Structure First Detail Next: Image Inpainting with Pyramid Generator

Shuyi Qu[1,2]   Zhenxing Niu[2]   Kaizhu Huang[1]   Jianke Zhu[3]
Matan Protter[2]   Gadi Zimerman[2]   Yinghui Xu[2]
[1]Xi'an Jiaotong-Liverpool University   [2]Alibaba DAMO Academy
[3]Zhejiang University
[1]`firstname.lastname@xjtlu.edu.cn`   [2]`firstname.lastname@alibaba-inc.com`
[3]`jkzhu@zju.edu.cn`

Abstract

Recent deep generative models have achieved promising performance in image inpainting. However, it is still very challenging for a neural network to generate realistic image details and textures, due to its inherent spectral bias. By our understanding of how artists work, we suggest to adopt a 'structure first detail next' workflow for image inpainting. To this end, we propose to build a Pyramid Generator by stacking several sub-generators, where lower-layer sub-generators focus on restoring image structures while the higher-layer sub-generators emphasize image details. Given an input image, it will be gradually restored by going through the entire pyramid in a bottom-up fashion. Particularly, our approach has a learning scheme of progressively increasing hole size, which allows it to restore large-hole images. In addition, our method could fully exploit the benefits of learning with high-resolution images, and hence is suitable for high-resolution image inpainting. Extensive experimental results on benchmark datasets have validated the effectiveness of our approach compared with state-of-the-art methods.

1. Introduction

Image inpainting aims at restoring corrupted images with reasonable and relevant contents, which is widely used in many real-world applications including image restoration, object removal, and photo editing. In recent years, deep generative models are broadly adopted to solve this problem, and have achieved impressive results [12, 25, 28, 16, 33, 21, 30, 11, 13, 15, 32]. However, it is still very challenging to generate coherent and realistic image details [29, 9]. Such challenges probably stem from the neural networks' spectral bias [19], *i.e.*, neural networks are biased towards learning low frequency components instead of high frequency details. To alleviate this problem, most state-of-the-art methods adopt a coarse-to-fine framework. Taking the DeepFillv2 [29] as an example, images are first restored with an coarse network, and then the details are further refined at the next stage.

Furthermore, there is a conflict in modeling image global structures and image local details [26, 9]. To tackle this problem, Yang *et al*. propose a joint loss function where the content term and texture term are used to model image global structure and local texture respectively [26]. However, modeling image structures often prefers to learn from large image regions or even the whole image, while modeling image details prefers to learn from small image patches. This dilemma requires different receptive field sizes, and it is hard to satisfy in single neural network.

By our understanding, image inpainting looks like artists drawing a picture for a scene. Artists usually first draw the global structures/sketches of the scene and then refine its local details [4, 6, 17]. Inspired by that, we suggest to explicitly separate the restoration of image structures from that of details, and follow the philosophy of '**structure first detail next**'. Thus, we propose to engage distinct sub-generators to restore image structures and details, respectively.

In particular, we build a Pyramid Generator for image inpainting. The pyramid generator is constructed by stacking several sub-generators from bottom to top layer. Inputs are downsampled into different resolutions and are fed into corresponding sub-generators. The sub-generators at low-layers focus on restoring the image global structures, while the high-layer sub-generators focus on restoring image local details. Given an image, it will be gradually inpainted by going through the pyramid from bottom to top. Obviously, our novel inpainting approach actually takes the 'structure first detail next' workflow.

Note that the previous *coarse-to-fine frameworks* in [29] only cascade two similar sub-generators (*i.e.*, they are trained with inputs of same resolution) and fail to separate effectively the image structures restoration and image details restoration. On the contrary, with the *multi-scale*

multi-layer stacking framework, our pyramid generator can not only effectively separate the structures and details modeling and indeed practise restoring structures before details.

It is well-known that image inpainting becomes much more challenging when the corrupted area is relative large [14, 9, 27]. Some recent work [9] illustrate that such **large-hole challenges** could be effectively alleviated by a progressive learning strategy, *i.e.*, first learn to restore small hole and then learn to restore a large hole.

Our approach exactly aligns to such strategy. As shown in Figure 2, both input image and mask image are simultaneously downsampled in our pyramid, thus the ratio of hole size to image size is identical for all layers. However, whether the hole size is large or small should be measured with respect to the size of convolutional receptive field (instead of image size). As shown in Figure 1, since all subgenerators at different layers have the same architecture as well as the same receptive field size, the hole size is *relatively* small for the low-layer sub-generator. Thus, the low-layer inpainting looks like the small-hole inpainting task. In contrast, the high-layer inpainting looks like large-hole inpainting. With our bottom-up inpainting workflow, we are actually aligning to the hole-increasing strategy, *i.e.*, first conduct small-hole inpainting and then large-hole inpainting. As a result, our pyramid generator has advantages of dealing with large holes.

Finally, our pyramid generator is more suitable for **high-resolution image inpainting**. Many previous works have shown that increasing the resolution of training images could benefit the high-resolution image inpainting [31, 27]. Our experimental results also validate such observation, as shown in Section 4.3. However, we find that such performance gain is still limited if we just directly train the existing models (*e.g.*, DeepFillv2 [29]) with high-resolution images (*e.g.*, on 512×512 images), as shown in Fig 7.

We argue that such observation is also related to the large-hole challenge. Since high-resolution training images often come along with large holes, if we directly train a generator with high-resolution images, we have to face the large-hole challenge. Nevertheless, due to the capability of handling the large-hole challenge, our approach could fully exploit the advantages of learning with high-resolution images. As a result, our approach is more suitable for high-resolution image inpainting.

To the best of our knowledge, this is the first work that adopts a pyramid of GANs for image inpainting. We highlight our contributions as follows:

- A Pyramid Generator is proposed to conduct image inpainting by following the strategy of 'structure first detail next'. With our dedicated multi-scale generative architecture, the conflict between image global structures and local details restoration could be alleviated.
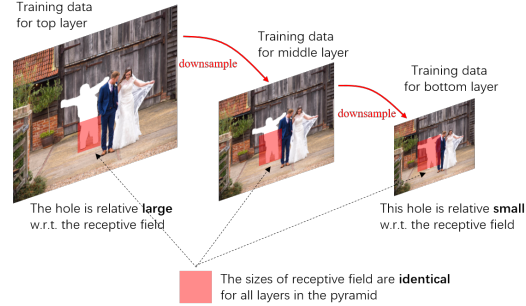

Figure 1. Bottom-layer inpainting task (right) looks like a small-hole inpainting task, while the top-layer inpainting (left) looks like a large-hole inpainting. It is obvious that the mask size of bottom layer is relative small with respect to the same receptive field size, while the mask size of top layer is relative large.

- Our pyramid generator has a learning scheme of progressively increasing hole size, which allows it to restore large holes.

- Our approach could totally reap the benefits of learning with high-resolution images, and hence is suitable for inpainting high-resolution images.

2. Related Work

2.1. Image Inpainting

Conventional image inpainting methods often utilize low-level image statistics [2]. PatchMatch [3] fills holes by searching similar patches from unfilled area. Although effective at textured images, when inpainting complex images with global structures, these methods often generate artifacts or incoherent content.

Recently, many deep learning based methods are proposed with great improvement [18, 7, 14, 29, 27]. Context encoder [18] presents the first attempt to apply the convolution neural network on image inpainting. Iizuka *et al.* improve the architecture with both global and local discriminators to keep consistency [7]. Partial convolution [14] is proposed to handle free-formed masks by using only valid pixels as conducting convolutions. Recently, DeepFillv2 [29] introduces a contextual attention module and a coarse-to-fine learning framework, which has significantly improved the inpainting performance. HiFill [27] focuses on high-resolution inpainting task and proposes a contextual residual aggregation mechanism.

2.2. Multi-scale Mechanism

Multi-scale design is applied broadly in many computer vision tasks thanks to its progressive refinement property [5, 20, 23, 22]. One representative work is SinGAN [22], where a pyramid of GANs is proposed to generate a realistic image sample of arbitrary size and aspect ratio. In
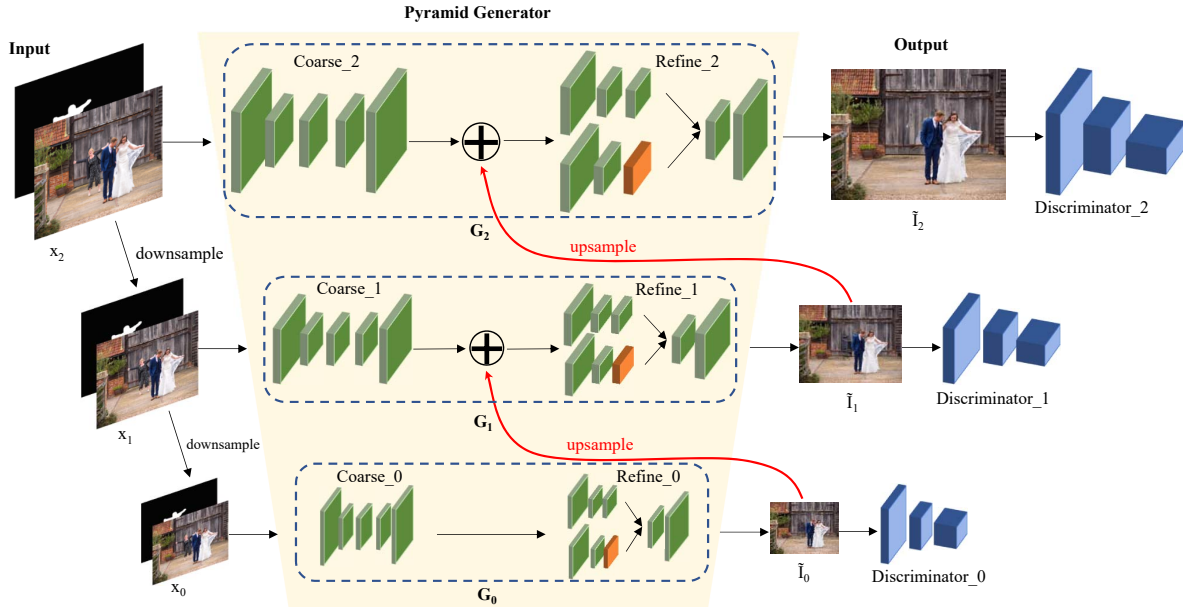
Figure 2. Architecture of our pyramid generator. The input image and mask are gradually downsampled until to the bottom layer, and then the model is trained in a bottom-up manner. The output of a lower layer will be fed to its higher layer to help refining the higher-layer's results. $\tilde{I}_2$ is regarded as the final output of our pyramid generator.

particular, the pyramid GANs has a multi-scale framework: images are gradually generated with the increasing of image scale, from bottom layer to the top one.

Multi-scale mechanism is also adopted to image inpainting task. In [26], images are gradually refined by increasing image scales, but only $L2$ content loss, VGG textual loss and TV loss are used for training. Since adversarial loss is not adopted for learning, it is still hard to generate realistic image details. Recently, a pyramid-context encoder PEN is proposed to incorporate high-level semantics with low-level pixels in [31]. PEN has only one encoder, and there is a feature-pyramid within the encoder.

On the contrary, our model is similar to SinGAN that has a stack of distinct sub-generators, where all the sub-generators are trained in an adversarial manner with independent discriminators. We argue that those distinct sub-generators could better model image structures and local details respectively.

3. Method

In this section, we first introduce the architecture of our pyramid generator including its layer fusion strategy and the adaptive dilation scheme. Next, we discuss the loss function and describe how to effectively train our pyramid generator.

3.1. Pyramid Generator

In order to fully capture the internal statistics of an image at different scales, [22] proposes a hierarchical GAN architecture to learn the distribution within each scale separately. Inspired by that, we adopt its hierarchical framework

to separate the modeling of image global structures and local details at different scales.

3.1.1 Architecture Design

Figure 2 shows the architecture of our pyramid generator which are stacked from some sub-generators. DeepFillv2 [29] is adopted as the sub-generator in this paper due to its state-of-the-art performance. At different layers, the sub-generator is trained with images of different resolutions.

Specifically, the pyramid consists of $N + 1$ sub-generators $\{G_0, ..., G_N\}$. Their inputs are $\{x_0, ..., x_N\}$, where each x_n is a concatenation of an image and a mask. In this paper, we adopt a three-layer architecture (*i.e.*, $N = 2$) since it is enough to inpaint high-resolution images.

We train the model on 512×512 images, so the input image is of 512×512 resolution. The mask is a binary image, where value of 1 represents corrupted area and 0 represents known area. As shown in Figure 2, the original training image and mask are noted as x_2 (with resolution of 512×512), which are gradually downsampled to x_1 (with resolution of 256×256) and x_0 (with resolution of 128×128), respectively.

In the pyramid, each layer has a sub-generator G_n and a corresponding discriminator D_n. The model is trained in a bottom-up manner: 1) G_0 is trained firstly by using x_0. Let $\tilde{I}_n$ indicate the recovered image of each layer. Thus, we have the output of bottom layer,

$$\tilde{I}_0 = G_0(x_0). \qquad (1)$$

2) The output $\tilde{I}_0$ is then upsampled and fed to sub-generator G_1, meanwhile x_1 is also fed to G_1. Both of them are used to train G_1. The training of G_2 is the same as G_1. As a result, we have the following outputs,

$$\tilde{I}_n = G_n(x_n, \tilde{I}_{n-1}), n > 0. \qquad (2)$$

Note that $\tilde{I}_2$ is regarded as the final output of our pyramid generator.

3.1.2 Fusion Strategy

In our pyramid generator, the output of a lower layer will be fed to a higher layer to help refine the higher-layer's results. For example, the output of G_0 will be fed to G_1 to refine the results of G_1. There are many possible fusion choices. Since there are two stages for each G_n (*i.e.*, coarse-stage and refine-stage), the output of G_0 could be fed to either the coarse-stage or the refine-stage of G_1.

On the other hand, the fusion could be conducted in feature-level or image-level. Feature-level fusion is similar to [23] that the feature maps of G_0 is upsampled and added to the corresponding feature maps of G_1. In contrast, the image-level fusion is to upsample the output image of G_0 and add to the internal produced image of G_1.

We have conducted extensive experiments to compare all those fusion options, and selected the best one according to experimental results. Specifically, we adopt the refine-stage as well as image-level fusion strategy. As shown in Figure 2, the output image of G_0 is upsampled and added with the output of coarse stage of G_1. Then their sum is fed to the refine stage of G_1. This process can be formulated as

$$\tilde{I}_n = G_n^r(G_n^c(x_n) + upsample(\tilde{I}_{n-1})), n > 0, \qquad (3)$$

where G_n^c and G_n^r indicate the coarse and refine stage of G_n respectively.

The detailed comparisons of those options are shown in Section 4.4. We guess that refine-stage fusion being better than coarse-stage fusion is due to that the high-frequency details of high-resolution image (*e.g.*, x_1) can be sufficiently exploited by the G_1^c before conducting fusion. By receiving both the output of G_1^c and G_0, the refine network G_1^r could properly leverage the good results of image details modeling (by G_1^c) and image structure modeling (by G_0).

Regarding to the feature-level fusion, we found that it is either inferior to image-level fusion or relatively difficult to stably achieve convergence. Besides, we also try another option that is to directly drop the coarse-stage G_1^c and only remain refine-stage G_1^r. Although it can reduce some computation, it is also difficult to stably get convergence.

3.1.3 Adaptive Dilation

Dilated convolutions [7] are broadly used in inpainting neural network, since it can not only explicitly adjust filter's field-of-view but also keep the resolution of feature maps. Keeping the resolution of feature maps is very important to the performance of those pixel-level algorithms such as image inpainting, and image segmentation.

On the other hand, adjusting filter's field-of-view is conducted by adjusting the *dilation rate* of dilated convolutions, which will determine the receptive field of convolutions. However, there is a conflict in modeling image global structures and image local details: modeling image structures often prefers to observe large image regions, while modeling image details prefers to observe small image patches. Therefore, it is very hard to select a proper *dilation rate* for a single-scale architecture to balance the distinct needs of modeling image structures and details.

Nevertheless, our pyramid generator has several independent sub-generators so that we could select distinct dilation rate for different sub-generators. Specifically, G_0 emphasizes modeling global structures and needs a relatively large receptive field, and hence we have 4 dilated convs with dilation rates $\{2, 4, 8, 12\}$. In contrast, G_1 and G_2 emphasize modeling image local details, and hence we only have 3 dilated convs with the dilation rates $\{2, 4, 8\}$. In this way, we can effectively solve the conflict of modeling image global structures and local details, which is called as adaptive dilation scheme in this paper.

3.2. Learning of Pyramid Generator

In this section, we will describe the loss function and the training details of our pyramid generator. Our pyramid generator is composed of several sub-generators, where each sub-generator is trained by using GAN mechanism. Particularly, for each sub-generator G_n we adopt the loss function as [29], which consists of a reconstruction term and an adversarial term,

$$\mathcal{L}_n = \mathcal{L}_{adv}(G_n, D_n) + \alpha\mathcal{L}_{re}(G_n), \qquad (4)$$

and we choose $\alpha = 1$ in our experiments.

The reconstruction term is defined as the $L1$ distance between the generated output $\tilde{I}_n$ and the ground-truth image I_n at the pixel level:

$$\mathcal{L}_{re}(G_n) = ||\tilde{I}_n - I_n||_1. \qquad (5)$$

For the adversarial term, we use the hinge loss. The loss for generator is

$$\mathcal{L}_G = -\mathbb{E}_{z \sim p_z, y \sim p_{data}} D\left(G\left(z\right), y\right), \qquad (6)$$

and for discriminator, it is

$$\mathcal{L}_D = \mathbb{E}_{x \sim p_{data(x)}} ReLU\left(\mathbb{1} - D\left(x\right)\right) \\ + \mathbb{E}_{z \sim p_{z(z)}} ReLU\left(\mathbb{1} + D\left(G\left(z\right)\right)\right). \qquad (7)$$

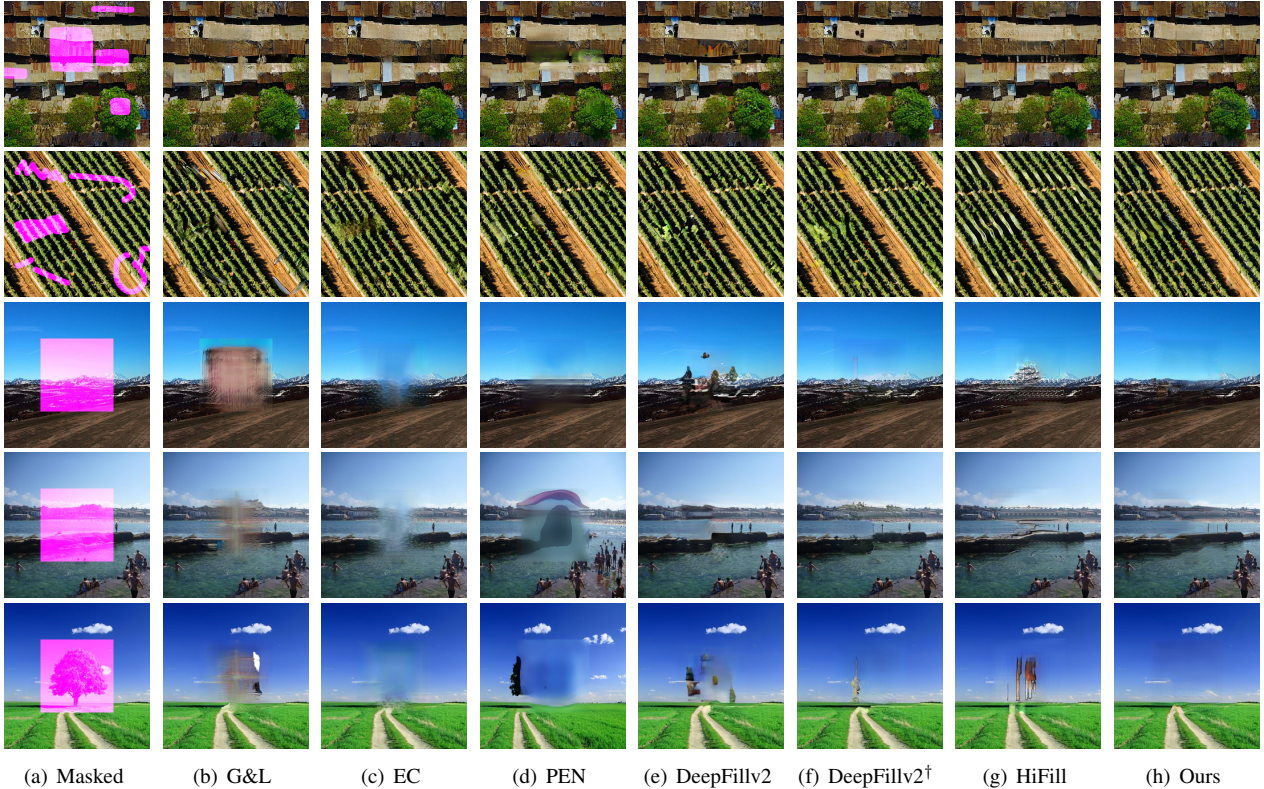| (a) Masked | (b) G&L | (c) EC | (d) PEN | (e) DeepFillv2 | (f) DeepFillv2† | (g) HiFill | (h) Ours |

Figure 3. Qualitative comparisons on Places2 val and DIV2K val set (with image resolution 512×512). † indicates methods retrained by us with 512×512 images. Best viewed by zooming-in.

Table 1. Quantitative comparisons on the Places2 val and DIV2K val set (with image resolution 512×512). Both center masks and free-form masks settings are considered. Up-arrow (↑) indicates higher score is better, while lower score is better for down-arrow (↓). † indicates that the model is trained on 512×512 images instead of 256×256 images. *: This is reported from [26] using single Titan X GPU.

Method	Places2 square mask			Places2 free-form mask			DIV2K square mask			DIV2K free-form mask			speed
	SSIM↑	PSNR↑	L1↓	SSIM↑	PSNR↑	L1↓	SSIM↑	PSNR↑	L1↓	SSIM↑	PSNR↑	L1↓	time/image
GL [7]	0.773	**20.72**	0.055	0.683	18.71	0.086	0.769	20.85	0.075	0.660	17.88	0.109	125ms
MNPS [26]	0.727	19.22	0.058	-	-	-	0.749	23.15	0.047	-	-	-	1min*
EC [17]	0.770	19.55	0.069	0.736	18.92	0.086	0.817	24.65	0.042	0.759	21.61	0.070	178ms
PEN [31]	0.754	19.36	0.056	0.724	19.72	0.068	0.792	23.42	0.036	0.758	22.40	0.053	347ms
DeepFillv2 [29]	0.769	18.86	0.056	0.737	18.59	0.073	0.814	23.47	0.043	0.765	20.91	0.064	69ms
DeepFillv2† [29]	0.769	19.40	0.053	0.740	19.41	0.067	0.827	25.72	0.036	0.789	23.40	0.053	69ms
HiFill [27]	0.749	19.87	0.051	0.701	19.14	0.074	0.791	23.63	0.040	0.736	20.67	0.073	24ms
ours	**0.777**	20.09	**0.050**	**0.746**	**19.90**	**0.064**	**0.840**	**26.74**	**0.034**	**0.796**	**24.25**	**0.050**	85ms

Taking the three-layer pyramid generator as an example, its total loss can be represented by

$$\mathcal{L}_{PG} = \lambda_0 \mathcal{L}_0 + \lambda_1 \mathcal{L}_1 + \lambda_2 \mathcal{L}_2, \qquad (8)$$

where $\mathcal{L}_0$, $\mathcal{L}_1$ and $\mathcal{L}_2$ indicate the loss of G_0, G_1, and G_2 respectively, with λ_0, λ_1, λ_2 the weights for them. In practice, we select the weight values empirically according to experimental results, i.e., we set $\lambda_0 = 10$, $\lambda_1 = 1$, and $\lambda_2 = 1$. We found that it is necessary to assign a large weight to the bottom sub-generator since the error back propagation path is relatively long for the bottom layer.

The structure of discriminator in all layers are identical to each other, as in [22]. Note that those discriminators are independently trained in our approach.

A big difference between our pyramid and SinGAN [22] is that SinGAN is trained layer by layer but all sub-generators in our pyramid are *jointly* trained. Actually, we have tried the layer-by-layer training scheme for our model, e.g., we first train G_0 and then train G_1 while fixing the parameters of G_0. Experimental results show that it is relatively difficult for G_1 to stably achieve convergence with this layer-by-layer training scheme. We will try to find out the underlying reason of such phenomena in future work.

4. Experiments

We conduct experiments on Places2 [34], CelebA-HQ [8], and DIV2K [1] to evaluate our approach. The offi-

cial train+val splits of Places2 are used to train our object-inpainting model, where each image is randomly cropped into 512×512 resolution. The 28,000 images in CelebA-HQ are used to train our face-inpainting model, where all images are resized into size 512×512 as in [10].

Although our two inpainting models are both trained on 512×512 images, they can be used to restore an image in any resolution. Besides evaluating our approach on 512×512 testing images, high-resolution testing image are also involved into the evaluation. Particularly, we choose DIV2K val as our high-resolution testing dataset, which consists of 100 images from the Internet in 2k resolution. These images have diverse contents in nature and are suitable for evaluations. Note that the images of DIV2K are randomly cropped into size 512×512 and 1024×1024 for evaluation, since it is very memory-consuming to conduct image inpainting on 2k resolution.

We adopt the original mask generation algorithm in [29], which generates center square mask plus random free-form masks for each training sample. All of our experiments are trained with TensorFlow v1.15, CUDA v10.0. The final model needs 5 days to converge on a single NVIDIA Tesla V100 GPU with batch size of 4, and we apply Adam as our optimizer. We are going to release our code in the future.

(a) Masked   (b) G&L   (c) DeepFill   (d) DeepFill†   (e) Ours

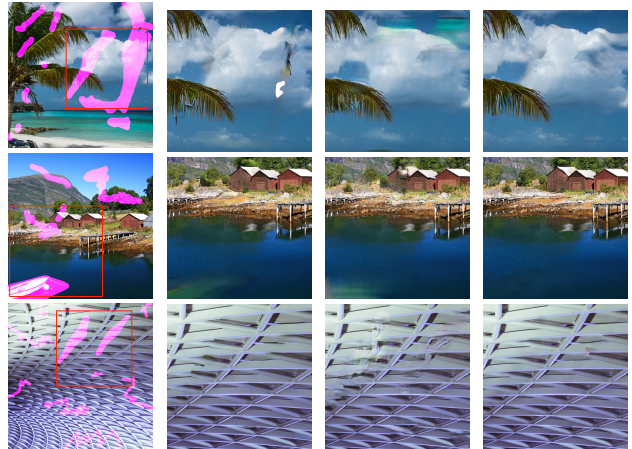Figure 4. Qualitative comparisons on CelebA val set (with image resolution 512×512).

4.1. Comparison to SOTA Methods

In this section, we compare our model with methods Global&Local (G&L) [7], MNPS [26], EdgeConnect (EC) [17], DeepFillv2 [29], PEN [31], and HiFill [27]. The original Global&Local, MNPS, EdgeConnect and PEN are trained with 256×256 images, while HiFill is trained with 512×512 images. For more comparison, we also re-trained DeepFillv2 on 512×512 images, noted as DeepFillv2†.

For numerical comparisons, we evaluate those methods on L1 loss, structural similarity index measure (SSIM) [24], and peak signal-to-noise ratio (PSNR). To calculate the in-

ference time per image, we test all the images in Places2 validation set with center masks on single NVIDIA GTX 2080Ti GPU.

Comparisons on 512×512 images Table 1 is the comparison of those methods on the datasets Places2 and DIV2K (with images in 512×512 resolution). We could find that our approach outperforms all the other methods on both the free-form mask inpainting and the center mask inpainting.

(a) Masked   (b) DeepFillv2†   (c) HiFill   (d) Ours

Figure 5. Qualitative comparisons on DIV2K val set (with image resolution 1024×1024).

Table 2. Quantitative comparisons on 1024×1024 images from DIV2K val set. Both center masks and free-form masks setting are considered.

Method	square mask			free-form mask		
	SSIM↑	PSNR↑	L1↓	SSIM↑	PSNR↑	L1↓
DeepFill† [29]	0.782	19.83	0.052	0.734	19.16	0.072
HiFill [27]	0.765	19.58	0.053	0.700	18.43	0.079
ours	**0.788**	**20.37**	**0.049**	0.734	**19.36**	**0.069**

Figure 3 shows the visual comparison for examples of resolution 512×512 from dataset Places2 and DIV2K. From the results, we could find Global&Local and Edge-Connect prone to producing color inconsistent and blurry content. The original DeepFillv2 could better recover content on small holes, but it tends to produce artifacts when inpainting large holes. Moreover, PEN is not good at generating realistic high-frequency details. Probably because they are trained on 256×256 images, they cannot properly handle 512×512 inpainting task.

On the other hand, DeepFillv2†, HiFill and our model are all trained on 512×512 images, and there is an obvious performance improvement. However, the restored content from DeepFillv2† is not very coherent to their surrounding pixels, and HiFIll is not good at recovering image global structures. In contrast, our approach can not only recover image global structures but also produce coherent image de-

tails.

The results on CelebA-HQ are shown in Figure 4. Compared to the vanilla DeepFillv2, DeepFillv2† could properly synthesize some face parts (*e.g.*, eyes). Nevertheless, it still cannot model the global structures of a face.

Comparisons on 1024×1024 images Table 2 illustrates the comparison of high-resolution image inpainting (with images resolution 1024×1024). On both free-form mask and center mask, our approach outperforms DeepFillv2† and HiFill.
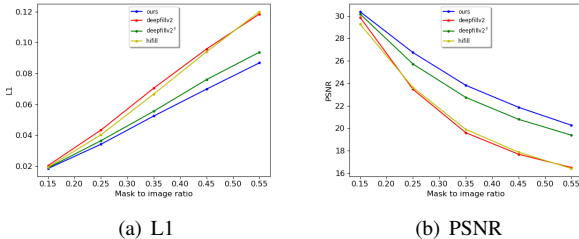

(a) L1   (b) PSNR

Figure 6. The performance degradation with respect to the increasing of hole size. With the ratio of hole size increased from 15% to 55%, the degradation magnitude of our model is much less than that of DeepFillv2 [29]. All the experiments are on DIV2K val set with square shape masks.

The results of these methods on high-resolution image inpainting are shown in Figure 5. Obviously, it is much more challenging to generate coherent and realistic image details on 1024×1024 images. For example, DeepFillv2† tends to generate incorrect semantic content to fill in the hole (e.g., the missing 'cloud' region are filled with 'tree' pixels), and there is visible color discrepancy for HiFill inpainting results. At the same time, our approach's inpainting results look more relevant and realistic.

4.2. Large-hole Inpainting

It is known that when the corrupted area is large the image inpainting becomes much more challenging. As shown in Figure 3, when the hole size becomes large, the center of the hole tends to have color discrepancy and blurriness.

Recent work found that such issues could be alleviated by adopting the progressive learning strategy [9], without the needs of modifying the generative model itself. The progressive learning strategy is to divide the training procedure into several stages so that the small-hole training data are first used to train the model at early stages whereas large-hole training data will be used to fine-tune the model at later stages. Even if it is effective sometimes, such strategy still has some limitations. For example, it is hard to decide when to move from one stage to the next stage.

As aforementioned, we have illustrated that our pyramid generator could not only align to the progressive learning strategy but also avoid the need for explicitly separating the training procedure into several stages. In this section, we

will evaluate our method by gradually increasing the hole size and show its advantage on large hole inpainting.

Specifically, the center mask setting is adopted in this experiment, since it is easy to control the ratio of hole size to the image size. We gradually increase such ratio from 15% to 55% with interval of 10%. As shown in Figure 6, the red line is the performance of DeepFillv2 [29] and the blue line indicates our model performance. There is a clear performance degradation for both methods when the ratio increases, but the degradation magnitude of our model is much less than that of DeepFillv2.
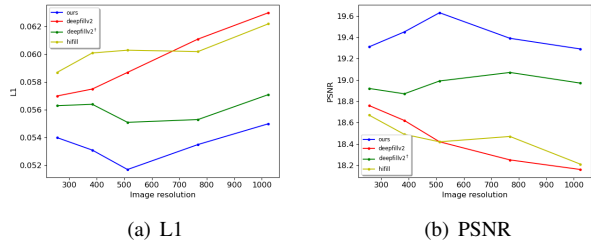

(a) L1   (b) PSNR

Figure 7. The performance degradation with respect to the increasing of image resolution. With the image resolution increased from 256 to 1024, our model could better cope against the performance degradation, and hence our model is more suitable for high-resolution image inpainting. Experiments are conduct on DIV2K val set with center square masks.

Besides, it is obvious that increasing resolution of training images also benefits large-hole inpainting task (*e.g.*, DeepFillv2† outperforms DeepFillv2), but it is still inferior to our approach.

4.3. High-resolution Image Inpainting

With the rapid development of high quality camera devices, the images we meet in daily life are often high-resolution. Thus, the real-world inpainting task is often about high-resolution image inpainting.

On the other hand, many previous works have demonstrated that increasing the resolution of training images is critical to the inpainting performance, especially for the high-resolution image inpainting task [31, 27].

In this section, we conduct some experiments to study such observations. Specifically, we train the same model DeepFillv2 [29] with images of different resolution: the original DeepFillv2 is trained with images of 256×256 resolution, while the model DeepFillv2† is trained with images of 512×512 resolution. From Figure 7, we can see that the inpainting performance could be improved by increasing the resolution of training images, which validates the conclusion of previous work.

Besides, Figure 7 tells us that the inpainting performance will decrease when the resolution of testing image is increasing. This indicates that the high-resolution image inpainting task is much more difficult than low-resolution in-

painting task. Moreover, it is worth noting that such performance degradation is different between the model DeepFillv2 and DeepFillv2[†]: the DeepFillv2[†] could better cope against the performance degradation than the DeepFillv2, *i.e.,* the green line drops slower than the red line according to the metric PSNR. Therefore, it is critical to use high-resolution images as training data.

More importantly, we also evaluate our pyramid generator (noted as blue line in Figure 7). Compared to the model DeepFillv2[†] and HiFill [27], we can see that our model outperforms those methods, although they are all trained with images of same resolution (*i.e.,* 512×512). It indicates that our model could fully exploit the benefits of learning with high-resolution images, and hence our model is good at high-resolution image inpainting.

4.4. Ablation Study

Except the recent architecture of our pyramid generator described in Section 3, there are many other design options. We have evaluated those design options and will discuss their performance in this section. Such evaluation are conducted on DIV2K dataset under the same settings as the previous experiments.

Table 3. Comparisons between our three-layer model and the two-layer variants.

	SSIM↑	PSNR↑	L1↓
$PG_{128+256}$	0.809	24.07	0.042
$PG_{256+512}$	0.827	26.03	0.035
$PG_{128+256+512}$ (Ours)	**0.840**	**26.74**	**0.034**

(a) Masked   (b) $PG_{128+256}$

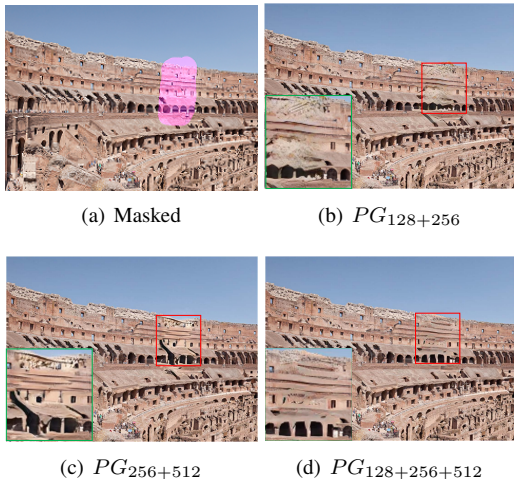(c) $PG_{256+512}$ (d) $PG_{128+256+512}$

Figure 8. Qualitative comparisons between our three-layer model and the two-layer variants.

How many layers should we have? Since our pyramid generator has a multi-layer architecture where each layer corresponds to a specific image scale, we could have many choices for the number of layers. Apart from our current implementation of 3 layers, we also try two variants of two-layer architecture: one variant is the combination of the 128-layer and 256-layer (noted as $PG_{128+256}$), the other variant is the combination of 256-layer and 512-layer (noted as $PG_{256+512}$).

The comparisons of our recent three-layer model and the two variants are illustrated in Table 3 and Figure 8. It shows that the two variants have distinct drawbacks. The model $PG_{128+256}$ tends to generate blurry content due to its disadvantage on modeling image high-frequency details. On the other hand, the model $PG_{256+512}$ tends to generate incoherent image structures because of the lacking of image global information. In contrast, our three-layer model $PG_{128+256+512}$ enjoys the advantages of the two variants, and could generate both coherent image structures and realistic image details.

How to fuse different layers? As mentioned above, there are many possible fusion options, and we compare them in this section. First, we consider feature-level fusion, which can be further categorized as coarse-stage fusion (*i.e.,* fuse with G_1^c) and refine-stage fusion (*i.e.,* fuse with G_1^r). Next, we consider image-level fusion. Besides the fusion scheme adopted in our model, there are additional two options. Since there are two path for the G_1^r (*i.e.,* attention path and non-attention path), we can feed the output of G_0 only to one path. From Table 4, we can see that feature-level refine-stage fusion cannot converge, and our fusion scheme is the best among them.

Table 4. Quantitative comparisons on different fusion strategies.

	feature-level		image-level		
	coarse	refine	att. path	non-att path	ours
SSIM↑	0.786	-	0.825	0.833	**0.840**
PSNR↑	23.16	-	25.94	25.96	**26.74**
L1↓	0.044	-	0.036	0.036	**0.034**

Is adaptive dilation effective? Due to the pyramid structure, our approach could adopt the *adaptive dilation* mechanism, which means the sub-generators at different layers could have different number of dilation layers, and different dilation rate configurations. In this section, we use the baseline that all sub-generators have same number of dilation layers and same dilation rate configurations. In particular, it has 4 dilation layers with dilation rates $\{2, 4, 8, 16\}$, which is the same as [29]. In contrast, in our approach G_0 has 4 dilation layers with dilation rates $\{2, 4, 8, 12\}$, and both G_1 and G_2 have 3 dilation layers with dilation rates $\{2, 4, 8\}$. As shown in Table 5, we find that our adaptive dilation mechanism benefits the performance improvement compared to the baseline methods.

Table 5. Quantitative comparisons between the standard and our adaptive dilation scheme.

	SSIM↑	PSNR↑	L1↓
standard dilation	0.829	26.38	0.035
adaptive dilation (Ours)	**0.840**	**26.74**	**0.034**

5. Conclusion

This paper proposes a 'structure first detail next' workflow for image inpainting. In particular, we introduce a Pyramid Generator by stacking several sub-generators, where image global structures and local details could be better separately modeled at different pyramid layers. Notice that our approach has a progressive learning scheme which allows it to restore images with large masks. In addition, our model is suitable for inpainting high-resolution images. Extensive experiments show that our approach outperforms the other state-of-the-art methods.

References

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

[2] Coloma Ballester, Marcelo Bertalmio, Vicent Caselles, Guillermo Sapiro, and Joan Verdera. Filling-in by joint interpolation of vector fields and gray levels. *IEEE transactions on image processing*, 10(8):1200–1211, 2001. 2

[3] Connelly Barnes, Eli Shechtman, Adam Finkelstein, and Dan B Goldman. Patchmatch: A randomized correspondence algorithm for structural image editing. *ACM Trans. Graph.*, 28(3):24, 2009. 2

[4] B. Edwards. Drawing on the right side of the brain: The definitive, 4th edition. In *Penguin Publishing Group*, 2012. 1

[5] David Eigen and Rob Fergus. Predicting depth, surface normals and semantic labels with a common multi-scale convolutional architecture. In *Proceedings of the IEEE international conference on computer vision*, pages 2650–2658, 2015. 2

[6] M. Eitz, J. Hays, and M. Alexa. How do humans sketch objects? In *ACM Transactions on graphics (TOG)*, 2012. 1

[7] Satoshi Iizuka, Edgar Simo-Serra, and Hiroshi Ishikawa. Globally and locally consistent image completion. *ACM Transactions on Graphics (ToG)*, 36(4):1–14, 2017. 2, 4, 5, 6

[8] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. In *Proceedings of International Conference on Learning Representations (ICLR) 2018*, 2018. International Conference on Learning Representations, ICLR ; Conference date: 30-04-2018 Through 03-05-2018. 5

[9] Soo Ye Kim, Kfir Aberman, Munchurl Kim, and Orly Liba. Zoom-to-inpaint: Image inpainting with high frequency details. In *arXiv: 2012.09401*, 2020. 1, 2, 7

[10] Cheng-Han Lee, Ziwei Liu, Lingyun Wu, and Ping Luo. Maskgan: Towards diverse and interactive facial image manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5549–5558, 2020. 6

[11] Jingyuan Li, Ning Wang, Lefei Zhang, Bo Du, and Dacheng Tao. Recurrent feature reasoning for image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7760–7768, 2020. 1

[12] Yijun Li, Sifei Liu, Jimei Yang, and Ming-Hsuan Yang. Generative face completion. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3911–3919, 2017. 1

[13] Liang Liao, Jing Xiao, Zheng Wang, Chia-Wen Lin, and Shin'ichi Satoh. Guidance and evaluation: Semantic-aware image inpainting for mixed scenes. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 683–700, Cham, 2020. Springer International Publishing. 1

[14] 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 *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 85–100, 2018. 2

[15] Hongyu Liu, Bin Jiang, Yibing Song, Wei Huang, and Chao Yang. Correction to: Rethinking image inpainting via a mutual encoder-decoder with feature equalizations. In Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages C1–C1, Cham, 2020. Springer International Publishing. 1

[16] Hongyu Liu, Bin Jiang, Yi Xiao, and Chao Yang. Coherent semantic attention for image inpainting. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4170–4179, 2019. 1

[17] Kamyar Nazeri, Eric Ng, Tony Joseph, Faisal Qureshi, and Mehran Ebrahimi. Edgeconnect: Structure guided image inpainting using edge prediction. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019. 1, 5, 6

[18] Deepak Pathak, Philipp Krahenbuhl, Jeff Donahue, Trevor Darrell, and Alexei A Efros. Context encoders: Feature learning by inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2536–2544, 2016. 2

[19] 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*, 2020. 1

[20] Wenqi Ren, Si Liu, Hua Zhang, Jinshan Pan, Xiaochun Cao, and Ming-Hsuan Yang. Single image dehazing via multi-scale convolutional neural networks. In *European conference on computer vision*, pages 154–169. Springer, 2016. 2

[21] Yurui Ren, Xiaoming Yu, Ruonan Zhang, Thomas H Li, Shan Liu, and Ge Li. Structureflow: Image inpainting via structure-aware appearance flow. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 181–190, 2019. 1

[22] Tamar Rott Shaham, Tali Dekel, and Tomer Michaeli. Singan: Learning a generative model from a single natural image. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4570–4580, 2019. 2, 3, 5

[23] Ting-Chun Wang, Ming-Yu Liu, Jun-Yan Zhu, Andrew Tao, Jan Kautz, and Bryan Catanzaro. High-resolution image synthesis and semantic manipulation with conditional gans. In

Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8798–8807, 2018. 2, 4

[24] 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

[25] Zhaoyi Yan, Xiaoming Li, Mu Li, Wangmeng Zuo, and Shiguang Shan. Shift-net: Image inpainting via deep feature rearrangement. In *Proceedings of the European conference on computer vision (ECCV)*, pages 1–17, 2018. 1

[26] Chao Yang, Xin Lu, Zhe Lin, Eli Shechtmanz, Oliver Wang, and Hao Li. High-resolution image inpainting using multi-scale neural patch synthesis. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2017. 1, 3, 5, 6

[27] Zili Yi, Qiang Tang, Shekoofeh Azizi, Daesik Jang, and Zhan Xu. Contextual residual aggregation for ultra high-resolution image inpainting. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2020. 2, 5, 6, 7, 8

[28] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Generative image inpainting with contextual attention. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5505–5514, 2018. 1

[29] Jiahui Yu, Zhe Lin, Jimei Yang, Xiaohui Shen, Xin Lu, and Thomas S Huang. Free-form image inpainting with gated convolution. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4471–4480, 2019. 1, 2, 3, 4, 5, 6, 7, 8

[30] Tao Yu, Zongyu Guo, Xin Jin, Shilin Wu, Zhibo Chen, Weiping Li, Zhizheng Zhang, and Sen Liu. Region normalization for image inpainting. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 12733–12740, 2020. 1

[31] Yanhong Zeng, Jianlong Fu, Hongyang Chao, and Baining Guo. Learning pyramid-context encoder network for high-quality image inpainting. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1486–1494, 2019. 2, 3, 5, 6, 7

[32] 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 Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm, editors, *Computer Vision – ECCV 2020*, pages 1–17, Cham, 2020. Springer International Publishing. 1

[33] Chuanxia Zheng, Tat-Jen Cham, and Jianfei Cai. Pluralistic image completion. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1438–1447, 2019. 1

[34] 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