PixelFolder: An Efficient Progressive Pixel Synthesis Network for Image Generation

Jing He[1], Yiyi Zhou[1⋆], Qi Zhang[2], Jun Peng[1], Yunhang Shen[2], Xiaoshuai Sun[1,3], Chao Chen[2], and Rongrong Ji[1,3]

[1]Media Analytics and Computing Lab, Department of Artificial Intelligence, School of Informatics, Xiamen University. [2]Youtu Lab, Tencent. [3]Institute of Artificial Intelligence, Xiamen University.
{blinghe, pengjun}@stu.xmu.edu.cn, {zhouyiyi, xssun, rrji}@xmu.edu.cn, {merazhang, aaronccchen}@tencent.com, shenyunhang01@gmail.com

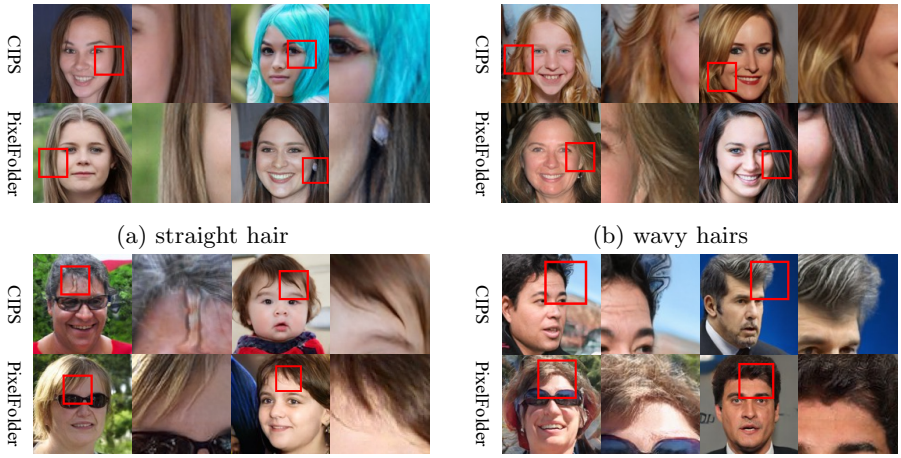
Abstract. Pixel synthesis is a promising research paradigm for image generation, which can well exploit pixel-wise prior knowledge for generation. However, existing methods still suffer from excessive memory footprint and computation overhead. In this paper, we propose a progressive pixel synthesis network towards efficient image generation, coined as *PixelFolder*. Specifically, PixelFolder formulates image generation as a progressive pixel regression problem and synthesizes images by a multi-stage paradigm, which can greatly reduce the overhead caused by large tensor transformations. In addition, we introduce novel *pixel folding* operations to further improve model efficiency while maintaining pixel-wise prior knowledge for end-to-end regression. With these innovative designs, we greatly reduce the expenditure of pixel synthesis, *e.g.*, reducing 89% computation and 53% parameters compared to the latest pixel synthesis method called *CIPS*. To validate our approach, we conduct extensive experiments on two benchmark datasets, namely FFHQ and LSUN Church. The experimental results show that with much less expenditure, PixelFolder obtains new state-of-the-art (SOTA) performance on two benchmark datasets, *i.e.*, 3.77 *FID* and 2.45 *FID* on FFHQ and LSUN Church, respectively. Meanwhile, PixelFolder is also more efficient than the SOTA methods like *StyleGAN2*, reducing about 72% computation and 31% parameters, respectively. These results greatly validate the effectiveness of the proposed PixelFolder. Our source code is available at https://github.com/BlingHe/PixelFolder.

Keywords: Pixel Synthesis; Image Generation; Pixel Folding

1 Introduction

As an important task of computer vision, image generation has made remarkable progress in recent years, which is supported by a flurry of generative adversarial

⋆ Corresponding Author.



(a) straight hair (b) wavy hairs

(c) bangs (d) short hairs

Fig. 1: Comparison of the generated faces by CIPS [2] and PixelFolder on FFHQ. Compared with CIPS, PixelFolder synthesizes more vivid faces and can also alleviate local incongruities via its novel network structure.

networks [4,5,7,9,15,18,19,20,25,42]. One of the milestone works is the StyleGAN series [19,20], which borrows the principle of style transfer [14] to build an effective generator architecture. Due to the superior performance in image quality, this style-driven modeling has become the mainstream paradigm of image generation [19,20], which also greatly influences and promotes the development of other generative tasks, such as image manipulation [8,21,49,51,55], image-to-image translation [6,16,17,27,36,54] and text-to-image generation [26,39,41,50].

In addition to the StyleGAN series, pixel synthesis [2,45] is another paradigm of great potential for image generation. Recently, Anokin *et al.* [2] propose a novel Conditionally-Independent Pixel Synthesis (CIPS) network for adversarial image generation, which directly computes each pixel value based on the random latent vector and positional embeddings. This end-to-end pixel regression strategy can well exploit pixel-wise prior knowledge to facilitate the generation of high-quality images. Meanwhile, it also simplifies the design of generator architecture, *e.g.*, only using 1×1 convolutions, and has a higher generation ability with non-trivial topologies [2]. On multiple benchmarks [19,42], this method exhibits comparable performance against the StyleGAN series, showing a great potential in image generation. In this paper, we also follow the principle of pixel synthesis to build an effective image generation network.

Despite the aforementioned merits, CIPS still has obvious shortcomings in model efficiency. Firstly, although CIPS is built with a simple network structure, it still requires excessive memory footprint and computation during inference. Specifically, this is mainly attributed to its high-resolution pixel tensors for end-to-end pixel regression, *e.g.*, $256 \times 256 \times 512$, which results in a large compu-

tational overhead and memory footprint, as shown in Fig. 2a. Meanwhile, the learnable coordinate embeddings also constitute a large number of parameters, making CIPS taking about 30% more parameters than StyleGAN2 [20]. These issues greatly limit the applications of CIPS in high-resolution image synthesis.

To address these issues, we propose a novel progressive pixel synthesis network towards efficient image generation, termed *PixelFolder*, of which structure is illustrated in Fig. 2b. Firstly, we transform the pixel synthesis problem to a progressive one and then compute pixel values via a multi-stage structure. In this way, the generator can process the pixel tensors of varying scales instead of the fixed high-resolution ones, thereby reducing memory footprint and computation greatly. Secondly, we introduce novel *pixel folding* operations to further improve model efficiency. In PixelFolder, the large pixel tensors of different stages are folded into the smaller ones, and then gradually unfolded (expanded) during feature transformations. These pixel folding (and unfolding) operations can well preserve the independence of each pixel, while saving model expenditure. These innovative designs help PixelFolder achieves high-quality image generations with superior model efficiency, which are also shown to be effective for *local imaging incongruity* found in CIPS [2], as shown in Fig. 1.

To validate the proposed PixelFolder, we conduct extensive experiments on two benchmark datasets of image generation, *i.e.*, FFHQ [19] and LSUN Church [42]. The experimental results show that PixelFolder not only outperforms CIPS in terms of image quality on both benchmarks, but also reduces parameters and computation by 53% and 89%, respectively. Compared to the state-of-the-art model, *i.e.*, StyleGAN2 [20], PixelFolder is also very competitive and obtains new SOTA performance on FFHQ and LSUN Church, *i.e.*, 3.77 FID and 2.45 FID, respectively. Meanwhile, the efficiency of PixelFolder is still superior, with 31% less parameters and 72% less computation than StyleGAN2.

To sum up, our contribution is two-fold:

1. We propose a progressive pixel synthesis network for efficient image generation, termed *PixelFolder*. With the multi-stage structure and innovative pixel folding operations, PixelFolder greatly reduces the computational and memory overhead while keeping the property of end-to-end pixel synthesis.
2. Retaining much higher efficiency, the proposed PixelFolder not only has better performance than the latest pixel synthesis method CIPS, but also achieves new SOTA performance on FFHQ and LSUN Church.

2 Related Work

Recent years have witnessed the rapid development of image generation supported by a bunch of generative adversarial network (GAN) [9] based methods [1,28,30,33,11,38,40,46,48]. Compared with previous approaches [23,47], GAN-based methods model the domain-specific data distributions better through the specific adversarial training paradigm, *i.e.*, a discriminator is trained to distinguish whether the images are true or false for the optimization of the generator. To further improve the quality of generations, a flurry of methods [7,42,5,3,10]

have made great improvements in both GAN structures and objective functions. Recent advances also resort to a progressive structure for high-resolution image generation. PGGAN [18] proposes a progressive network to generate high-resolution images, where both generator and discriminator start their training with low-resolution images and gradually increase the model depth by adding-up the new layers during training. StyleGAN series [19,20] further borrow the concept of "*style*" into the image generation and achieve remarkable progress. The common characteristic of these progressive methods is to increase the resolution of hidden features by up-sampling or deconvolution operations. Differing from these methods, our progressive modeling is based on the principle of pixel synthesis with pixel-wise independence for end-to-end regression.

In addition to being controlled by noise alone, some methods exploit coordinate information for image generation. CoordConv-GAN [32] introduces pixel coordinates in every convolution based on DCGAN [42], which proves that pixel coordinates can better establish geometric correlations between the generated pixels. COCO-GAN [29] divides the image into multiple patches with different coordinates, which are further synthesized independently. CIPS [2] builds a new paradigm of using coordinates for image generation, *i.e.*, pixel regression, which initializes the prior matrix based on pixel coordinates and deploys multiple 1×1 convolutions for pixel transformation. This approach not only greatly simplifies the structure of generator, but also achieves competitive performance against existing methods. In this paper, we also follow the principle of pixel regression to build the proposed PixelFolder.

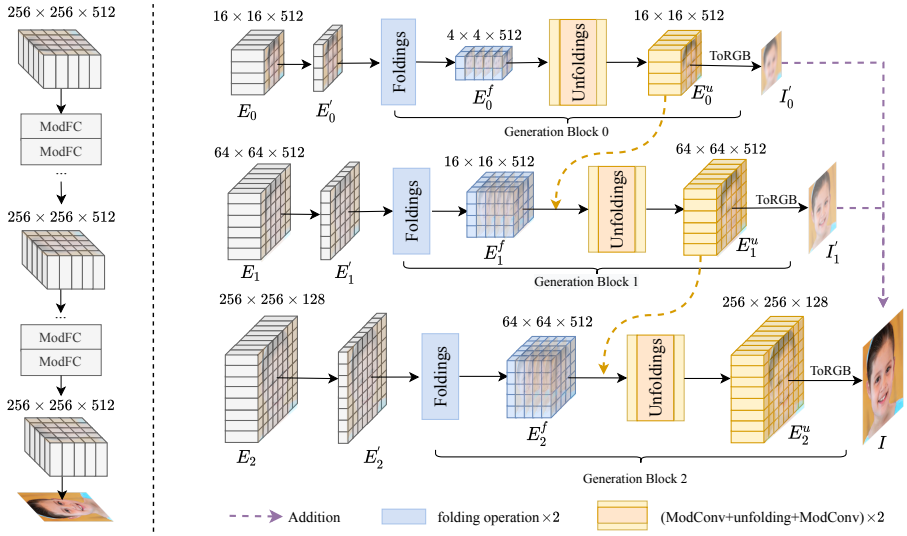
Our work is also similar to a recently proposed method called INR-GAN [45], which also adopts a multi-stage structure. In addition to the obvious differences in network designs and settings, PixelFolder is also different from INR-GAN in the process of pixel synthesis. In INR-GAN, the embeddings of pixels are gradually up-sampled via *nearest neighbor interpolation*, which is more in line with the progressive models like StyleGAN2 [20] or PGGAN [18]. In contrast, PixelFolder can well maintain the independence of each pixel during multi-stage generation, and preserve the property of end-to-end pixel regression via pixel folding operations.

3 Preliminary

Conditionally-Independent Pixel Synthesis (CIPS) is a novel generative adversarial network proposed by Anokhin *et al.* [2]. Its main principle is to synthesis each pixel conditioned on a random vector $z \in Z$ and the pixel coordinates (x, y), which can be defined by

$$I = \{G(x, y; \mathbf{z}) | (x, y) \in mgrid(H, W)\}, \tag{1}$$

where $mgrid(H, W) = \{(x, y) | 0 \leq x \leq W, 0 \leq y \leq H\}$ is the set of integer pixel coordinates, and $G(\cdot)$ is the generator. Similar to StyleGAN2 [20], z is turned into a style vector w via a mapping network and then shared by all pixels. Afterwards, w is injected into the generation process via ModFC layers [2].



(a) CIPS (b) PixelFolder (Ours)

Fig. 2: A comparison of the architectures of CIPS [2] (left) and the proposed PixelFolder (right). PixelFolder follows the pixel synthesis principle of CIPS, but regards image generation as a multi-stage regression problem, thereby reducing the cost of large tensor transformations. Meanwhile, novel *pixel folding* operations are also applied in PixelFodler to further improve model efficiency.

An important design in CIPS is the positional embeddings of synthesized pixels, which are consisted of Fourier features and coordinate embeddings. The Fourier feature of each pixel $e_{fo}(x, y) \in \mathbb{R}^d$ is computed based on the coordinate (x, y) and transformed by a learnable weight matrix $B_{fo} \in \mathbb{R}^{2\times d}$ and sin activation. To improve model capacity, Anokhin *et al.* also adopt the coordinate embedding $e_{co}(x, y) \in \mathbb{R}^d$, which has $H \times W$ learnable vectors in total. Afterwards, the final pixel vector $e(x, y) \in \mathbb{R}^d$ is initialized by concatenating these two types of embeddings and then fed to the generator.

Although CIPS has a simple structure and can be processed in parallel [2], its computational cost and memory footprint are still expensive, mainly due to the high-resolution pixel tensor for end-to-end generation. In this paper, we follow the principle of CIPS defined in Eq. 1 to build our model and address the issue of model efficiency via a progressive regression paradigm.

4 PixelFolder

4.1 Overview

The structure of the proposed PixelFodler is illustrated in Fig.2. To reduce the high expenditure caused by end-to-end regression for large pixel tensors, we first

transform pixel synthesis to a multi-stage generation problem, which can be formulated as

$$I = \sum_{i=0}^{K-1} \{G_i(x_i, y_i; \mathbf{z}) | (x_i, y_i) \in mgrid(H_i, W_i)\}, \tag{2}$$

where i denotes the index of generation stages. At each stage, we initialize a pixel tensor $\mathbf{E}_i \in \mathbb{R}^{H_i \times W_i \times d}$ for generation. The RGB tensors $I'_i \in \mathbb{R}^{H_i \times W_i \times 3}$ predicted by different stages are then aggregated for the final pixel regression. This progressive paradigm can avoid the constant use of large pixel tensors to reduce excessive memory footprint. In literature [18,45,52,53], it is also shown effective to reduce the difficulty of image generation.

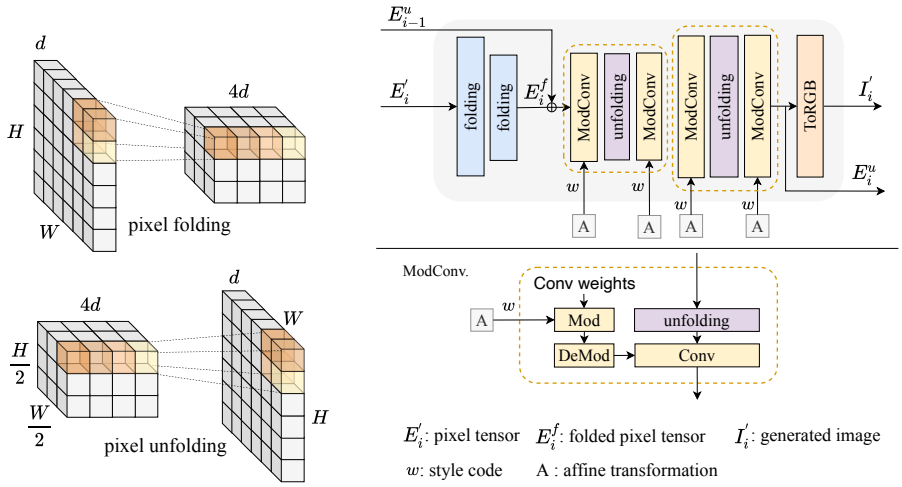
To further reduce the expenditure of each generation stage, we introduce novel *pixel folding* operations to PixelFolder. As shown in Fig.2, the large pixel tensor is first projected onto a lower-dimension space, and their local pixels, *e.g.*, in 2×2 patch, are then concatenated to form a new tensor with a smaller resolution, denoted as $\mathbf{E}_i^f \in \mathbb{R}^{\frac{H_i}{k} \times \frac{W_i}{k} \times d}$, where k is the scale of folding. After passing through the convolution layers, the pixel tensor is decomposed again (truncated from the feature dimension), and combined back to the original resolution. We term these parameter-free operations as *pixel folding* (and unfolding). Folding features is not uncommon in computer vision, which is often used as an alternative to the operations like *down-sampling* or *pooling* [31,34,35,44]. But in PixelFolder, it not only acts to reduce the tensor resolution, but also serves to maintain the independence of folded pixels.

To maximize the use of pixel-wise prior knowledge at different scales, we further combine the folded tensor E_i^f with the unfolded pixel tensor E_{i-1}^u of the previous stage, as shown in Fig. 2b. With the aforementioned designs, PixelFolder can significantly reduce memory footprint and computation, while maintaining the property of pixel synthesis.

4.2 Pixel folding

The illustration of *pixel folding* is depicted in Fig. 3a, which consists of two operations, namely *folding* and *unfolding*. The folding operation spatially decomposes the pixel tensor into multiple local patches, and straighten each of the patches to form a smaller but deeper tensor. On the contrary, the unfolding operation will truncate the folded pixel vectors from the feature dimension to recover the tensor resolution.

Particularly, pixel folding can effectively keep the independence and spatial information of each pixel regardless of the varying resolutions of the hidden tensors. This also enables the pixel-wise prior knowledge to be fully exploited for image generation. In addition, when the pixels are folded, they can receive more interactions via convolutions, which is found to be effective for the issue of *local imagery incongruity* caused by insufficient local modeling [2].



(a) pixel foldings (b) generation block

Fig. 3: (a) The illustrations of pixel folding and unfolding operations. These parameter-free operations can maintain the pixel-wise independence when changing the tensor resolution. (b) The detailed structure of the generation block in PixelFolder. The number of parameterized layers in PixelFolder is much smaller than those of CIPS and StyleGAN2.

4.3 Pixel tensor initialization

Similar to CIPS [2], we also apply Fourier features and coordinate embeddings to initialize the pixel tensors. Specifically, given the coordinate of a pixel (x, y), Fourier feature $e_{fo}(x, y)$ is obtained by

$$e_{fo}(x, y) = \sin\left[B_{fo}(x', y')^T\right], \tag{3}$$

where $x' = \frac{2x}{W_i - 1} - 1$ and $y' = \frac{2y}{H_i - 1} - 1$, and $B_{fo} \in \mathbb{R}^{2\times d}$ is the projection weight matrix. The coordinate embedding is a parameterized vector, denoted as $e_{co}(x, y) \in \mathbb{R}^d$. Afterwards, these two types of embeddings are concatenated and projected to obtain the new pixel tensor, denoted as $\mathbf{E}_i \in \mathbb{R}^{H_i \times W_i \times d}$.

In principle, Fourier features serve to preserve the spatial information and capture the relationships between pixels [2,32]. The learnable coordinate embeddings can increase model capacity to improve image quality, *e.g.*, to avoid wave-like artifacts [2]. In PixelFolder, we only apply coordinate embeddings to the first generation stage to keep model compactness, and we found this trade-off has little detriment to image quality during experiments.

4.4 Generation blocks

The detailed structure of generation blocks in PixelFolder is given in Fig. 3b. After folding operations, a modulated convolution (*ModConv*) layer [20] is de-

ployed for feature transformation. Then unfolding operations are used to recover the resolution, each followed by another ModConv layer. In practice, we use two folding and unfolding operations to gradually reduce and recover the tensor resolution, respectively, which is to avoid the drastic change of tensor resolution during feature transformation. The convolution filter is set to 3×3, considering the issue of local imaging incongruity. Besides, we also carefully set the resolution and folded pixels of each generation stage to ensure that the output tensor of current stage can be integrated into the next stage. Similar to StyleGAN2 [20], the style vector w is injected into the ModConv layers via modulating their convolution filter, *i.e.*, being mapped to scale vector s with an affine network. Finally, the recovered pixel tensors are linearly projected onto RGB space as the output of each stage, which are then aggregated for the final regression. Due to our efficient modeling strategy, PixelFolder uses only 12 convolution layers in all generation stages, thus having much fewer parameters than the SOTA methods like StyleGAN2 [20] and CIPS [2].

5 Experiments

To validate the proposed PixelFolder, we conduct extensive experiments on two benchmark datasets[1], namely Flickr Faces-HQ [19] and LSUN Church [42], and compare it with a set of state-of-the-art (SOTA) methods including CIPS [2], StyleGAN2 [20] and INR-GAN [45].

5.1 Datasets

Flickr Faces-HQ (FFHQ) [19] consistes of $70,000$ high-quality human face images, which all have a resolution of 1024×1024. The images were crawled from Flickr and automatically aligned and cropped.
LSUN Church is the sub-dataset of Large-scale Scene UNderstanding(LSUN) benchmark [42]. It contains about $126,000$ images of churches in various architectural styles, which are collected from natural surroundings.

5.2 Metrics

To validate the proposed PixelFolder, we conduct evaluations from the aspects of image quality and model efficiency, respectively. The metrics used for image quality include *Fréchet Inception Distance* (FID) [12] and *Precision and Recall* (P&R) [24,43]. FID measures the distance between the real images and the generated ones from the perspective of mean and covariance matrix. P&R evaluates the ability of fitting the true data distribution. Specifically, for each method, we randomly generate $50,000$ images for evaluation. In terms of model efficiency, we adopt the number of parameters (#Params), *Giga Multiply Accumulate Operations* (GMACs) [13], and generation speed (im/s) to measure model compactness, computation overhead and model inference, respectively.

[1] More experiments on other datasets and high-resolution are available in the supplementary material.

	#Parm (M) ↓	GMACs ↓	Speed (im/s) ↑
INR-GAN [45]	107.03	38.76	**84.55**
CIPS [2]	44.32	223.36	11.005
StyleGAN2 [20]	30.03	83.77	44.133
PixelFolder (ours)	**20.84**	**23.78**	77.735

Table 1: Comparison between PixelFolder, StyleGAN2, CIPS and INR-GAN in terms of parameter size (#Params), computation overhead (GMACs) and inference speed. Here, "M" denotes millions, and "im/s" is image per-second. ↑ denotes that lower is better, while ↓ is *vice verse*. PixelFolder is much superior than other methods in both model compactness and efficiency, which well validates its innovative designs.

5.3 Implementation

In terms of the generation network, we deploy three generation stages for PixelFolder, and their resolutions are set to 16, 64 and 256, respectively. In these operations, the scale of folding and unfolding k is set to 2, *i.e.*, the size of local patches is 2×2. The dimensions of initialized tensors are all 512, except for the last stage which is set to 128. Then these initialized tensors are all reduced to 32 via linear projections before pixel folding. The recovered pixel tensors after pixel unfolding are also projected to *RGB* by linear projections. For the discriminator, we use a residual convolution network following the settings in StyleGAN2 [20] and CIPS [2], which has *FusedLeakyReLU* activation functions and minibatch standard deviation layers [18].

In terms of training, we use *non-saturating logistic GAN* loss [20] with *R1* penalty [37] to optimize PixelFolder. *Adam* optimizer [22] is used with a learning rate of 2×10^{-3}, and its hyperparameters β_0 and β_1 are set to 0 and 0.99, respectively. The batch size is set to 32 , and the models are trained on 8 NVIDIA V100 32GB GPUs for about four days.

5.4 Quantitative analysis

Comparison with the state-of-the-arts. We first compare the efficiency of PixelFolder with CIPS [2], StyleGAN2 [20] and INR-GAN [45] in Tab. 1. From this table, we can find that the advantages of PixelFolder in terms of parameter size, computation complexity and inference speed are very obvious. Compared with CIPS, our method can reduce parameters by 53%, while the reduction in computation complexity (GMACs) is more distinct, about 89%. The inference speed is even improved by about 7×. These results strongly confirm the validity of our progressive modeling paradigm and pixel folding operations applied to PixelFolder. Meanwhile, compared with StyleGAN2, the efficiency of PixelFolder is also superior, which reduces 31% parameters and 72% GMACs and speed up the inference by about 76%. Also as a multi-stage method, INR-GAN is still

Method	FFHQ, 256×256			LSUN Church, 256×256		
	FID ↓	Precision ↑	Recall ↑	FID ↓	Precision ↑	Recall ↑
INR-GAN [45]	4.95	0.631	0.465	4.04	0.590	0.465
CIPS [2]	4.38	0.670	0.407	2.92	0.603	0.474
StyleGAN2 [20]	3.83	0.661	**0.528**	3.86	-	-
PixelFolder(Ours)	**3.77**	**0.683**	0.526	**2.45**	**0.630**	**0.542**

Table 2: The performance comparison of PixelFolder and the SOTA methods on FFHQ [20] and LSUN Church [42]. The proposed PixelFolder not only has better performance than existing pixel synthesis methods, *i.e.*, INR-GAN and CIPS, but also achieves new SOTA performance on both benchmarks.

Settings	#Parm (M) ↓	GMACs ↓	FID ↓	Precision ↑	Recall ↑
Fold+Unfold (base)	**20.84**	**23.78**	**5.49**	**0.679**	**0.514**
Fold+DeConv	29.41	86.53	5.60	0.667	0.371
Down-Sampling+DeConv	29.21	89.38	5.53	0.679	0.456

Table 3: Ablation study on FFHQ. The models of all settings are trained with 200k steps for a quick comparison. These results show the obvious advantages of pixel folding (Fold+Unfold) over down-sampling and DeConv.

inferior to the proposed PixelFolder in terms of parameter size and computation overhead, *i.e.*, nearly 5× more parameters and 1.6× more GMACs. In terms of inference, INR-GAN is a bit faster mainly due to its optimized implementation [2]. Conclusively, these results greatly confirm the superior efficiency of PixelFolder over the compared image generation methods.

We further benchmark these methods on FFHQ and LUSN Church, of which results are given in Tab. 2. From this table, we can first observe that on all metrics of two datasets, the proposed PixelFolder greatly outperforms the latest pixel synthesis network, *i.e.*, CIPS[2] and INR-GAN [45], which strongly validates the motivations of our method about efficient pixel synthesis. Meanwhile, we can observe that compared to StyleGAN2, PixelFolder is also very competitive and obtains new SOTA performance on FFHQ and LSUN Church, *i.e.*, 3.77 FID and 2.45 FID, respectively. Overall, these results suggest that PixelFolder is a method of great potential in image generation, especially considering its high efficiency and low expenditure.

Ablation studies. We further ablates pixel folding operations on FFHQ, of which results are given in Tab. 3. Specifically, we replace the pixel folding and unfolding with down-sampling and deconvolution (DeConv.) [20], respectively.

From these results, we can observe that although these operations can also serve to reduce or recover tensor resolutions, their practical effectiveness is much

[2] INR-GAN optimizes the CUDA kernels to speed up inference.

Settings	#Parm (M) ↓	GMACs ↓	FID ↓	Precision ↑	Recall ↑
PixelFolder	20.84	23.78	**4.78**	**0.602**	**0.517**
w/o coordinate embeddings	**20.32**	**23.64**	4.95	0.598	0.500
w/o multi-stage connection	20.84	23.78	5.46	0.532	0.441

Table 4: Ablation study on LSUN Church. The models of all settings are trained with 200k steps for a quick comparison. "*w/o design*" is not cumulative and only represents the performance of PixelFolder without this design/setting.

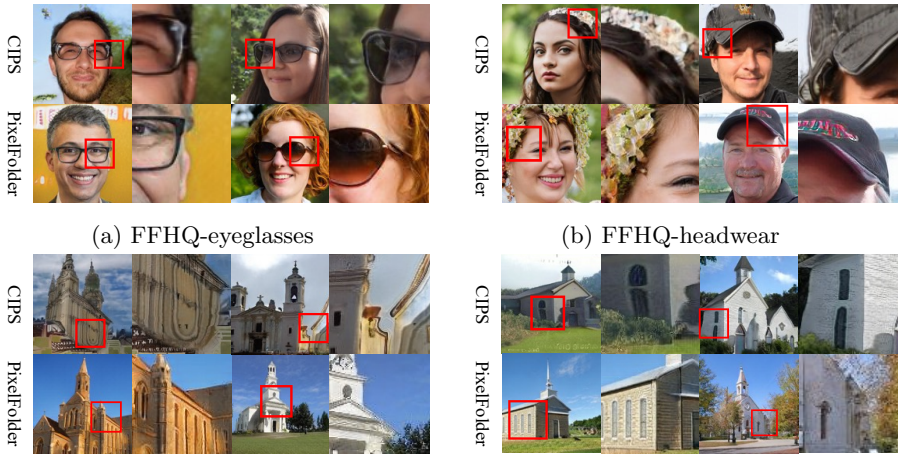
Fig. 4: Comparison of the image interpolations by CIPS [2] and PixelFolder. The interpolation is computed by $\mathbf{z} = \alpha\mathbf{z}_1 + (1-\alpha)\mathbf{z}_2$, where $\mathbf{z}_1$ and $\mathbf{z}_2$ refer to the left-most and right-most samples, respectively.

inferior than our pixel folding operations, *e.g.* 5.49 FID (fold+unfold) *v.s.* 8.36 FID (down-sampling+DeConv). These results greatly confirm the merit of pixel folding in preserving pixel-wise independence, which can help the model exploit pixel-wise prior knowledge. In Tab. 4, we examine the initialization of pixel tensor and the impact of multi-stage connection. From this table, we can see that only using Fourier features without coordinate embeddings slightly reduces model performance, but this impact is smaller than that in CIPS [2]. This result also subsequently suggests that PixelFolder do not rely on large parameterized tensors to store pixel-wise prior knowledge, leading to better model compactness. Meanwhile, we also notice that without the multi-stage connection, the performance drops significantly, suggesting the importance of joint multi-scale pixel regression, as discussed in Sec. 4.1. Overall, these ablation results well confirm the effectiveness of the designs of PixelFolder.

5.5 Qualitative analysis

To obtain deep insight into the proposed PixelFolder, we further visualize its synthesized images as well as the ones of other SOTA methods.

Comparison with CIPS. We first compare the image interpolations of PixelFolder and CIPS on two benchmarks, *i.e.*, FFHQ and LSUN Church, as shown in Fig. 4. It can be obviously seen that the interpolations by PixelFolder are more natural and reasonable than those of CIPS, especially in terms of local imaging. We further present more images synthesized by two methods in Fig. 1 and Fig.



(a) FFHQ-eyeglasses (b) FFHQ-headwear

(c) LSUN Church-wavy texture (d) LSUN Church-pixel offset

Fig. 5: Comparison of the generated images by CIPS [2] and PixelFolder on FFHQ and LSUN Church. The overall quality of images generated by PixelFolder is better than that of CIPS. Meanwhile, PixelFolder can better handle the local imagery incongruity, confirming the effectiveness of its designs.

5. From these examples, a quick observation is that the overall image quality of PixelFolder is better than CIPS. The synthesized faces by PixelFolder look more natural and vivid, which also avoid obvious deformations. Meanwhile, the surroundings and backgrounds of the generated church images by PixelFolder are more realistic and reasonable, as shown in Fig. 5c-5d. In terms of local imaging, the merit of PixelFolder becomes more obvious. As discussed in this paper, CIPS is easy to produce local pixel incongruities due to its relatively independent pixel modeling strategy [2]. This problem is reflected in its face generations, especially the hair details. In contrast, PixelFolder well excels in local imaging, such as the synthesis of accessories and hat details, as shown in Fig. 5a-5b. Meanwhile, CIPS is also prone to wavy textures and distortions in the church images, while these issues are greatly alleviated by PixelFolder. Conclusively, these findings well validate the motivations of PixelFolder for image generation.

Comparison of stage-wise visualizations. We also compare PixelFolder with CIPS, StyleGAN2 and INR-GAN by visualizing their stage-wise results, as shown in Fig. 6. From these examples, we can first observe that the intermediate results of other progressive methods, *i.e.*, StyleGAN2 and INR-GAN, are too blurry to recognize. In contrast, PixelFolder and CIPS can depict the outline of generated faces in the initial and intermediate stages. This case suggests that PixelFolder and CIPS can well exploit the high-frequency information provided by Fourier features [2], verifying the merits of end-to-end pixel regression. We

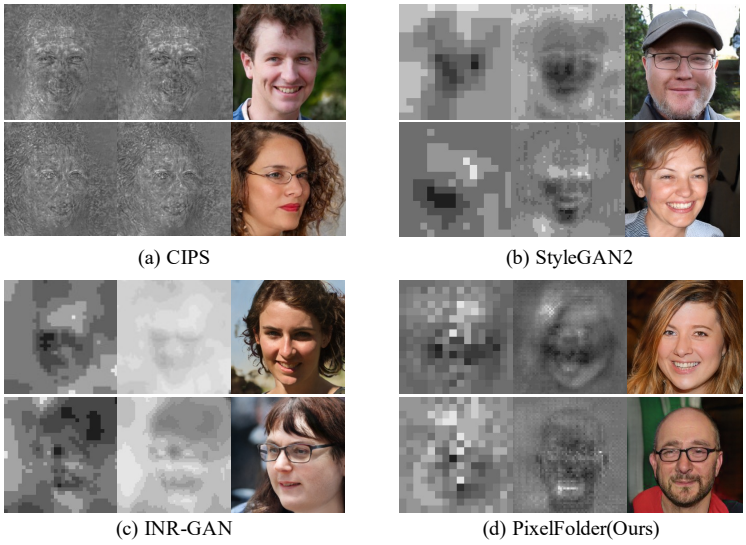


Fig. 6: Comparison of the stage-wise synthesis by the SOTA methods and PixelFolder. The color spaces of the first two hidden images are uniformly adjusted for better observation. We chose the hidden images of all methods from the same number of convolution layers. Pixel-synthesis based methods, such as CIPS [2] and PixelFolder, present more interpretable results in initial steps, where PixelFolder can also provide better outline details.

can also see that PixelFolder can learn more details than CIPS in the intermediate features, which also suggests the superior efficiency of PixelFolder in face generation. Meanwhile, the progressive refinement (from left to right) also makes PixelFolder more efficient than CIPS in computation overhead and memory footprint. We attribute these advantages to the pixel folding operations and the multi-stage paradigm of PixelFolder, which can help the model exploit prior knowledge in different generation stages.

Comparison of pixel folding and its alternatives. In Fig. 7, we visualize the generations of PixelFolder with pixel folding operations and the alternatives mentioned in Tab. 3. From these examples, we can find that although down-sampling and DeConv. can also serve to change the resolution of hidden pixel tensors, their practical effectiveness is still much inferior than that of pixel folding. We attribute these results to the unique property of pixel folding in preserving pixel-wise prior knowledge for end-to-end pixel regression. Meanwhile, we also note that when using these alternatives, there is still the problem of local image incongruity, which however can be largely avoided by pixel foldings. These results greatly validate the motivation and effectiveness of the pixel folding operations.

(a) folding+unfolding (b) folding+DeConv (c) downsample+DeConv

Fig. 7: Comparisons of PixelFolder with pixel folding operations (folding+unfolding) and the alternatives (*i.e.*, folding+DeConv. and downsampling+DeConv). Compared with these alternatives, pixel folding operations can well preserve pixel-wise prior knowledge for generation, leading to much better image quality. Meanwhile, pixel folding can also well tackle with local imagery incongruities.

6 Conclusions

In this paper, we propose a novel pixel synthesis network towards efficient image generation, termed *PixelFolder*. Specifically, PixelFolder considers the pixel synthesis as a problem of progressive pixel regression, which can greatly reduce the excessive overhead caused by large tensor transformations. Meanwhile, we also apply novel *pixel folding* operations to further improve model efficiency while preserving the property of end-to-end pixel regression. With these novel designs, PixelFolder requires much less computational and memory overhead than the latest pixel synthesis methods, such as CIPS and INR-GAN. Meanwhile, compared with the state-of-the-art method StyleGAN2, PixelFolder is also more efficient. With much higher efficiency, the proposed PixelFolder exhibits new SOTA performance on FFHQ and LSUN Church benchmarks, *i.e.*, 3.77 FID and 2.45 FID, respectively, yielding a great potential in image generation.

Acknowledgements This work is supported by the National Science Fund for Distinguished Young (No.62025603), the National Natural Science Foundation of China (No.62025603, No. U1705262, No. 62072386, No. 62072387, No. 62072389, No. 62002305, No.61772443, No. 61802324 and No. 61702136) and Guangdong Basic and Applied Basic Research Foundation (No.2019B1515120049).

References

1. Afifi, M., Brubaker, M.A., Brown, M.S.: Histogan: Controlling colors of gan-generated and real images via color histograms. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 7941–7950 (2021)
2. Anokhin, I., Demochkin, K., Khakhulin, T., Sterkin, G., Lempitsky, V., Korzhenkov, D.: Image generators with conditionally-independent pixel synthesis. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 14278–14287 (2021)
3. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: International conference on machine learning. pp. 214–223. PMLR (2017)
4. Brock, A., Donahue, J., Simonyan, K.: Large scale gan training for high fidelity natural image synthesis. In: International Conference on Learning Representations (2018)
5. Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., Abbeel, P.: Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In: Proceedings of the International Conference on Neural Information Processing Systems. pp. 2180–2188 (2016)
6. Choi, Y., Choi, M., Kim, M., Ha, J.W., Kim, S., Choo, J.: Stargan: Unified generative adversarial networks for multi-domain image-to-image translation. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 8789–8797 (2018)
7. Denton, E.L., Chintala, S., Fergus, R., et al.: Deep generative image models using a laplacian pyramid of adversarial networks. Advances in neural information processing systems **28** (2015)
8. Dolhansky, B., Ferrer, C.C.: Eye in-painting with exemplar generative adversarial networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 7902–7911 (2018)
9. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. Advances in neural information processing systems **27** (2014)
10. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein gans. Advances in neural information processing systems **30** (2017)
11. He, Z., Kan, M., Shan, S.: Eigengan: Layer-wise eigen-learning for gans. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 14408–14417 (2021)
12. Heusel, M., Ramsauer, H., Unterthiner, T., Nessler, B., Hochreiter, S.: Gans trained by a two time-scale update rule converge to a local nash equilibrium. Advances in neural information processing systems **30** (2017)
13. Howard, A.G., Zhu, M., Chen, B., Kalenichenko, D., Wang, W., Weyand, T., Andreetto, M., Adam, H.: Mobilenets: Efficient convolutional neural networks for mobile vision applications. arXiv preprint arXiv:1704.04861 (2017)
14. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 1501–1510 (2017)
15. Hudson, D.A., Zitnick, C.L.: Generative adversarial transformers. Advances in neural information processing systems **139** (2021)
16. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 1125–1134 (2017)

17. Ji, J., Ma, Y., Sun, X., Zhou, Y., Wu, Y., Ji, R.: Knowing what to learn: A metric-oriented focal mechanism for image captioning. IEEE Transactions on Image Processing **31**, 4321–4335 (2022). https://doi.org/10.1109/TIP.2022.3183434
18. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of gans for improved quality, stability, and variation. In: International Conference on Learning Representations (2018)
19. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 4401–4410 (2019)
20. Karras, T., Laine, S., Aittala, M., Hellsten, J., Lehtinen, J., Aila, T.: Analyzing and improving the image quality of stylegan. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 8110–8119 (2020)
21. Kim, H., Choi, Y., Kim, J., Yoo, S., Uh, Y.: Exploiting spatial dimensions of latent in gan for real-time image editing. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 852–861 (2021)
22. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. arXiv preprint arXiv:1412.6980 (2014)
23. Kingma, D.P., Welling, M.: Auto-encoding variational bayes. arXiv preprint arXiv:1312.6114 (2013)
24. Kynkäänniemi, T., Karras, T., Laine, S., Lehtinen, J., Aila, T.: Improved precision and recall metric for assessing generative models. Advances in neural information processing systems **32** (2019)
25. Lee, K., Chang, H., Jiang, L., Zhang, H., Tu, Z., Liu, C.: Vitgan: Training gans with vision transformers. arXiv preprint arXiv:2107.04589 (2021)
26. Li, B., Qi, X., Lukasiewicz, T., Torr, P.: Controllable text-to-image generation. Advances in neural information processing systems **32** (2019)
27. Li, X., Zhang, S., Hu, J., Cao, L., Hong, X., Mao, X., Huang, F., Wu, Y., Ji, R.: Image-to-image translation via hierarchical style disentanglement. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 8639–8648 (2021)
28. Liang, J., Zeng, H., Zhang, L.: High-resolution photorealistic image translation in real-time: A laplacian pyramid translation network. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 9392–9400 (2021)
29. Lin, C.H., Chang, C.C., Chen, Y.S., Juan, D.C., Wei, W., Chen, H.T.: Coco-gan: Generation by parts via conditional coordinating. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 4512–4521 (2019)
30. Lin, J., Zhang, R., Ganz, F., Han, S., Zhu, J.Y.: Anycost gans for interactive image synthesis and editing. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 14986–14996 (2021)
31. Liu, H., Navarrete Michelini, P., Zhu, D.: Deep networks for image-to-image translation with mux and demux layers. In: Proceedings of the European Conference on Computer Vision. pp. 0–0 (2018)
32. Liu, R., Lehman, J., Molino, P., Petroski Such, F., Frank, E., Sergeev, A., Yosinski, J.: An intriguing failing of convolutional neural networks and the coordconv solution. Advances in neural information processing systems **31** (2018)
33. Liu, R., Ge, Y., Choi, C.L., Wang, X., Li, H.: Divco: Diverse conditional image synthesis via contrastive generative adversarial network. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 16377–16386 (2021)

34. Luo, G., Zhou, Y., Sun, X., Ding, X., Wu, Y., Huang, F., Gao, Y., Ji, R.: Towards language-guided visual recognition via dynamic convolutions. arXiv preprint arXiv:2110.08797 (2021)
35. Luo, G., Zhou, Y., Sun, X., Wang, Y., Cao, L., Wu, Y., Huang, F., Ji, R.: Towards lightweight transformer via group-wise transformation for vision-and-language tasks. IEEE Transactions on Image Processing **31**, 3386–3398 (2022)
36. Ma, Y., Ji, J., Sun, X., Zhou, Y., Wu, Y., Huang, F., Ji, R.: Knowing what it is: Semantic-enhanced dual attention transformer. IEEE Transactions on Multimedia pp. 1–1 (2022). https://doi.org/10.1109/TMM.2022.3164787
37. Mescheder, L., Geiger, A., Nowozin, S.: Which training methods for gans do actually converge? In: International conference on machine learning. pp. 3481–3490. PMLR (2018)
38. Park, T., Efros, A.A., Zhang, R., Zhu, J.Y.: Contrastive learning for unpaired image-to-image translation. In: Proceedings of the European Conference on Computer Vision. pp. 319–345. Springer (2020)
39. Park, T., Zhu, J.Y., Wang, O., Lu, J., Shechtman, E., Efros, A., Zhang, R.: Swapping autoencoder for deep image manipulation. Advances in neural information processing systems **33**, 7198–7211 (2020)
40. Patashnik, O., Wu, Z., Shechtman, E., Cohen-Or, D., Lischinski, D.: Styleclip: Text-driven manipulation of stylegan imagery. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2085–2094 (2021)
41. Peng, J., Zhou, Y., Sun, X., Cao, L., Wu, Y., Huang, F., Ji, R.: Knowledge-driven generative adversarial network for text-to-image synthesis. IEEE Transactions on Multimedia (2021)
42. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. arXiv preprint arXiv:1511.06434 (2015)
43. Sajjadi, M.S., Bachem, O., Lucic, M., Bousquet, O., Gelly, S.: Assessing generative models via precision and recall. Advances in neural information processing systems **31** (2018)
44. Shi, W., Caballero, J., Huszár, F., Totz, J., Aitken, A.P., Bishop, R., Rueckert, D., Wang, Z.: Real-time single image and video super-resolution using an efficient sub-pixel convolutional neural network. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 1874–1883 (2016)
45. Skorokhodov, I., Ignatyev, S., Elhoseiny, M.: Adversarial generation of continuous images. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 10753–10764 (2021)
46. Tang, H., Bai, S., Zhang, L., Torr, P.H., Sebe, N.: Xinggan for person image generation. In: Proceedings of the European Conference on Computer Vision
47. Van Den Oord, A., Vinyals, O., et al.: Neural discrete representation learning. Advances in neural information processing systems **30** (2017)
48. Wang, Y., Qi, L., Chen, Y.C., Zhang, X., Jia, J.: Image synthesis via semantic composition. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 13749–13758 (2021)
49. Wang, Y., Chen, X., Zhu, J., Chu, W., Tai, Y., Wang, C., Li, J., Wu, Y., Huang, F., Ji, R.: Hififace: 3d shape and semantic prior guided high fidelity face swapping. arXiv preprint arXiv:2106.09965 (2021)
50. Xu, T., Zhang, P., Huang, Q., Zhang, H., Gan, Z., Huang, X., He, X.: Attngan: Fine-grained text to image generation with attentional generative adversarial networks. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 1316–1324 (2018)

51. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Generative image inpainting with contextual attention. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 5505–5514 (2018)
52. Zhang, H., Xu, T., Li, H., Zhang, S., Wang, X., Huang, X., Metaxas, D.N.: Stackgan++: Realistic image synthesis with stacked generative adversarial networks. IEEE transactions on pattern analysis and machine intelligence **41**(8), 1947–1962 (2018)
53. Zhang, Z., Xie, Y., Yang, L.: Photographic text-to-image synthesis with a hierarchically-nested adversarial network. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 6199–6208 (2018)
54. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: Proceedings of the IEEE International Conference on Computer Vision. pp. 2223–2232 (2017)
55. Zhu, P., Abdal, R., Qin, Y., Wonka, P.: Sean: Image synthesis with semantic region-adaptive normalization. In: Proceedings of the IEEE conference on Computer Vision and Pattern Recognition. pp. 5104–5113 (2020)