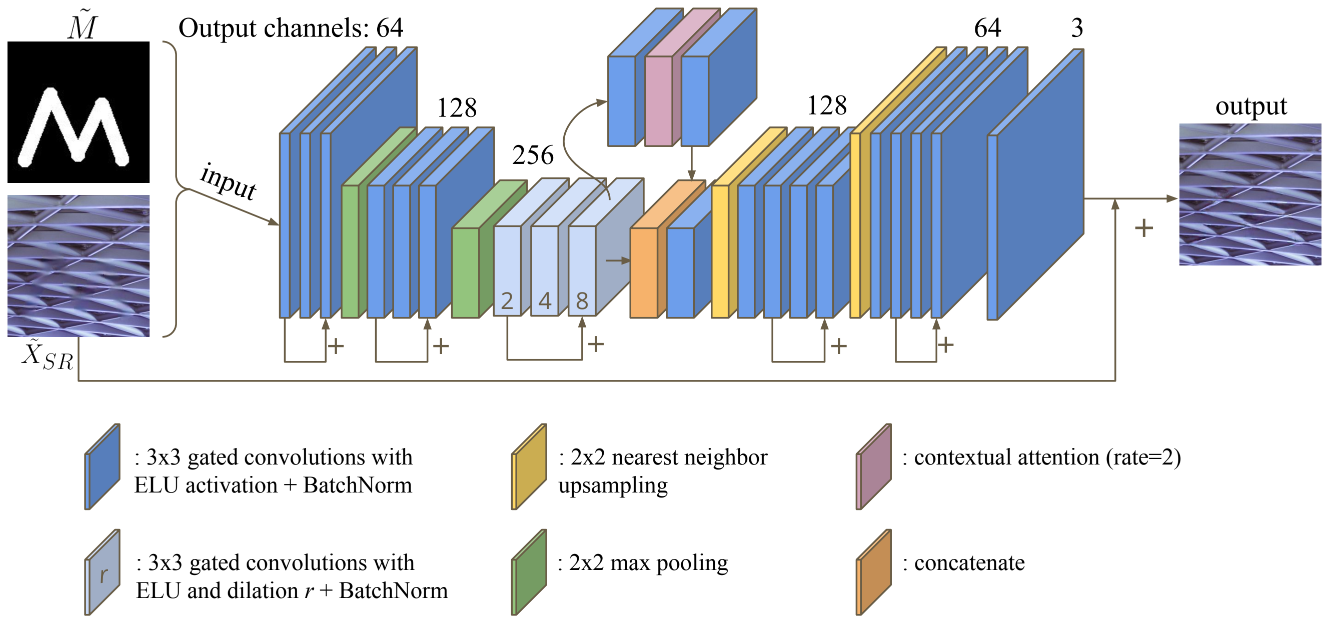
A.4. Discriminator

The discriminator architecture used for training the refinement network is shown in Figure 8. We use a hinge GAN loss with a PatchGAN [17] discriminator that aims to discriminate between the HR prediction generated by the refinement network (blended with the HR label) and the HR



M
Output channels: 64
128
256
128
64
3
input
output
X_c
X_m
2
4
8
+
×
r
: 3x3 gated convolutions with ELU activation + BatchNorm
: 3x3 gated convolutions with ELU activation and dilation r + BatchNorm
: 2x2 nearest neighbor upsampling
: 2x2 max pooling

Figure 9: Architecture of the coarse inpainting network in our zoom-to-inpaint framework.



$\tilde{M}$
Output channels: 64
128
256
128
64
3
input
output
$\tilde{X}_{SR}$
2
4
8
+
r
: 3x3 gated convolutions with ELU activation + BatchNorm
: 2x2 nearest neighbor upsampling
: contextual attention (rate=2)
: 3x3 gated convolutions with ELU and dilation r + BatchNorm
: 2x2 max pooling
: concatenate

Figure 10: Architecture of the refinement network in our zoom-to-inpaint framework.

label. 5×5 convolution filters with stride 2 and leaky ReLU [28] activation are used. Spectral normalization [29] is applied at every layer for stable training of the GAN framework.

B. Complexity Analysis

We compare the number of parameters and inference time of [31, 48, 50, 53] in Table 4. The average inference time in milliseconds is measured over 100 center crops of size 256×256 from the DIV2K validation dataset with large masks for Pluralistic [53], DeepFill-v2 [50], EdgeConnect [31] and Ours, which were all trained on 256×256 images. For HiFill [48], which was trained on 512×512, we enlarge the mask by nearest neighbor upsampling and the image by bicubic upsampling to match the resolution it was trained on, and measure the inference time, same as the way we performed the qualitative and quantitative evaluations. Image file I/O (read and write) times are excluded and we only measure the time it takes to run inference of the CNN for all

Method	HiFill [48]	Pluralistic [53]	DeepFill-v2 [50]	EdgeConnect [31]	Ours
Number of Parameters	2.7M*	6M*	4.1M*	21.53M	4.5M
Inference Time	127 ms (60 ms)	37 ms	71 ms	21 ms	293 ms

*Values copied from the publication.

Table 4: Comparison of the number of parameters and average inference time in milliseconds on 100 images. The inference time for HiFill is measured on crops enlarged to 512×512. Other methods are measured for 256×256 crops. Value in brackets denotes the time measured without pre- and post-processing for HiFill.

Ablations	ch.	Small masks				Large masks				Number of Parameters
		PSNR ↑	SSIM ↑	MS-SSIM ↑	L1 Error ↓	PSNR ↑	SSIM ↑	MS-SSIM ↑	L1 Error ↓	
No zoom	64	32.12	0.9714	0.9812	0.00441	25.89	0.8977	0.9180	0.01747	4.03M
Bicubic zoom	64	32.80	0.9753	0.9832	0.00391	26.09	0.9016	0.9219	0.01657	4.03M
No zoom	70	32.72	0.9723	0.9818	0.00432	26.00	0.8993	0.9195	0.01761	4.82M
Bicubic zoom	70	32.78	0.9730	0.9833	0.00413	25.99	0.9000	0.9212	0.01685	4.82M
SR zoom	64	33.40	0.9770	0.9853	0.00363	26.95	0.9080	0.9307	0.01497	4.48M
SR zoom+L_∇	64	**34.08**	**0.9787**	**0.9886**	**0.00329**	**27.07**	**0.9094**	**0.9346**	**0.01462**	4.48M

Table 5: Extended ablation study results including models with 70 output channels for *No zoom* and *Bicubic zoom*. The *SR zoom* models still outperform *No zoom* and *Bicubic zoom* models even with less number of trainable parameters, signifying that the benefits of *SR zoom* come from the framework design rather than network capacity. Values in **bold** denote best performance and ch. denotes the number of output channels at each convolution layer.

methods except HiFill. For HiFill, we measured the inference time both with and without pre- and post-processing, which includes their proposed residual aggregation module. For our model, we measure the time it takes to process the input image through the entire pipeline including the coarse network, SR network, refinement network and final including downscaling. Because our refinement network handles higher resolution images to generate better high frequency details, the inference time takes longer compared to other methods. Computation was on an NVIDIA Tesla T4 GPU.

C. Extended Quantitative Evaluations

C.1. Ablation Study

We provide an extended ablation study result in Table 5 on 256×256 center crops of the DIV2K validation set with both small and large masks. It includes the four ablation models introduced in the main paper: (i) No zoom, (ii) Bicubic zoom, (iii) SR zoom, and (iv) SR zoom+L_∇. For (i) No zoom, we replace the SR component with an identity transform that simply copies the coarse output without an upsampling component, so that the refinement is applied at the original resolution. For (ii) Bicubic zoom, we replace the SR component with bicubic upsampling. Both (i) and (ii) are trained without the gradient loss, L_∇. For (iii) SR zoom, we add back the SR zoom but not the gradient loss, and (iv) SR zoom+L_∇ is our final framework with gradient loss.

The *SR zoom* models, (iii) and (iv), contain an additional trainable component – the SR network. These models have 4.48M trainable parameters, compared to *No zoom* and *Bicubic zoom* with 4.03M trainable parameters at the same number of output channels. To test the effect of the capacity of the network on performance, we increase the number of output channels from 64 to 70 in the coarse network and the refinement network of the *No zoom* and *Bicubic zoom* frameworks so that they have more number of parameters (4.82M) than the *SR zoom* models. The quantitative results along with the number of parameters are provided in Table 5. As shown in Table 5, the *SR zoom* models still outperform the *No zoom* and *Bicubic zoom* models even with less number of parameters and we can conclude that the benefits of our *SR zoom* framework are not from the increased number of parameters.

C.2. Quantitative Comparison

We provide an extended table on the quantitative comparison with other inpainting methods [31, 48, 50, 53] in Table 7. For reference, in addition to the values measured on our testsets, which were also reported in the main paper, we have added the evaluation values from the original publications, measured on the Places2 dataset. Note that they were evaluated under different settings in their original papers, as summarized in Table 6, and are not directly comparable to the values we have measured on the small and large masks.

D. User study

As mentioned in the main paper, we conducted a user study, in which 13 and 15 users evaluated the inpainting

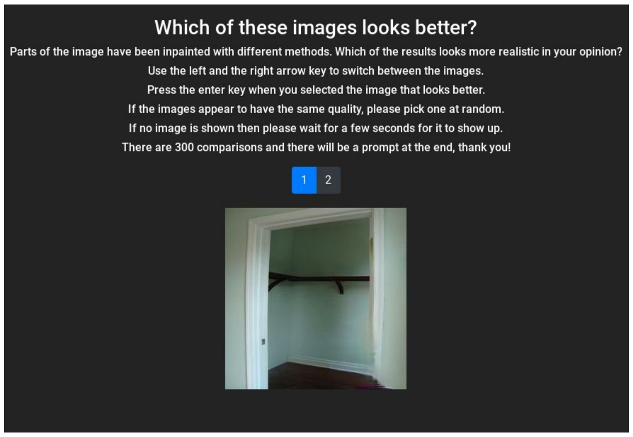
Method	Testing conditions in original publications
HiFill [48]	Tested on Places2 validation set, randomly cropped by 512 × 512. Used random masks from [25] as well as their own random object masks.
Pluralistic [53]	Only provided quantitative evaluations on ImageNet (256 × 256). Proposed own random masks with random lines, circles and ellipses.
DeepFill-v2 [50]	Tested on Places2 validation set (256 × 256). Proposed random brush stroke masks that can be generated on-the-fly during training (*same as our large masks*).
EdgeConnect [31]	Tested on Places2 (256 × 256) but no mention on which images were used. Used random masks from [25].

Table 6: Descriptions on quantitative evaluations performed in the original publications, which were used for obtaining values in gray in Table 7, copied from the original publications.

Method	Places2 (256 × 256)			
	PSNR ↑	SSIM ↑	MS-SSIM ↑	L1 Error ↓
Values copied from the publications				
HiFill [48]	-	-	0.8840	0.05439
Pluralistic [53]	-	-	-	-
DeepFill-v2 [50]	-	-	-	0.09100
EdgeConnect [31]	27.95	0.9200	-	0.01500
Our measurement, Small masks				
HiFill [48]	31.12	0.9586	0.9742	0.00744
Pluralistic [53]	33.23	0.9670	0.9807	0.00558
DeepFill-v2 [50]	34.03	0.9719	0.9834	0.00485
EdgeConnect [31]	33.98	0.9718	0.9841	0.00388
Ours	**34.78**	**0.9755**	**0.9863**	**0.00357**
Our measurements, Large masks				
HiFill [48]	24.94	0.8891	0.9134	0.02034
Pluralistic [53]	26.17	0.9022	0.9191	0.01784
DeepFill-v2 [50]	26.77	0.9158	0.9326	0.01536
EdgeConnect [31]	27.61	0.9166	0.9382	0.01328
Ours	**27.71**	**0.9202**	**0.9415**	**0.01314**

Table 7: Extended quantitative comparison on Places2. Values copied from the original publications are denoted in gray for reference. Values from the original papers are not directly comparable to our measurements as they were all tested under different settings, as summarized in Table 6. Values in **bold** denote the best performance on our test set with small and large masks.

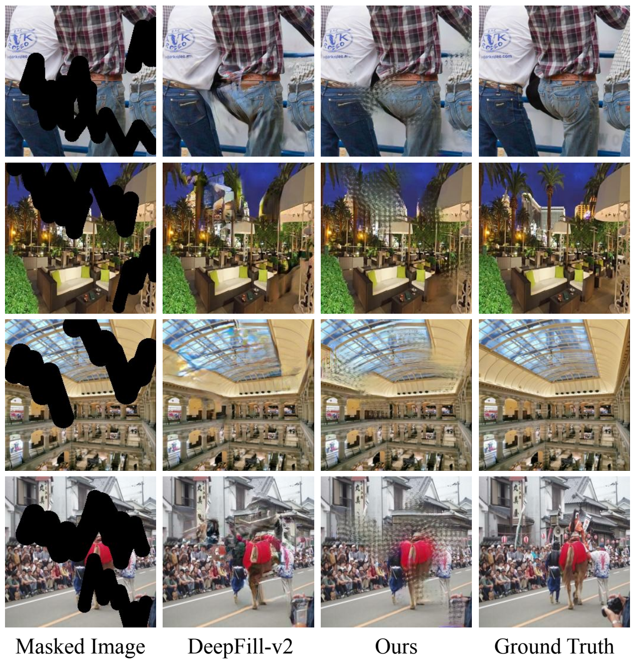
results on small and large masks, respectively. Each user compared 300 image pairs, where one is generated by our method and the other is generated by one of the inpainting methods in [31, 48, 50, 53]. The appearing order of methods (whether ours comes first, or the compared method comes first) and the appearing order of image pairs was randomized for each user. A screenshot of the user study is shown in Figure 11. Participants could toggle between the two compared images and select a preferred version before proceeding to the next image pair comparison. No time limitations were given. The raw number of counts obtained from the user study are shown in Table 8, from which we calculated the percentage of users that preferred our method in Table 2 of the main paper.


Which of these images looks better?

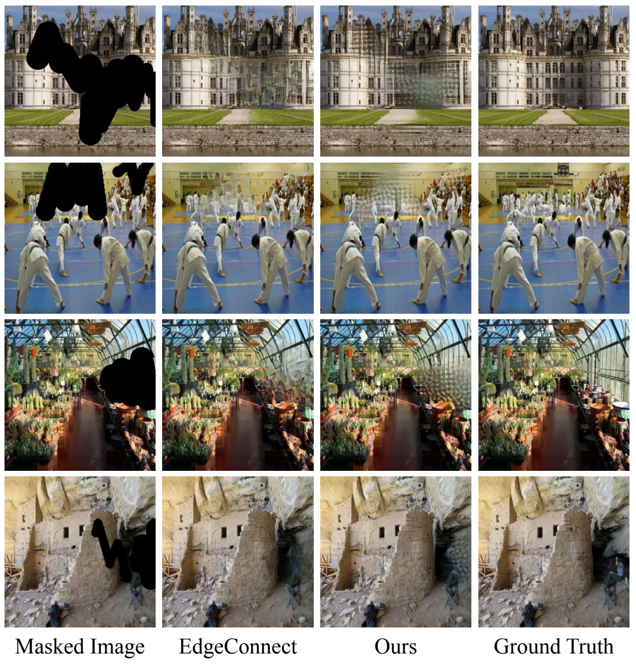
Parts of the image have been inpainted with different methods. Which of the results looks more realistic in your opinion?
Use the left and the right arrow key to switch between the images.
Press the enter key when you selected the image that looks better.
If the images appear to have the same quality, please pick one at random.
If no image is shown then please wait for a few seconds for it to show up.
There are 300 comparisons and there will be a prompt at the end, thank you!

Figure 11: Screenshot of user study presented to participants.

As shown in Table 8 and Table 2 in the main paper, more users preferred DeepFill-v2 [50] and EdgeConnect [31] than our method for large masks. As explained in the main paper, some examples with large masks contain sizeable missing regions, on which both the compared method and ours tend to generate artifacts. Figure 12 shows some of the example pairs presented to the users, where results by both methods (DeepFill-v2 vs Ours, or EdgeConnect vs Ours) contain artifacts. In these cases, it becomes difficult to compare the quality of the inpainted results, and users select the result with a less objectionable artifact. As shown in Figure 12, our method tends to produce repetitive high frequency artifacts that are displeasing. We consider this artifact pattern to be a highly likely reason to why the users preferred DeepFill-v2 or EdgeConnect over ours when the inpainted regions are very large.



(a) Examples of DeepFill-v2 vs. Ours



(b) Examples of EdgeConnect vs. Ours

Figure 12: Example image pairs with large masks shown during the user study, containing artifacts. Because both the compared method and our method contain artifacts in these cases, it is difficult to compare the quality of the generated results, and users select images containing less objectionable artifacts. Our method tends to produce repetitive high frequency artifacts that are displeasing, resulting in more users preferring DeepFill-v2 [50] and EdgeConnect [31] for large masks.

	HiFill [48] vs. Ours	Pluralistic [53] vs. Ours	DeepFill-v2 [50] vs. Ours	EdgeConnect [31] vs. Ours	Total
			Small masks		
Counts	239 vs. 736	106 vs. 869	300 vs. 675	349 vs. 626	3900
Prefer Ours	75.49%	89.13%	69.23%	64.21%	74.51%
			Large masks		
Counts	412 vs. 713	322 vs. 803	780 vs. 345	738 vs. 387	4500
Prefer Ours	63.38%	71.38%	30.67%	34.4%	49.96%

Table 8: Raw counts of the user study results on small and large masks, used to compute the user study percentage values in Table 2 in the main paper. 13 and 15 users participated in the user study with small and large masks, respectively. Each user compared 300 image pairs.

E. Analysis on Different Scale Factors

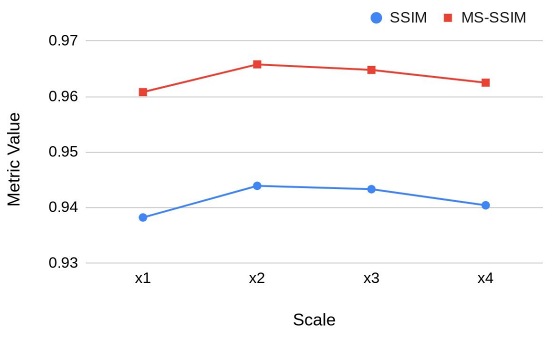
In the main paper, refinement is computed at double ($\times 2$) the resolution of coarse inpainting. To analyze the effect of scale factors on our zoom-to-inpaint framework, we retrain the *SR zoom* model (without gradient loss) on three different scale factors ($\times 2$, $\times 3$, $\times 4$) and report the results on the validation set of DIV2K (256×256 center crops) in Table 9. To avoid out-of-memory errors during training due to larger scale factors ($\times 3$ and $\times 4$ compared to $\times 2$ in the original model), all models are trained on 128×128 patches instead of 256×256 in the original zoom-to-inpaint framework. We also compare with the *No zoom* model retrained on 128×128 patches, which can be considered as scale $\times 1$. Figure 13 shows a graph with SSIM and MS-SSIM values plotted for the four models in Table 9 for small and large masks. As shown in Table 9 and Figure 13, all SR zoom models ($\times 2$, $\times 3$, $\times 4$) outperform the No zoom model ($\times 1$), demonstrating the benefits of refining at higher resolution. The best performing model is the $\times 2$ scale model for small masks, and the $\times 3$ scale model for large masks. For high scale factors (eg. $\times 4$), the refinement network has to handle very large holes, which becomes highly challenging. Note that the loss coefficients (λ) for the VGG loss and the GAN

Models	scale	Small masks				Large masks			
		PSNR ↑	SSIM ↑	MS-SSIM ↑	L1 Error ↓	PSNR ↑	SSIM ↑	MS-SSIM ↑	L1 Error ↓
No zoom	$\times 1$	29.91	0.9382	0.9608	0.00955	23.54	0.8186	0.8555	0.03066
SR zoom	$\times 2$	**30.85**	**0.9439**	**0.9658**	**0.00855**	24.23	0.8273	0.8700	0.02821
	$\times 3$	30.70	0.9433	0.9648	0.00868	**24.36**	**0.8290**	**0.8710**	**0.02739**
	$\times 4$	29.92	0.9404	0.9625	0.00932	23.91	0.8265	0.8664	0.02963

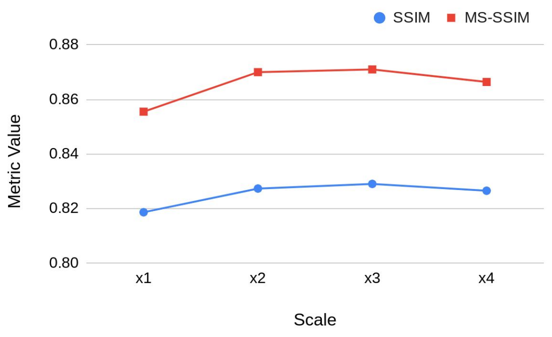
Table 9: Experiment on different SR scale factors trained with 128×128 patches. The *No zoom* model can be considered as a $\times 1$ scale version of the framework. The best performing model is $\times 2$ scale model for small masks and $\times 3$ scale model for large masks. Best values denoted in **bold**.

Frequency Range	Models				Difference		
	(a) No zoom	(b) Bicubic zoom	(c) SR zoom	(d) SR zoom+$L_{\bigtriangledown}$	(b)-(a)	(c)-(a)	(d)-(a)
Low Frequency	**0.9858**	**0.9870**	**0.9888**	**0.9907**	0.0012	0.0030	0.0049
Mid Frequency	0.9730	0.9756	0.9772	0.9793	0.0026	0.0042	0.0063
High Frequency	0.9674	0.9717	0.9727	0.9743	**0.0043**	**0.0053**	**0.0069**

Table 10: SSIM values measured for different frequency ranges using Laplacian pyramids in [7] on ablation models. It can be observed that the SSIM gain increases for higher frequency components on models (b), (c) and (d) over (a). The difference in SSIM is plotted as a bar graph in Fig. 6 in the main paper. Highest values denoted in **bold**.



(a) Small Masks



(b) Large Masks

Figure 13: Graph showing SSIM and MS-SSIM performance plotted for models with scale factors $\times 1$, $\times 2$, $\times 3$ and $\times 4$, for (a) Small Masks and (b) Large Masks.

loss were optimized on the model with $\times 2$.

F. Numerical Values for Frequency Analysis

In Table 10, we provide the raw numerical values of SSIM used for plotting the bar graph in Fig. 6 in the main paper. For this experiment, Laplacian pyramids were constructed as in [7] for the four ablation models ((a) No zoom, (b) Bicubic zoom, (c) SR zoom, (d) SR zoom+$L_{\bigtriangledown}$) and the ground truth using a traditional 5-tap Gaussian kernel, and SSIM values of the ablation models were measured for each frequency range to examine the performance gain at different frequency ranges. Low, mid and high frequency ranges each correspond to level 2, 1 and 0 in the Laplacian pyramid, respectively, with the last level (level 2) being the remaining blurred image. As shown in Table 10, the absolute SSIM values tend to be higher for lower frequencies, which are easier to reconstruct. However, the relative SSIM gain of (b), (c) and (d) over (a) is higher for higher frequencies, showing the benefits of the HR refinement models in generating high frequency components in the inpainted results.

G. Additional Visual Results

G.1. Comparison to Inpainting Methods

We provide additional comparisons to existing inpainting methods, HiFill [48], Pluralistic [53], DeepFill-v2 [50] and EdgeConnect [31], in Figure 14. Four rows at the top are results on large masks, and four rows at the bottom show results on small masks. Our zoom-to-inpaint model is able to generate accurate structure information (2nd row, 5th row) as well as high frequency details (6th row).

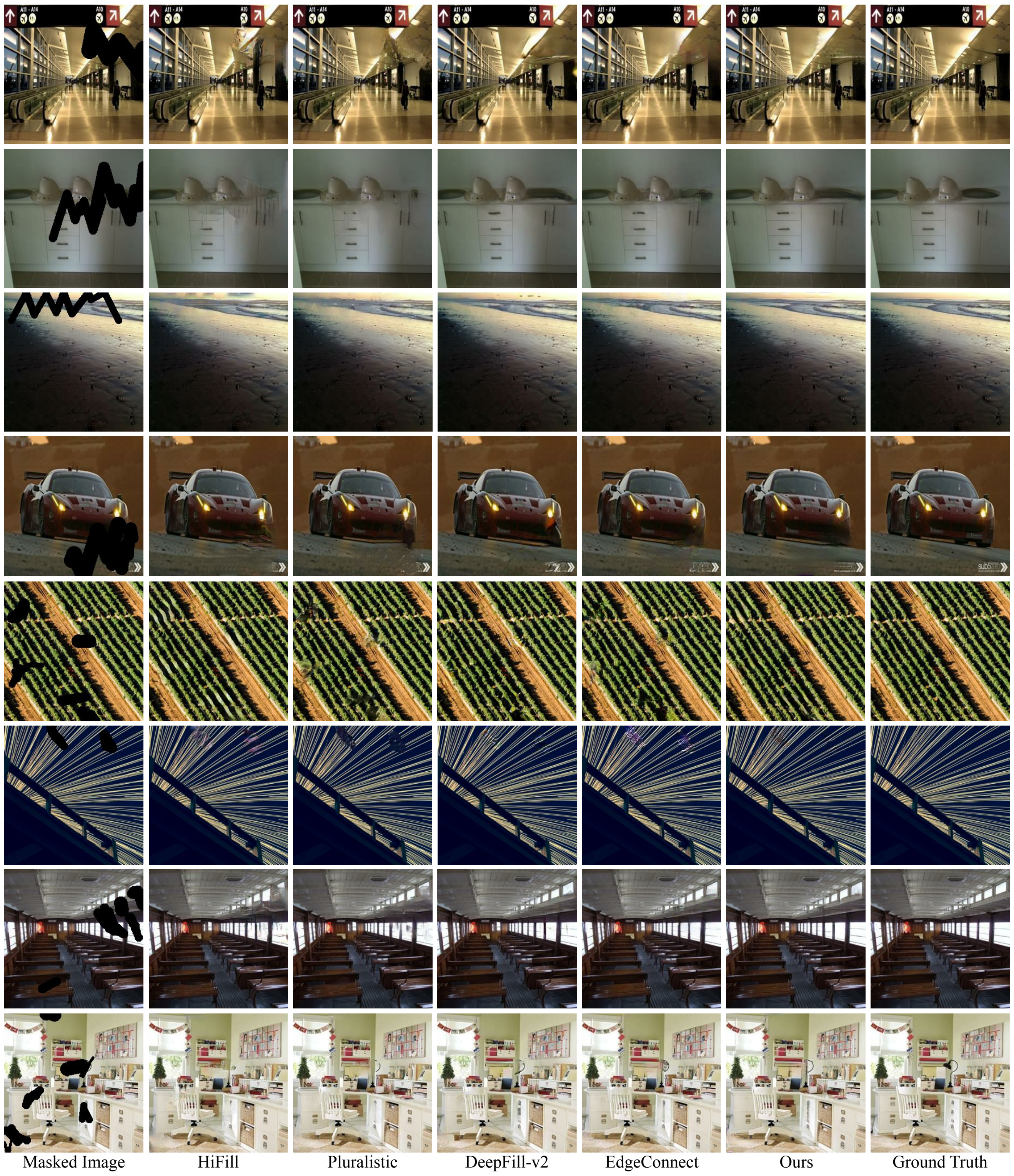


Figure 14: Additional qualitative comparisons with HiFill [48], Pluralistic [53], DeepFill-v2 [50], EdgeConnect [31]. Our zoom-to-inpaint model is able to reconstruct natural structures as shown in the examples in the 1st and 2nd rows, and generate high frequency components such as fine edges as shown in 6th and 7th rows.

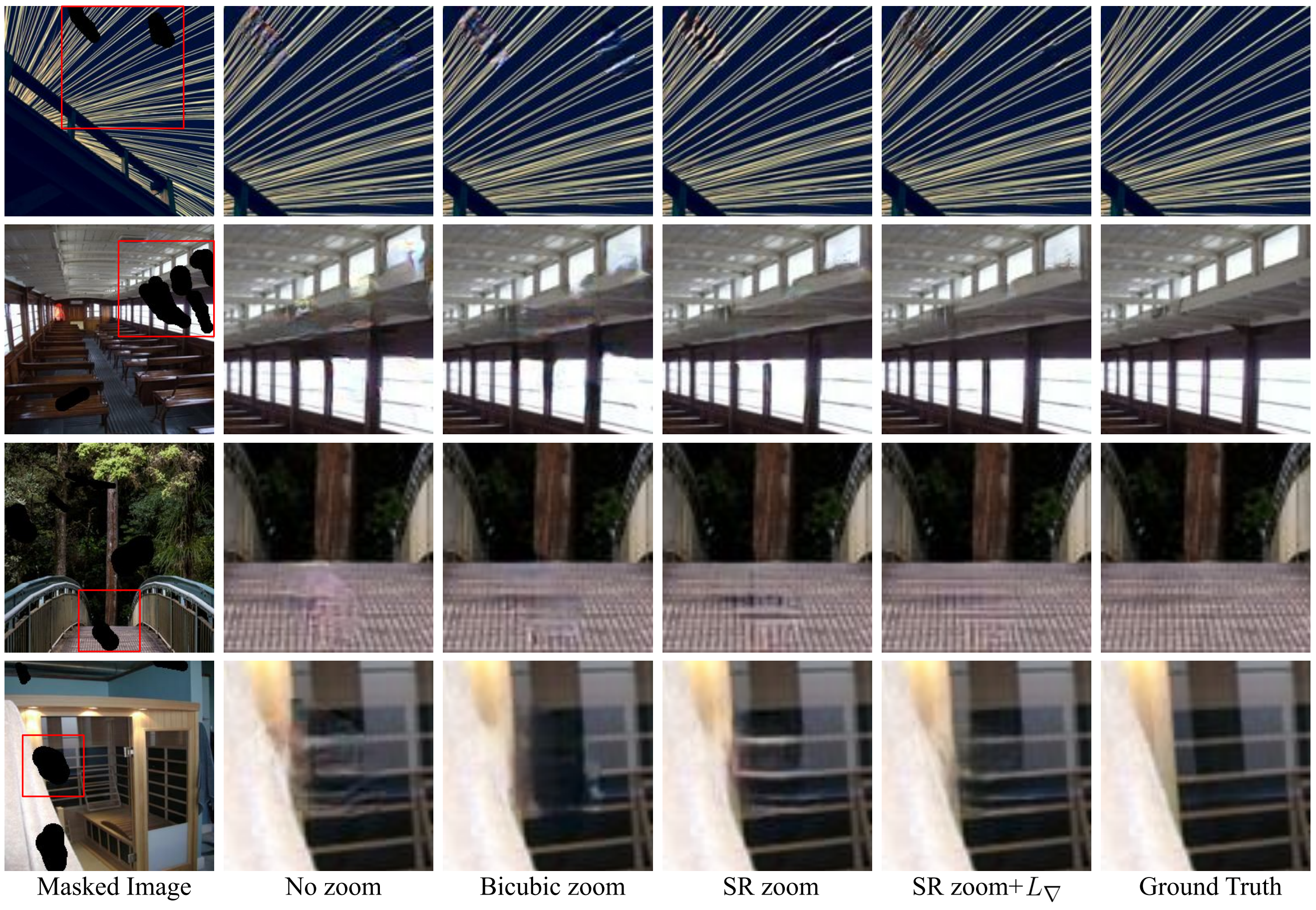


Figure 15: Additional visual comparison on results generated by the ablation models. SR zoom helps improve the generation of high frequency components, and gradient loss (L_∇) enhances the details even further.

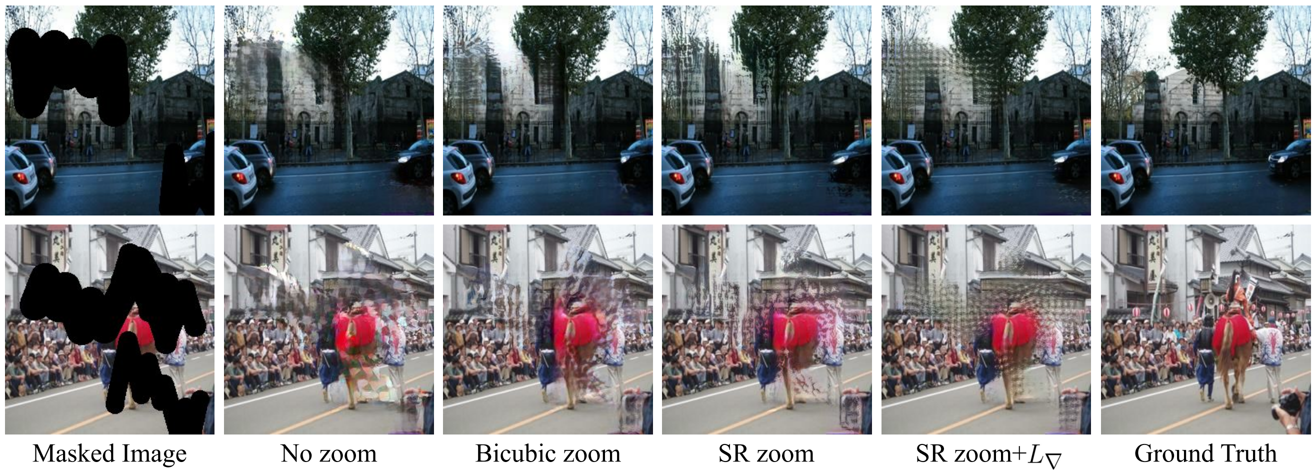


Figure 16: Artifacts generated by the ablation models on inputs with large masks. More high frequency artifacts tend to be generated in the order of No zoom, Bicubic zoom, SR zoom and SR zoom+L_∇. SR zoom+L_∇ tends to generate repetitive high frequency patterns that are unpleasant, also observed in the user study section.

G.2. Ablation Models

In Figure 15, we further provide additional results generated by the four ablation models: No zoom, Bicubic zoom, SR zoom and SR zoom+$L_{\bigtriangledown}$. As can be seen, SR zoom improves the generation of high frequency details compared to No zoom and Bicubic zoom models, and gradient loss improves them even further. We have also compared the artifacts generated by the four models in Figure 16. In these failure cases, higher frequency artifacts tend to be generated in the order of No zoom, Bicubic zoom, SR zoom and SR zoom+$L_{\bigtriangledown}$, same as the order of the ability to generate high frequency details in successfully inpainted results.