

Zoom-to-Inpaint: Image Inpainting with High-Frequency Details – Supplementary Material –

Soo Ye Kim^{1,2†} Kfir Aberman² Nori Kanazawa² Rahul Garg² Neal Wadhwa²
Huiwen Chang² Nikhil Karnad² Munchurl Kim¹ Orly Liba²

¹KAIST
Daejeon, Republic of Korea

²Google Research
Mountain View CA, USA

1. Detailed network architectures

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

1.1. Coarse network

The coarse network in our framework has an encoder-decoder-based architecture with gated convolutions [16] and ELU activation [2]. Different from the coarse network of [16], we add residual connections at each encoder and decoder level, and insert batch normalization (BatchNorm) [5] to every layer. We find that the residual connections help to propagate the information at each level and help convergence in training. 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 [12]. At the bottleneck, the gated convolutions are applied with dilation rate set to 2, 4 and 8 like in [16], 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 3. The network output is computed by blending with the input as mentioned in Equation 2 in the main paper.

1.2. Super-resolution network

The super-resolution (SR) network in our framework is shown in Figure 1. It has four cascaded residual blocks, each composed of two convolution layers with ReLU activation [3]. We employ pixel shuffle [13] at the end so that most computations are performed at low resolution (LR), to increase the size of the effective receptive field of the network, as well as to reduce computational complexity. 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.

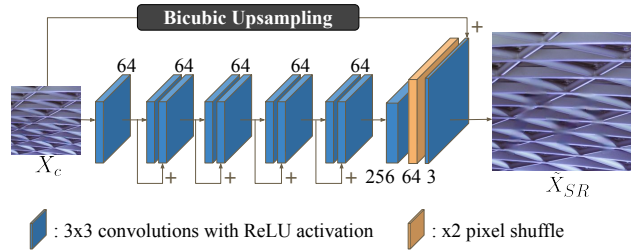


Figure 1. Architecture of the SR network in our zoom-to-inpaint framework.

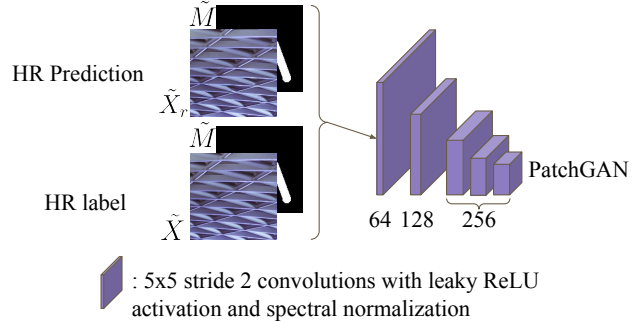


Figure 2. Architecture of the PatchGAN discriminator used during training.

We add a global residual connection with bicubic upsampling as in [7] 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 .

1.3. Refinement network

The architecture of our refinement network is shown in Figure 4. The refinement network is similar to the coarse network except that a contextual attention [15] module is added to the bottleneck. The contextual attention module copies patches from the surrounding known regions into the

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

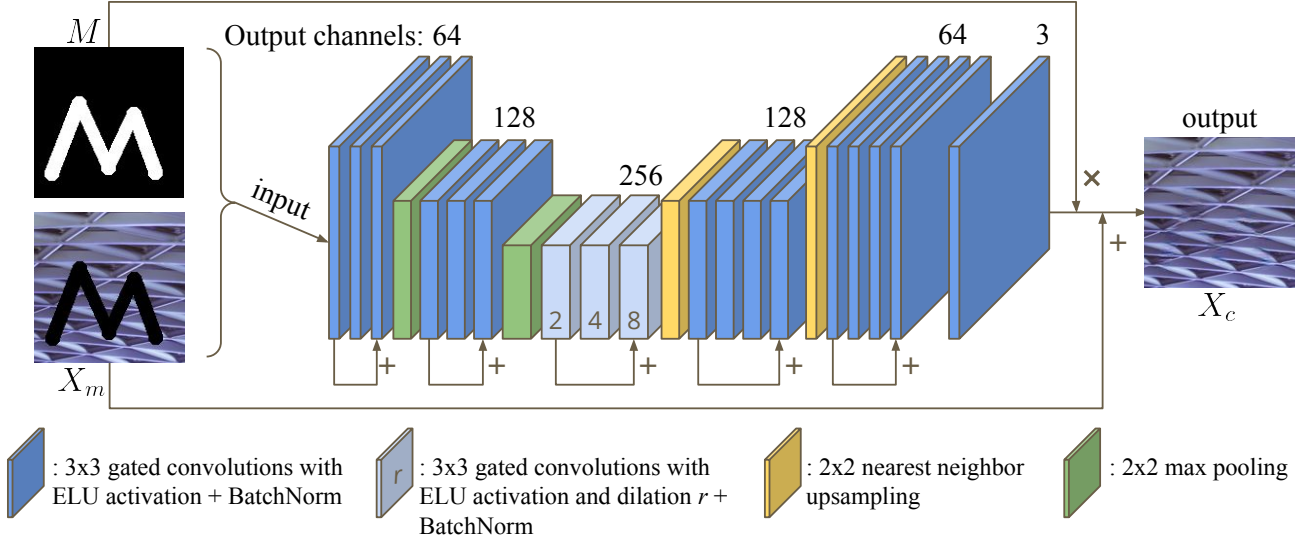


Figure 3. Architecture of the coarse inpainting network in our zoom-to-inpaint framework.

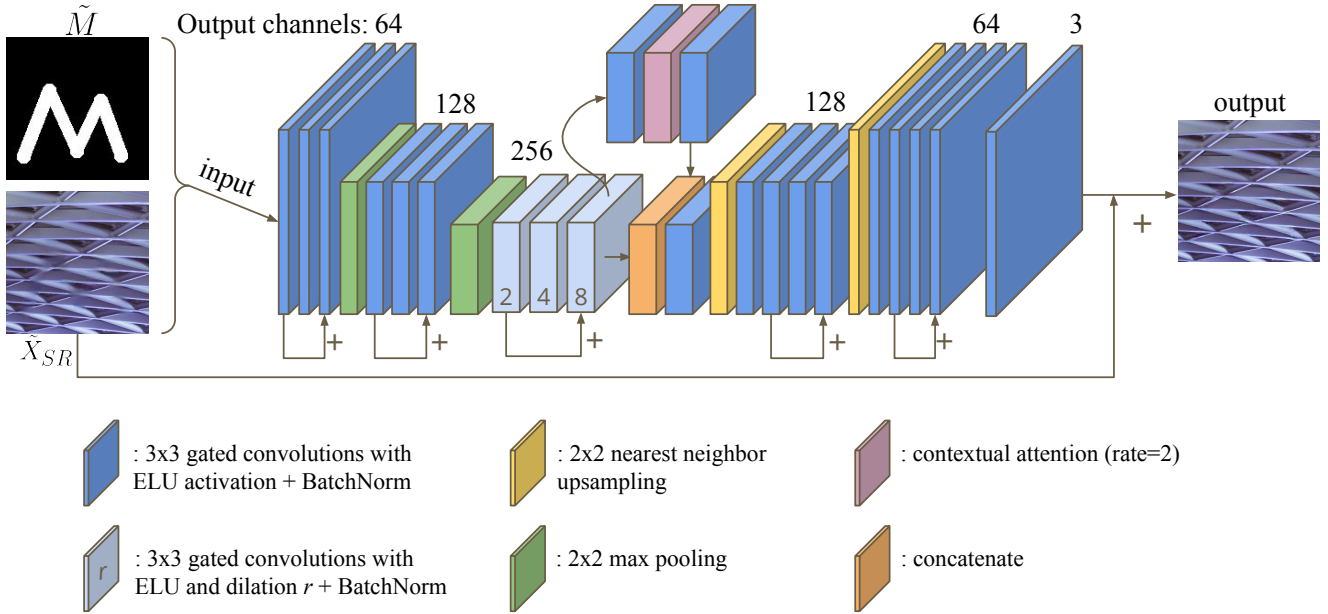


Figure 4. Architecture of the refinement network in our zoom-to-inpaint framework.

regions to be inpainted based on the computed similarity. For memory-wise efficiency, the attention map is computed on $\times 1/2$ downsampled 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 magnifica-

tion (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.

1.4. Discriminator

The network architecture of the discriminator used for training is shown in Figure 2. We use a hinge GAN loss with a PatchGAN [6] discriminator that aims to discriminate between the HR label and the HR prediction generated by the refinement network (blended with the HR label). 5×5 con-

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)
$n = 1$	$\mathcal{U}\{1, 12\}$	$\mathcal{U}\{1, d/12\}$	$\mathcal{U}\{5, 30\}$	$\mathcal{U}\{0, 2\pi\}$	$a + \mathcal{U}\{7\pi/8, 9\pi/8\}$	5.61% / 19.16%
$n = 2$	$\mathcal{U}\{4, 12\}$	$\mathcal{N}\{d/8, d/16\}$	$\mathcal{U}\{12, 40\}$	$2\pi/5 + \mathcal{U}\{\mathcal{U}\{-2\pi/15, 0\}, \mathcal{U}\{0, 2\pi/15\}\}$	$2\pi - a$	15.12% / 50.24%

Table 1. Probability distributions of mask generation parameters for masks at $n = 1, 2$. d denotes the diagonal length of the image.

Method	HiFill [14]	Pluralistic [18]	DeepFill-v2 [16]	EdgeConnect [11]	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 2. Comparison of the number of parameters and average inference time in milliseconds (ms) 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.

volution filters with stride 2, and leaky ReLU [9] activation are used. Spectral normalization [10] is applied at every layer for stable training of the GAN framework.

2. Mask generation

Table 1 shows a detailed configuration of the parameters used for generating masks at $n = 1$ (*masks*) and $n = 2$ (*large masks*). Each mask generation parameter is drawn from a probability distribution shown in Table 1, and a random mask is generated using those parameters following the mask generation scheme in DeepFill-v2 [16]. A mask is generated by drawing random strokes on a 256×256 tensor filled with zeros. First, a random starting location (x, y) and a random number of vertices is chosen for each stroke. Then for each stroke, a piece-wise stroke is drawn between the vertices with a randomly chosen length of piece-wise stroke (distance between two vertices) and a random angle. A random thickness is fixed for each stroke. The *large masks* ($n = 2$) are generated with the original parameters used in [16], and *masks* ($n = 1$) are generated using adjusted parameters, modified so that the invalid pixel ratio is smaller and the generated masks are more confined.

3. Complexity analysis

We compare the number of parameters and inference time of HiFill [14], Pluralistic [18], DeepFill-v2 [16], EdgeConnect [11] and Ours in Table 2. The average inference time in milliseconds (ms) is measured over 100 center crops of size 256×256 of the DIV2K validation dataset for Pluralistic [18], DeepFill-v2 [16], EdgeConnect [11] and Ours, which were all trained on 256×256 images. For HiFill [14], 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 on the CNN for all 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 bicubic downscaling. Because our refinement network handles higher-resolution images to generate better high-frequency details, the inference time is longer compared to other inpainting methods. However, an average inference time of 293 ms still allows interactive inpainting with users. The inference time was measured on an NVIDIA Tesla T4 GPU.

4. Extended quantitative evaluations

4.1. Ablation study

We provide an extended ablation study result in Table 3 on 256×256 center crops of the DIV2K validation set on both mask sizes. 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 refinement is applied at the original resolution like in other conventional 2-stage inpainting frameworks. For (ii) Bicubic zoom, we replace the SR component with bicubic upsampling, and for (iii), we add back the SR zoom. (i), (ii) and (iii) are trained without the gradient loss, L_{∇} . Lastly, (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

Ablations	ch.	Masks				Large Masks				Number of Parameters
		PSNR \uparrow	SSIM \uparrow	MS-SSIM \uparrow	L1 Error \downarrow	PSNR \uparrow	SSIM \uparrow	MS-SSIM \uparrow	L1 Error \downarrow	
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 3. 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.

Frequency Range	Models				Difference		
	(a) No zoom	(b) Bicubic zoom	(c) SR zoom	(d) SR zoom+ L_{∇}	(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 4. SSIM values measured for different frequency ranges using Laplacian pyramids in [1] 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 Figure 6 in the main paper. Highest values denoted in **bold**.

same number of 64 output channels. To test the effect of the network capacity 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 parameters (4.82M) than the SR zoom models. The quantitative results along with the number of parameters are provided in Table 3. As shown in Table 3, 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.

4.2. Numerical values for frequency analysis

In Table 4, we provide the raw numerical values of SSIM used for plotting the bar graph in Figure 6 of the main paper. For this experiment, Laplacian pyramids [1] were constructed for the four ablation models ((a) No zoom, (b) Bicubic zoom, (c) SR zoom, (d) SR zoom+ L_{∇}) 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 4, 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 gener-

Method	Places2 (256 \times 256)			
	PSNR \uparrow	SSIM \uparrow	MS-SSIM \uparrow	L1 Error \downarrow
Values copied from the publications				
HiFill [14]	-	-	0.8840	0.05439
Pluralistic [18]	-	-	-	-
DeepFill-v2 [16]	-	-	-	0.09100
EdgeConnect [11]	27.95	0.9200	-	0.01500
Our measurements – Masks				
HiFill [14]	31.12	0.9586	0.9742	0.00744
Pluralistic [18]	33.23	0.9670	0.9807	0.00558
DeepFill-v2 [16]	34.03	0.9719	0.9834	0.00485
EdgeConnect [11]	33.98	0.9718	0.9841	0.00388
Ours	34.78	0.9755	0.9863	0.00357
Our measurements – Large Masks				
HiFill [14]	24.94	0.8891	0.9134	0.02034
Pluralistic [18]	26.17	0.9022	0.9191	0.01784
DeepFill-v2 [16]	26.77	0.9158	0.9326	0.01536
EdgeConnect [11]	27.61	0.9166	0.9382	0.01328
Ours	27.71	0.9202	0.9415	0.01314

Table 5. Extended quantitative comparison on Places2. Values copied from the original publications are denoted in gray for reference, but they are not directly comparable to our measurements as they were all tested under different settings, as summarized in Table 6. Values in **bold** denote best performance.

ating high-frequency components in the inpainted results.

4.3. Quantitative comparison

We provide an extended table on the quantitative comparison with other inpainting methods [11, 14, 16, 18] in Ta-

Method	Testing conditions in original publications
HiFill [14]	Tested on Places2 validation set, randomly cropped by 512×512 . Used random masks from [8] as well as their own random object masks.
Pluralistic [18]	Only provided quantitative evaluations on ImageNet (256×256). Proposed own random masks with random lines, circles and ellipses.
DeepFill-v2 [16]	Tested on Places2 validation set (256×256). Proposed random brush stroke masks that can be generated on-the-fly during training (same as our <i>large masks</i>).
EdgeConnect [11]	Tested on Places2 (256×256) but no mention on which images were used. Used random masks from [8].

Table 6. Descriptions on quantitative evaluations performed in the original publications, which were used for obtaining the values in gray in Table 5 that were copied from the original publications.

Method	Places2 (256×256)		DIV2K (256×256)	
	LPIPS ↓	FID ↓	LPIPS ↓	FID ↓
<i>Masks</i>				
HiFill [14]	0.0306	24.71	0.0258	12.85
Pluralistic [18]	0.0236	82.43	0.0219	128.21
DeepFill-v2 [16]	0.0178	16.58	0.0172	8.61
EdgeConnect [11]	0.0177	16.41	0.0171	8.51
Ours	0.0175	14.32	0.0147	7.14
<i>Large Masks</i>				
HiFill [14]	0.0889	56.86	0.0989	56.65
Pluralistic [18]	0.0745	62.73	0.0810	150.28
DeepFill-v2 [16]	0.0585	38.22	0.0636	33.89
EdgeConnect [11]	0.0554	35.41	0.0611	32.57
Ours	0.0603	38.85	0.0628	42.25

Table 7. Quantitative comparison using perceptual metrics, LPIPS and FID. Values in **bold** denote best performance.

ble 5. For reference, in addition to the values measured on our Places2 test set, which were also reported in the main paper, we have added the metric values from the original publications measured on Places2. 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 *masks* and *large masks*. Furthermore, we provide a comparison on perceptual metrics – FID [4] and LPIPS [17] in Table 7. Our model outperforms all other methods for *masks*, and is 2nd or 3rd best for *large masks*. The results on perceptual metrics seem to correspond to those of the user study shown in Section 5, where we also provide an analysis on *large masks*.

5. User study

As mentioned in the main paper, we conducted a user study, in which 13 users evaluated the inpainting results. 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 [11, 14, 16, 18]. The appearing order of methods (whether ours comes first, or the com-

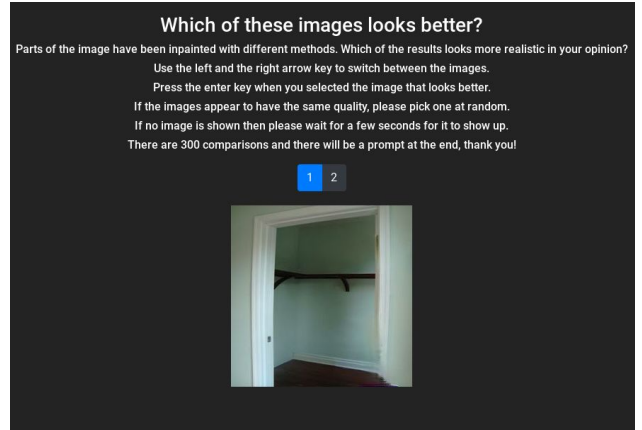


Figure 5. Screenshot of user study presented to participants.

pared 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 5. Participants could toggle between the two compared images and select a preferred version before proceeding to the next image pair for comparison. No time limitations were given. The raw number of counts obtained from the user study is shown in Table 8, from which we calculated the percentage of users that preferred our method in Table 2 of the main paper.

Results on *large masks* and failure case analysis. In Table 8, we show the results on *large masks* that were additionally evaluated on another 15 users. Although our approach outperforms all other methods for *masks*, for *large masks*, more users preferred DeepFill-v2 [16] and EdgeConnect [11]. As briefly explained in the Conclusion section in the main paper, some examples with *large masks* contain sizeable missing regions, on which both the compared method and ours tend to generate severe artifacts. Figure 6 shows some of the example pairs (DeepFill-v2 vs Ours, or EdgeConnect vs Ours) containing artifacts that were presented to the users. In these extreme cases, it becomes difficult to compare the quality of the inpainted re-

	HiFill [14] vs. Ours	Pluralistic [18] vs. Ours	DeepFill-v2 [16] vs. Ours	EdgeConnect [11] vs. Ours	Total
<i>Masks</i>					
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%
<i>Large Masks</i>					
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 *masks* (top) and *large masks* (bottom). The values at the top were used to compute the preference rate in Table 2 of the main paper. 13 and 15 users participated in the user study with *masks* and *large masks*, respectively, and each user compared 300 image pairs.

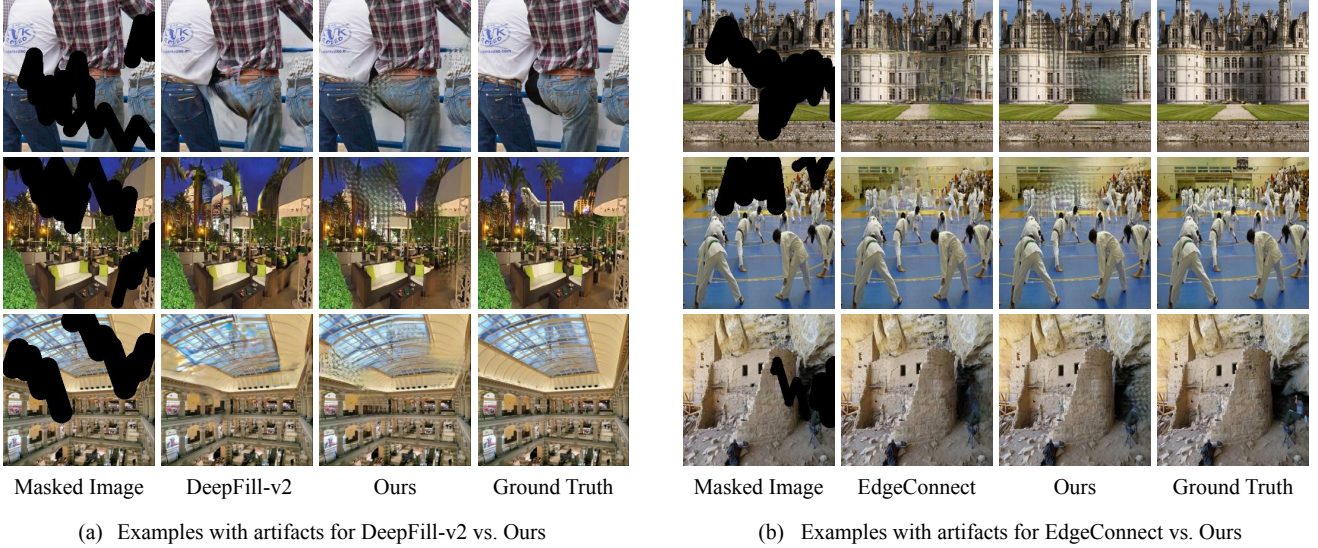


Figure 6. Failure case examples with *large masks* shown during the user study. As both the compared method and our method contain artifacts in these extreme cases with very large masks, it is difficult to compare the quality of the generated results, and users select images containing less objectionable artifacts. Our method tends to produce a repetitive high-frequency pattern that is displeasing, resulting in more users preferring DeepFill-v2 [16] and EdgeConnect [11] for *large masks*. Note that these are not average results and more typical results are shown in Figure 4 (b) of the main paper and Figure 8 (b).

sults, and users select the result with a less objectionable artifact. As shown in Figure 6, our method tends to produce a repetitive high-frequency pattern that is displeasing in these cases. We consider this artifact pattern to be a highly likely reason as to why users preferred DeepFill-v2 or EdgeConnect over ours when the inpainted regions are very large. Note that these are failure cases on extreme cases, and not average results of our method on *large masks*. More typical results on *large masks* are shown in Figure 4 (b) of the main paper and Figure 8 (b) of this Supplementary Material.

6. Analysis on different scale factors

In the main paper, refinement is performed at double ($\times 2$) the resolution of the coarse output. 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. As in the original framework, each component (coarse, SR and refinement network) is pretrained first, before the end-to-end progressive learning. We empirically find the loss coefficients (λ) on the model with $\times 2$ and use the same values for models with $\times 3$ and $\times 4$. We also compare with the *No zoom* model retrained on 128×128 patches, which can be considered as scale $\times 1$. Figure 7 shows a graph with SSIM and MS-SSIM values plotted for the four models in Table 9 for both mask sizes. As shown in Table 9 and Figure 7, all SR zoom models ($\times 2$, $\times 3$, $\times 4$) outperform the No zoom model ($\times 1$), demonstrating

Models	scale	<i>Masks</i>				<i>Large Masks</i>			
		PSNR \uparrow	SSIM \uparrow	MS-SSIM \uparrow	L1 Error \downarrow	PSNR \uparrow	SSIM \uparrow	MS-SSIM \uparrow	L1 Error \downarrow
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 *masks* and $\times 3$ scale model for *large masks*. Best values denoted in **bold**.

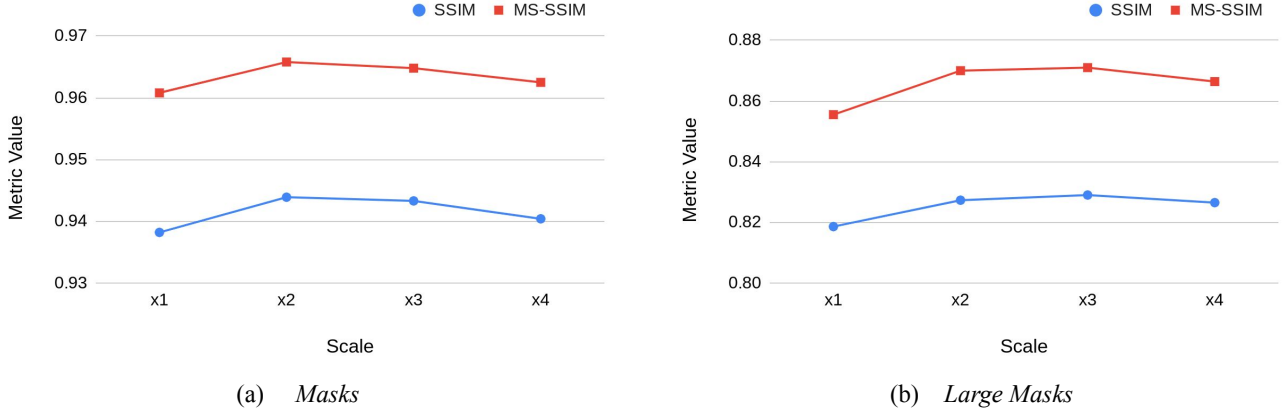


Figure 7. Graphs showing SSIM and MS-SSIM performance plotted for models with scale factors $\times 1$, $\times 2$, $\times 3$ and $\times 4$, for (a) *Masks* and (b) *Large Masks*.

the benefits of refining at higher resolution. The best performing model is the $\times 2$ scale model for *masks*, and the $\times 3$ scale model for *large masks*. There is a trade-off with the scale factor increase, as more information can be learned from HR labels with higher scale factors possibly leading to better reconstruction quality, but at the same time, the refinement network has to handle larger holes, which becomes highly challenging. We expect there would also be a correlation with the network capacity when searching for an optimal scale factor for HR refinement, which would be related to the ability to learn from HR labels.

7. Additional visual results

7.1. Comparison to other inpainting methods

We provide additional comparisons to existing inpainting methods, HiFill [14], Pluralistic [18], DeepFill-v2 [16] and EdgeConnect [11], in Figure 8. Four rows at the top are results on *masks*, and four rows at the bottom show results on *large masks*. Our zoom-to-inpaint model is able to generate accurate structure information (1st row, 5th row, 6th row) as well as high-frequency details (2nd row, 3rd row).

7.2. Ablation models

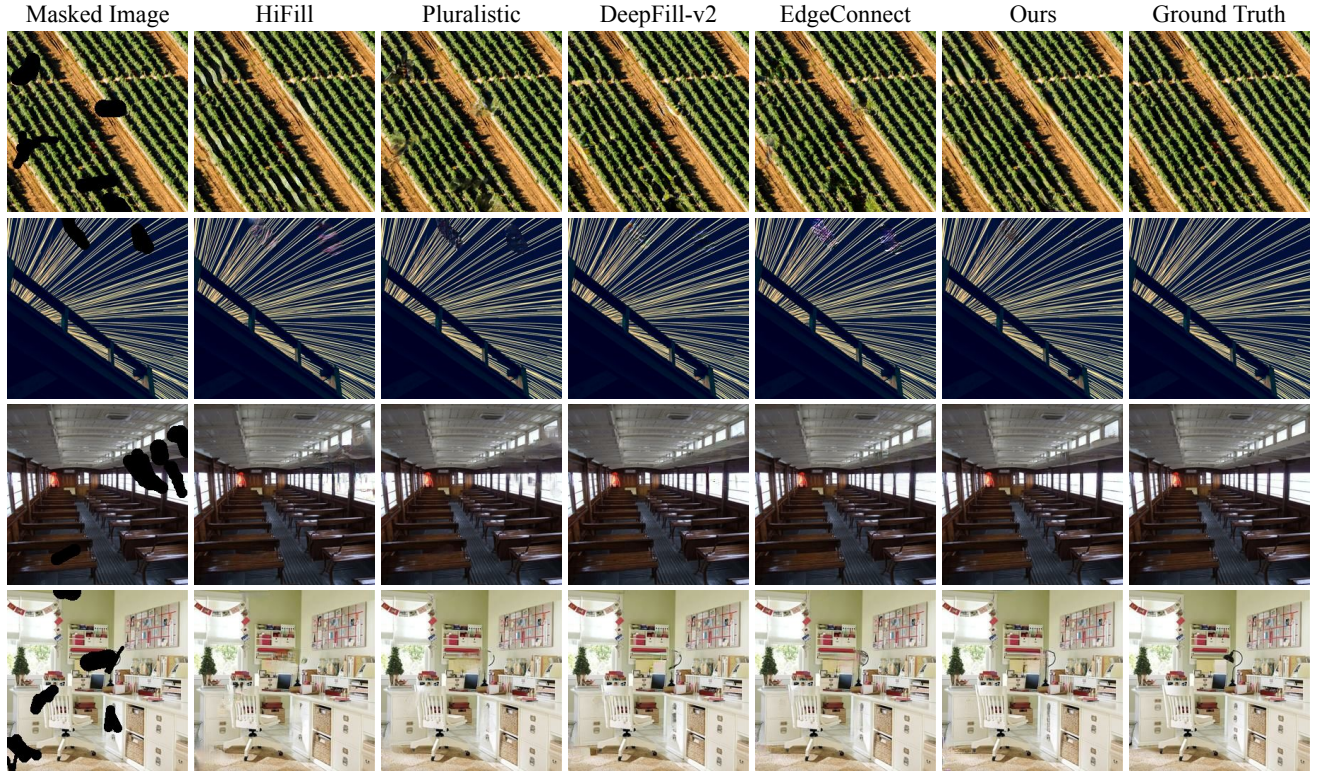
In Figure 9, we further provide additional results generated by the four ablation models: No zoom, Bicubic zoom, SR zoom and SR zoom+ L_{∇} . As shown, *SR zoom* improves the generation of high-frequency details compared to *No zoom* and *Bicubic zoom* models, and gradient loss improves them even further.

7.3. Intermediate results

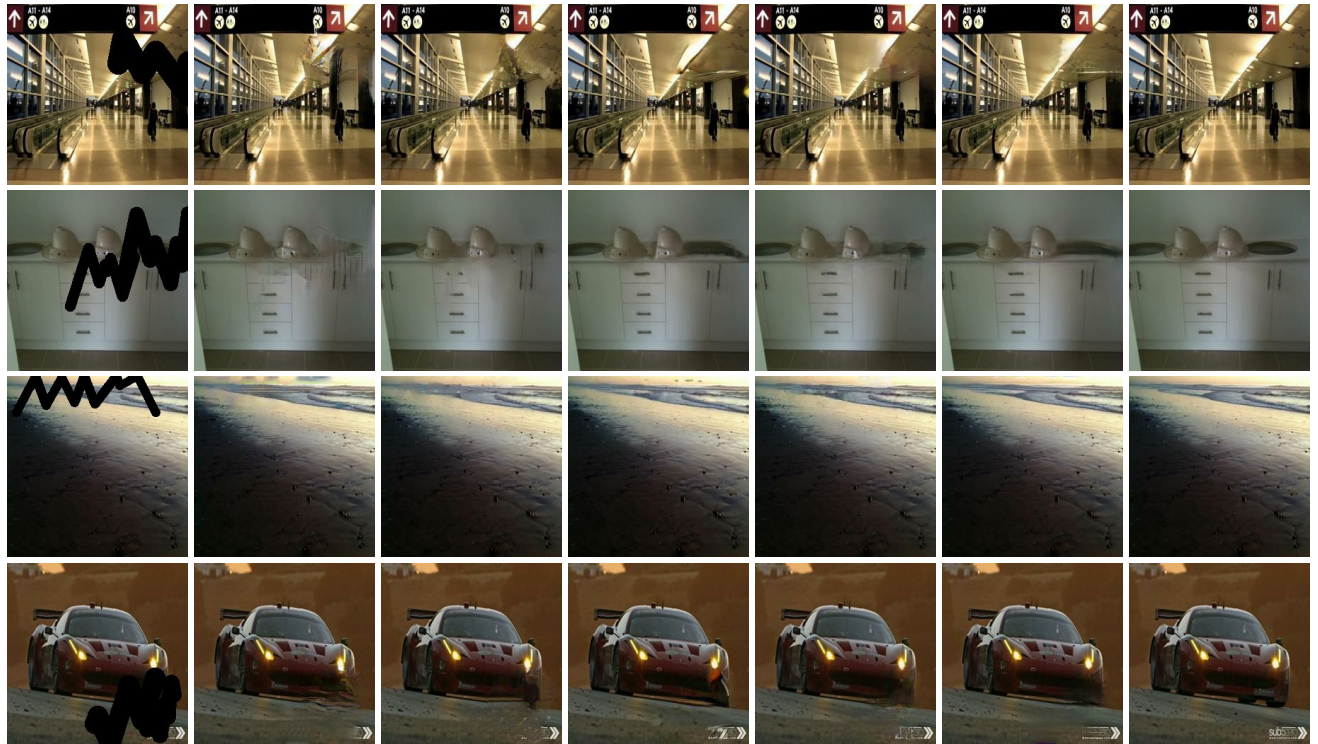
Our zoom-to-inpaint model is a 4-stage framework, and the intermediate output of each stage can be extracted. Figure 2 in the main paper shows the actual intermediate results produced by our framework. In Figure 10, we provide additional examples of intermediate results, specifically X_m , X_c , \tilde{X}_{SR} , \tilde{X}_r and X_r , along with LR and HR labels X and \tilde{X} .

References

- [1] Peter Burt and Edward Adelson. The laplacian pyramid as a compact image code. *IEEE Transactions on Communications*, 31(4):532–540, 1983. 4
- [2] Djork-Arné Clevert, Thomas Unterthiner, and Sepp Hochreiter. Fast and accurate deep network learning by exponential



(a) *Masks*



(b) *Large Masks*

Figure 8. Additional qualitative comparisons to HiFill [14], Pluralistic [18], DeepFill-v2 [16] and EdgeConnect [11] on (a) *Masks* and (b) *Large Masks*. Our zoom-to-inpaint model is able to reconstruct natural structures as shown in the examples in the 1st, 5th and 6th rows, and generate high-frequency components such as fine edges as shown in 2nd and 3rd rows.

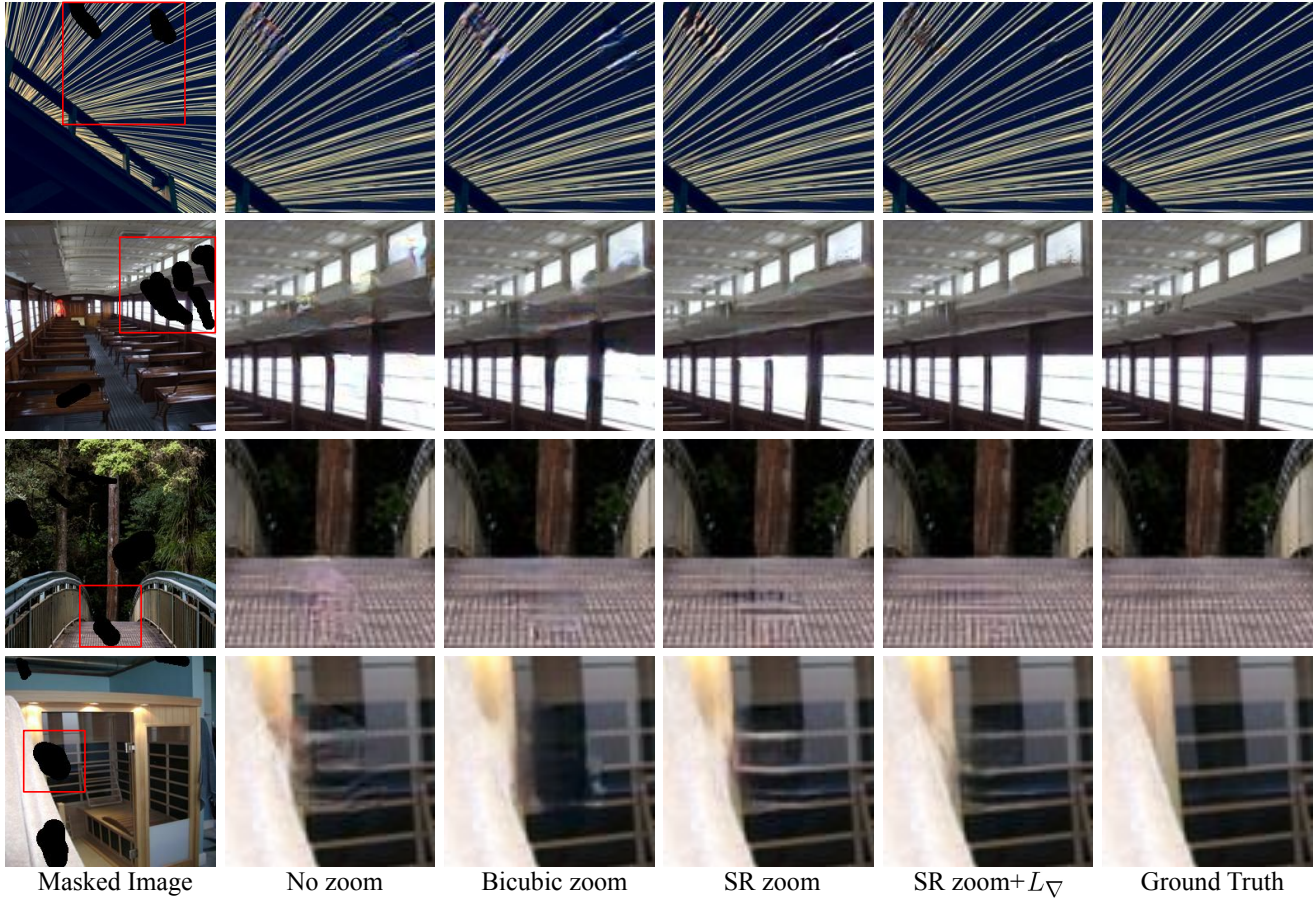


Figure 9. Additional visual comparison on results generated by the ablation models. SR zoom helps improve the generation of high-frequency components, and gradient loss (L_{∇}) enhances the details even further.

- linear units (elus). In *International Conference on Learning Representations*, 2016. 1
- [3] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *International Conference on Artificial Intelligence and Statistics*, 2011. 1
- [4] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *International Conference on Neural Information Processing Systems*, 2017. 5
- [5] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, 2015. 1
- [6] 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
- [7] 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. 1
- [8] 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. 5
- [9] Andrew L. Maas, Awni Y. Hannun, and Andrew Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *International Conference on Machine Learning*, 2013. 3
- [10] Takeru Miyato, Toshiki Kataoka, Masanori Koyama, and Yuichi Yoshida. Spectral normalization for generative adversarial networks. In *International Conference on Learning Representations*, 2018. 3
- [11] 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. 3, 4, 5, 6, 7, 8
- [12] Augustus Odena, Vincent Dumoulin, and Chris Olah. Deconvolution and checkerboard artifacts. *Distill*, 2016. 1
- [13] 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

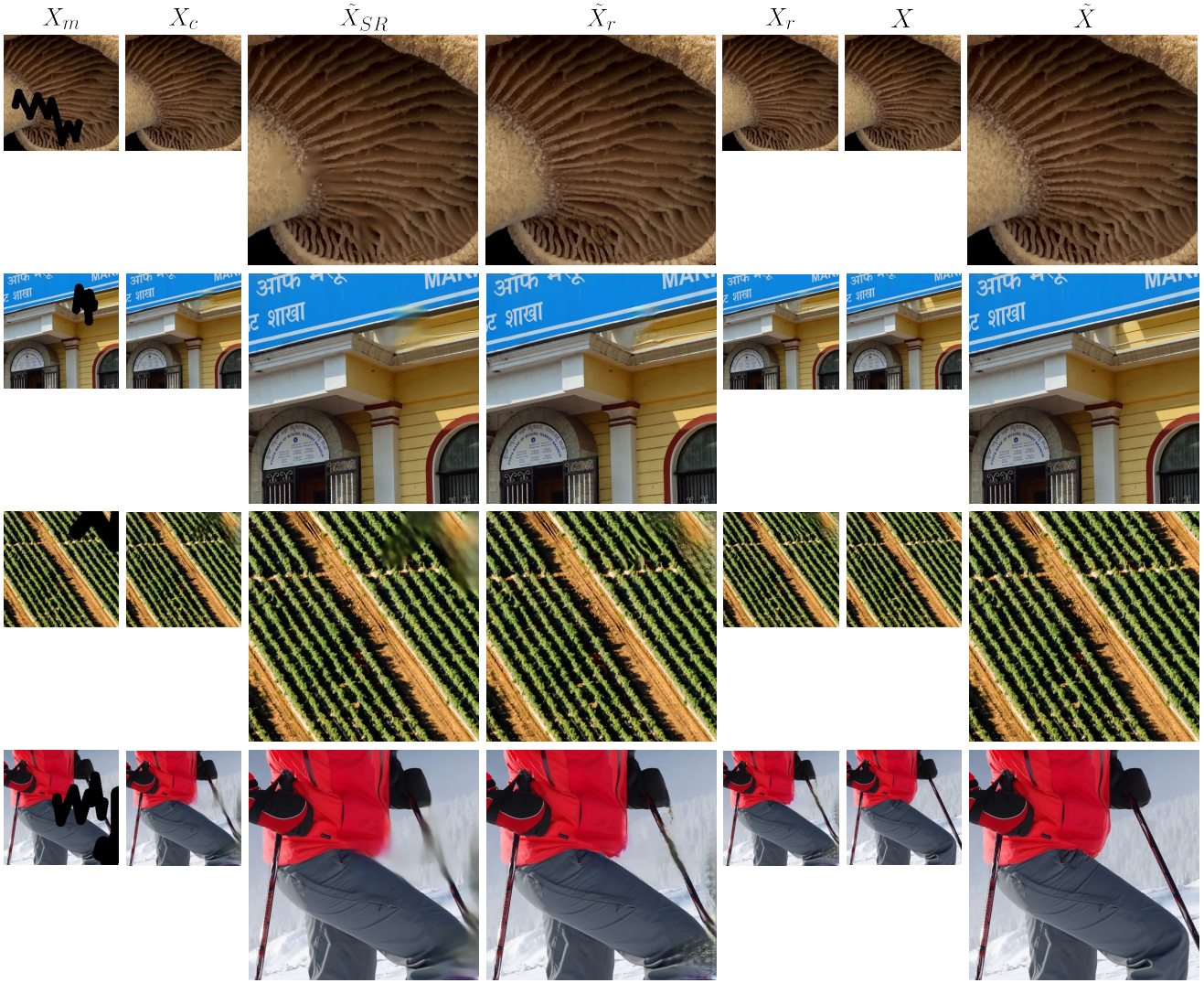


Figure 10. Intermediate results extracted from our framework.

IEEE Conference on Computer Vision and Pattern Recognition, 2016. 1

- [14] 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. 3, 4, 5, 6, 7, 8
- [15] 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. 1
- [16] 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. 1, 3, 4, 5, 6, 7, 8
- [17] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. In *IEEE Conference on*

Computer Vision and Pattern Recognition, 2018. 5

- [18] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Pluralistic image completion. In *IEEE Conference on Computer Vision and Pattern Recognition*, 2019. 3, 4, 5, 6, 7, 8