Spiral Generative Network for Image Extrapolation

Dongsheng Guo[1†][0000−0002−7084−9427], Hongzhi Liu[1†][0000−0002−8688−0066], Haoru Zhao[1][0000−0002−2583−9449], Yunhao Cheng[1][0000−0001−9496−5986], Qingwei Song[1][0000−0001−5244−6010], Zhaorui Gu[1][0000−0002−6673−7932], Haiyong Zheng[1*][0000−0002−8027−0734], and Bing Zheng[1,2*][0000−0003−2295−3569]

[1] Underwater Vision Lab (ouc.ai), Ocean University of China
[2] Sanya Oceanographic Institution, Ocean University of China
{guodongsheng,liuhongzhi,zhaohaoru,chengyunhao,songqingyu}@stu.ouc.edu.cn
{guzhaorui,zhenghaiyong,bingzh}@ouc.edu.cn
[*] Corresponding authors [†] Equal contribution

Abstract. In this paper, motivated by human natural ability to perceive unseen surroundings imaginatively, we propose a novel Spiral Generative Network, SpiralNet, to perform image extrapolation in a spiral manner, which regards extrapolation as an evolution process growing from an input sub-image along a spiral curve to an expanded full image. Our SpiralNet, consisting of ImagineGAN and SliceGAN, disentangles image extrapolation problem into two independent sub-tasks as semantic structure prediction (via ImagineGAN) and contextual detail generation (via SliceGAN), making the whole task more tractable. The design of SliceGAN implicitly harnesses the correlation between generated contents and extrapolating direction, divide-and-conquer while generation-by-parts. Extensive experiments on datasets covering both objects and scenes under different cases show that our method achieves state-of-the-art performance on image extrapolation. We also conduct ablation study to validate efficacy of our design. Our code is available at https://github.com/zhenglab/spiralnet.

Keywords: Image extrapolation · GAN · cGAN · SpiralNet

1 Introduction

Suppose that, given a sub-image (*e.g.*, part of a human face), what happens in your mind when you are asked to draw the entire image (*i.e.*, a whole face) beyond its boundary? Actually, although the surrounding regions are unseen, we humans usually first imagine the entire image preliminarily according to the prior knowledge [21,23], while then draw the details outward from inside progressively based on the sub-image and the imaginary image [36].

Image extrapolation [48] is such a task in computer vision, which aims to fill the surrounding region of a sub-image, *e.g.*, completing an object appearance with part of it or predicting the unseen view from a scene picture. This task

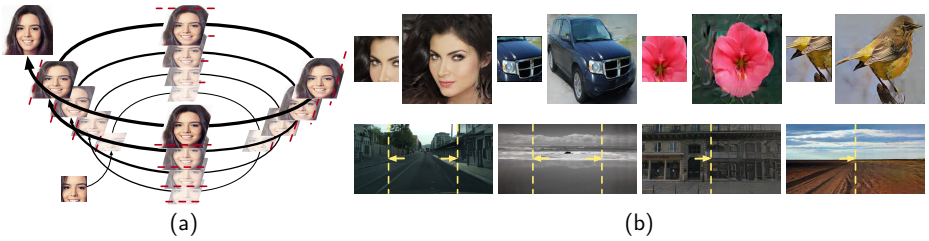

Fig. 1. (a) Our SpiralNet expands a sub-image in four directions evolving along a spiral curve to reach a full image. (b) Exemplar results on different datasets in different cases.

is extremely challenging since that: (a) the extrapolated image must be realistic with a reasonable and meaningful context; and (b) the extrapolated region should be consistent in structure and texture with the original sub-image.

Recently, although extrapolating an image is so challenging even for our humans, thanks to the development of Generative Adversarial Network (GAN) [11], a lot of efforts have been made on this task to step forward achieving good performance as well. However, existing GAN-based methods [48,43] for image extrapolation mainly generate a whole image and paste the given part onto it, making the final image look jarring. In addition, due to distant contextual generation problem, directly applying inpainting methods [27,51] tends to generate blurry or repetitive pixels with inconsistent semantics [43].

In this work, motivated by human natural ability to perceive unseen surroundings imaginatively, we propose a novel Spiral Generative Network, **SpiralNet** for short, performing the extrapolation in a spiral fashion. We regard image extrapolation as an evolution process, as illustrated in Fig. 1a, growing from an input sub-image along a spiral curve to an expanded full image. Essentially, SpiralNet is a progressive part-to-whole generation method, "drawing" a full image in four directions slice by slice in a spiral way. In such a way, generation of large surrounding area is divided into turns of easier slice generations, thus yielding results with semantic consistency and vivid details. Fig. 1b shows our extrapolating examples in different cases, and we can see that the extrapolating results are all realistic themselves while consistent with original sub-images.

Our **contributions** include: (a) A novel generative framework that extrapolates a sub-image to a full image in a spiral fashion; (b) A SliceGAN that tackles slice-wise image generation and an ImagineGAN that generates imaginary output guiding SliceGAN, equipping a new hue-color loss; (c) State-of-the-art performance on a variety of datasets for image extrapolation in different cases.

2 Related Work

2.1 Generative Adversarial Networks

Starting from the groundbreaking work by Goodfellow et al. [11], GANs have drawn wide attention in computer vision world. Then, many efforts have been

made on GANs to improve the generative performance [37,29,1,12,31,19,4,20]. Thereinto conditional GAN (cGAN) [30] is allowed to generate images that have certain conditions or attributes, which can be widely used in many tasks, for instance, image-to-image translation [17,55,16,25,5]. Image extrapolation aims to generate the surrounding regions from the visual content, thus can be considered as an image-conditioned generation task.

Recent cGAN-based models have shown promising results on similar tasks like image inpainting [50,27,51,32], image editing [54,7], and texture synthesis [26,49,42]. But for image extrapolation task, it's really hard for a cGAN to generate semantically consistent content with visually pleasing details, while directly using inpainting methods to image extrapolation is prone to resulting in poor results due to distant contextual generation problem [48,43].

2.2 Image Extrapolation

Image extrapolation fills content outside of visual images. Previous possible solutions can be typically categorized as non-parametric [9,35,2,3,52,45,40] and parametric [48,43] methods. Non-parametric methods mainly formulate the problem into matching and stitching based on a pre-constructed dataset, specifically, they usually retrieve the candidate images by subimage matching, and stitch these wrapped images into the input. Thereby they work in a data-driven manner that is strictly limited by the used dataset, also it's hard to be applied in complex cases like fine texture or sophisticated scene. Recently GAN-based approaches have made great efforts in overcoming weaknesses of non-parametric methods. Particularly, Wang et al. [48] first proposed a cGAN-based approach to address the issues of size expansion and one-side constraints. Teterwak et al. [43] also followed the cGAN framework by introducing semantic conditioning to the discriminator for one-side image extension.

Compared to current cGAN-based extrapolation methods, our method disentangles image extrapolation problem into two relatively independent sub-tasks as semantic structure prediction (via ImagineGAN) and contextual detail generation (via SliceGAN), making the whole task more tractable. The design of SliceGAN implicitly harnesses the correlation between generated contents and extrapolating direction, divide-and-conquer while generation-by-parts.

3 Spiral Generative Network

We regard image extrapolation as an evolution process shown in Fig. 2, growing from an input sub-image along a spiral curve to an expanded full image. Given an input image $\mathbf{X} \in \mathbb{R}^{h \times w \times c}$ and filling margin $m = (m^l, m^t, m^r, m^b)$, where m^l, m^t, m^r, and m^b refer to left, top, right, and bottom filling margin respectively. The goal of image extrapolation is to output an image $\hat{\mathbf{Y}} \in \mathbb{R}^{h' \times w' \times c}$ with a visually pleasing appearance, where $h' = h + m^t + m^b$, $w' = w + m^l + m^r$, and $\mathbf{X}$ is a sub-image of $\hat{\mathbf{Y}}$.

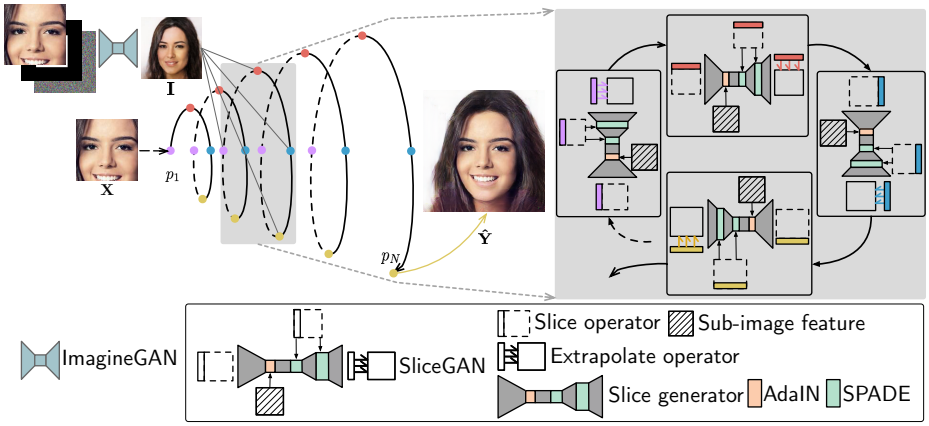

Fig. 2. Our spiral growing evolution for image extrapolation. Refer to text for details.

We consider that $\mathbf{X}$ evolves along a series of points $P = \{p_1, p_2, \cdots, p_N\}$ on a spiral curve until it reaches $\hat{\mathbf{Y}}$ after N growth. Each point p on the spiral curve is represented by its turn number and corresponding growing direction (*i.e.*, left, top, right, and bottom). For convenience, we consider that the growing size τ at each point is the same.

According to given margin m and growing size τ, we can figure out total number of points N and total number of turns T for the spiral growing. As to point p on spiral curve, we denote growing function at p as $G_p(\cdot)$. For the k-th point p_k, the input $\mathbf{X}_{p_k}$ grows to $\mathbf{X}_{p_{k+1}}$ by $\mathbf{X}_{p_{k+1}} = G_{p_k}(\mathbf{X}_{p_k})$, where p_{k+1} represents next point on spiral curve. While growing from $\mathbf{X}_{p_k}$ to $\mathbf{X}_{p_{k+1}}$, sizes of input/output and filling margin change accordingly. Finally, $\mathbf{X} \in \mathbb{R}^{h \times w \times c}$ evolves to $\hat{\mathbf{Y}} \in \mathbb{R}^{h' \times w' \times c}$ through growing at N points in T turns, and the evolution can be expressed as:

$$\hat{\mathbf{Y}} = F(\mathbf{X}) = G_{p_N}(G_{p_{N-1}}(\cdots(G_{p_1}(\mathbf{X})))). \tag{1}$$

Meanwhile, h and w change to h' and w' respectively, m_{p_N} becomes $(0, 0, 0, 0)$.

Notably, four total numbers of turns in four directions are not necessarily equal, since sub-image $\mathbf{X}$ may not be located in the center of $\hat{\mathbf{Y}}$, such that the growth in four directions will not stop at the same time.

3.1 ImagineGAN

We present ImagineGAN to "draw" an imaginary result of extrapolation according to given sub-image, regarded as a coarse reference for SliceGAN to refine. We propose this strategy by mimicking human imagination [21,23] to address the image extrapolation task.

Our ImagineGAN is essentially a cGAN with an encoder-decoder generator $G_{\mathbf{I}}$, encoding given sub-image $\mathbf{X} \in \mathbb{R}^{h \times w \times c}$ with a margin mask $\mathbf{M}$ and an

uniform noise distribution $\mathbf{Z}$, to generate an imaginary output $\mathbf{I} \in \mathbb{R}^{h \times w \times c}$: $\mathbf{I} = G_{\mathbf{I}}(\mathbf{X}, \mathbf{M}, \mathbf{Z})$. Both inputs $\mathbf{X}, \mathbf{M}, \mathbf{Z}$ and output $\mathbf{I}$ have the same small size (*e.g.*, 128×128), taking full advantage of GAN for low-resolution image generation.

Except for adversarial loss, our ImagineGAN is designed with extra losses for better performance. In particular, we propose a novel hue-color loss in this task, to eliminate bright color spots and avoid dark while stabilize the training.

Hue-Color Loss. Hue is the most basic element of a color and what most people think of when they think "color" [28,10]. Thus, it would be helpful to keep consistent hue during extrapolation. However, according to cylindrical HSL/HSV representations of an RGB colorcube [13], same hue may lead to quite different color appearance, which should be avoided for extrapolation. To constrain both hue consistency and color harmony, we formulate a new hue-color loss as:

$$\mathcal{L}^{\mathbf{I}}_{hue} = \frac{1}{h \times w} \sum_{i,j} \left\{ 1 - \min \left[\cos(\mathbf{I}_{ij}, \bar{\mathbf{Y}}_{ij}), \cos(\mathbb{1} - \mathbf{I}_{ij}, \mathbb{1} - \bar{\mathbf{Y}}_{ij}) \right] + \xi \right\}^{\gamma}, \quad (2)$$

where $\bar{\mathbf{Y}} \in \mathbb{R}^{h \times w \times c}$ is the downscaled result of real image $\mathbf{Y}$, ξ is a very small number added to avoid zero, $\gamma < 1$ is used to stretch the difference for better optimization, and we set $\xi = 0.001$ and $\gamma = 0.4$ in our experiments.

Compared to color loss [47] and reconstruction loss [17], our hue-color loss cares about real "color" regardless of gray (please refer to *supplementary file* for mathematical derivation), which is really beneficial to tasks like image extrapolation, which requires both semantic consistency and visual realism. Actually, in our work, we find that synthesized images usually become dark while bright color spots appear on much colorful situations (*e.g.*, Flowers [33]), and our hue-color loss does solve this issue. Furthermore, we surprisingly find that this loss can stabilize the training as well. Please see Section 4.5 for ablative experiments.

Perceptual Loss. Following previous works [32,18], we also use perceptual loss $\mathcal{L}^{\mathbf{I}}_{perc}$ to penalize imaginary output $\mathbf{I}$ for that is not perceptually similar to $\bar{\mathbf{Y}}$, by defining a distance measure between activation maps:

$$\mathcal{L}^{\mathbf{I}}_{perc} = \mathbb{E}\left[\sum_{u} \frac{1}{N_u} \|\sigma_u(\mathbf{I}) - \sigma_u(\bar{\mathbf{Y}})\|_1 \right], \quad (3)$$

where N_u is the number of elements in the u-th activation layer, σ_u is the activation map of the u-th layer of a pretrained network (*e.g.*, VGG-19 [41]).

Adversarial Loss. The adversarial loss $\mathcal{L}^{\mathbf{I}}_{adv}$ of ImagineGAN is:

$$\mathcal{L}^{\mathbf{I}}_{adv}(G_{\mathbf{I}}, D_{\mathbf{I}}) = \mathbb{E}_{(\bar{\mathbf{Y}}, \mathbf{X})}[\log(D_{\mathbf{I}}(\bar{\mathbf{Y}}, \mathbf{X}))] + \mathbb{E}_{(\mathbf{I}, \mathbf{X})}[\log(1 - D_{\mathbf{I}}(\mathbf{I}, \mathbf{X}))], \quad (4)$$

where the generator $G_{\mathbf{I}}$ is trained to minimize this objective against an adversarial discriminator $D_{\mathbf{I}}$ that tries to maximize it.

Total Loss. The total loss of ImagineGAN is:

$$\mathcal{L}^{\mathbf{I}}_{total} = \lambda^{\mathbf{I}}_{adv}\mathcal{L}^{\mathbf{I}}_{adv} + \lambda^{\mathbf{I}}_{hue}\mathcal{L}^{\mathbf{I}}_{hue} + \lambda^{\mathbf{I}}_{perc}\mathcal{L}^{\mathbf{I}}_{perc}, \quad (5)$$

where $\lambda^{\mathbf{I}}_{adv}$, $\lambda^{\mathbf{I}}_{hue}$, and $\lambda^{\mathbf{I}}_{perc}$ are weights to balance different losses. We empirically set $\lambda^{\mathbf{I}}_{adv} = 0.1$, $\lambda^{\mathbf{I}}_{hue} = 10$, and $\lambda^{\mathbf{I}}_{perc} = 1$ for our experiments in this work.

3.2 SliceGAN

We devise a novel slice-wise GAN, dubbed SliceGAN, to implement growing function $G_p(\bullet)$ (at point p). As shown in Fig. 2, SliceGAN (at point p) consists of slice operator ψ_p, slice generator G_p^S, extrapolate operator ϕ_p, as well as a spiral discriminator D_S and an extrapolate discriminator D_E (both unshown in Fig. 2).

For the k-th point p_k ($k = 1, 2, \cdots, N$), SliceGAN G_{p_k} takes an extrapolated image $\mathbf{X}_{p_k}$ and an imaginary image $\bar{\mathbf{I}}$ as inputs, and outputs an extrapolated image $\mathbf{X}_{p_{k+1}}$. The imaginary image $\bar{\mathbf{I}}$ has the same size of $\hat{\mathbf{Y}}$, and is upscaled from $\mathbf{I}$ which has the same size of $\mathbf{X}$ and is generated by our ImagineGAN.

Slice Operator. Slice operator aims to cut slice from image. The cutting size of slice operator is equal to growing size at each point, namely τ. For SliceGAN at p_k, there are two slice operators that cut slices $S_{p_k}^{\mathbf{X}}$ and $S_{p_k}^{\bar{\mathbf{I}}}$ from $\mathbf{X}_{p_k}$ and $\bar{\mathbf{I}}$ respectively, *i.e.*, $S_{p_k}^{\mathbf{X}} = \psi_{p_k}(\mathbf{X}_{p_k})$ and $S_{p_k}^{\bar{\mathbf{I}}} = \psi_{p_k}(\bar{\mathbf{I}})$.

Slice Generator. To better use the information from imaginary slice $S_{p_k}^{\bar{\mathbf{I}}}$, original sub-image $\mathbf{X}$, and closest slice $S_{p_k}^{\mathbf{X}}$ for slice-wise extrapolation both semantically and visually, we design a new Encoder-AdaIN-SPADE|Decoder structure for slice-wise generator $G_{p_k}^S$ shown in Fig. 3. The encoder takes in charge of imaginary slice, fusing sub-image style in its latent space by AdaIN [15], and then the decoder combines with semantic information from closest slice via SPADE [34], yielding extrapolated slice: $S_{p_k}^O = G_{p_k}^S(\mathbf{X}, S_{p_k}^{\mathbf{X}}, S_{p_k}^{\bar{\mathbf{I}}})$.

It is worth mentioning that slice $S_{p_k}^{\mathbf{X}}$ in $\mathbf{X}_{p_k}$ is closest to extrapolated slice $S_{p_k}^O$ in $\mathbf{X}_{p_{k+1}}$, while $S_{p_k}^{\bar{\mathbf{I}}}$ in $\bar{\mathbf{I}}$ is corresponding to extrapolated slice $S_{p_k}^O$ in $\mathbf{X}_{p_{k+1}}$. In such a way, we semantically combine meaningful slice information from sub-image and imaginary image for slice-wise extrapolating. Notably, our SliceGAN is designed without independent discriminator, for considering semantic coherence and computational complexity (see Section 3.3).

Extrapolate Operator. Extrapolate operator aims to output an extrapolated image $\mathbf{X}_{p_{k+1}}$ by stitching output slice $S_{p_k}^O$ back to input $\mathbf{X}_{p_k}$: $\mathbf{X}_{p_{k+1}} = \phi_{p_k}(S_{p_k}^O, \mathbf{X}_{p_k})$. Now, we complete extrapolation at point p_k using one SliceGAN.

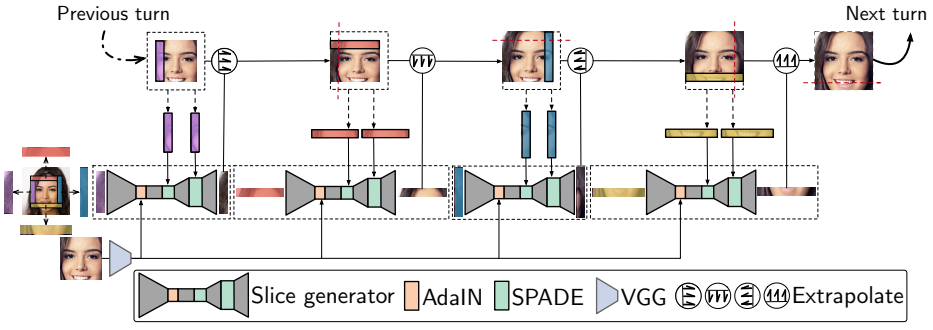

Fig. 3. Four SliceGANs in one spiral turn.

Shared Spiral SliceGAN. Our SpiralNet includes N slice generators for N points on spiral curve, while a complete spiral turn has four slice generators in four directions as shown in Fig. 3. When we need to grow more for extrapolation, we will have more slice generators, and accordingly the number of parameters for the whole SpiralNet will be huge. To tackle this issue, we share the weights of all slice generators. That is, our SpiralNet only has one independent SliceGAN.

3.3 Spiral Loss Design

Adversarial Loss. We devise a spiral discriminator D_S and an extrapolate discriminator D_E to distinguish the whole spiral evolving result $\hat{\mathbf{Y}}$ and the partial extrapolating region $\hat{\mathbf{E}}$ from the corresponding real ones $\mathbf{Y}$ and $\mathbf{E}$, where $\mathbf{E} = \mathbf{Y} \odot (\mathbb{1} - \bar{\mathbf{M}})$ and $\hat{\mathbf{E}} = \hat{\mathbf{Y}} \odot (\mathbb{1} - \bar{\mathbf{M}})$ ($\bar{\mathbf{M}} \in \mathbb{R}^{h' \times w' \times 1}$, $\odot$ represents Hadamard product). Then, the adversarial losses are:

$$\mathcal{L}_{adv}^{S}(F, D_S) = \mathbb{E}_{\mathbf{Y}}[\log(D_S(\mathbf{Y}))] + \mathbb{E}_{\hat{\mathbf{Y}}}[\log(1 - D_S(\hat{\mathbf{Y}}))], \tag{6}$$

$$\mathcal{L}_{adv}^{E}(F, D_E) = \mathbb{E}_{\mathbf{E}}[\log(D_E(\mathbf{E}))] + \mathbb{E}_{\hat{\mathbf{E}}}[\log(1 - D_E(\hat{\mathbf{E}}))], \tag{7}$$

where F is evolving function in Eq. 1, which is trained to minimize this objective against D_S and D_E that try to maximize it. The spiral adversarial loss is:

$$\mathcal{L}_{adv} = \left(\mathcal{L}_{adv}^{S} + \mathcal{L}_{adv}^{E}\right)/2. \tag{8}$$

Here spiral discriminator takes care of overall consistency, while extrapolate discriminator mainly focuses on stitching continuity.

L1 Loss. We minimize reconstructed differences between $\mathbf{Y}$ and $\hat{\mathbf{Y}}$ by:

$$\mathcal{L}_{L1} = \mathbb{E}_{(\hat{\mathbf{Y}}, \mathbf{Y})}\left[\|\hat{\mathbf{Y}} - \mathbf{Y}\|_1\right]. \tag{9}$$

Style Loss. We adopt style loss [38] to measure differences between covariances of activation maps:

$$\mathcal{L}_{style} = \mathbb{E}_{v}\left[\|G_v^{\sigma}(\hat{\mathbf{Y}}) - G_v^{\sigma}(\mathbf{Y})\|_1\right], \tag{10}$$

where G_v^{σ} is a $G_v \times G_v$ Gram matrix constructed from activation maps σ_v.

Total Loss. The total loss of our SpiralNet is:

$$\mathcal{L}_{total} = \lambda_{adv}\mathcal{L}_{adv} + \lambda_{L1}\mathcal{L}_{L1} + \lambda_{style}\mathcal{L}_{style} + \lambda_{hue}\mathcal{L}_{hue}, \tag{11}$$

where $\mathcal{L}_{hue}$ is defined the same as Eq. 2, λ_{adv}, λ_{L1}, λ_{style}, and λ_{hue} are weights to balance different losses. We empirically set $\lambda_{adv} = 0.1$, $\lambda_{L1} = 10$, $\lambda_{style} = 250$, and $\lambda_{hue} = 10$ for our experiments in this work.

3.4 Case of Unknown Margin

Suppose that only given input sub-image $\mathbf{X} \in \mathbb{R}^{h \times w \times c}$ and output size $h' \times w' \times c$, it's hard to know the position of $\mathbf{X}$ in $\hat{\mathbf{Y}}$, such that margin m is unknown. In this

case, previous approach [48] is unable to work. While, for our approach, thanks to the design of imaginary strategy (without margin mask input), we can match input sub-image $\mathbf{X}$ within upscaled imaginary output $\bar{\mathbf{I}}$ to locate position of $\mathbf{X}$ in $\hat{\mathbf{Y}}$. In such a way, we actually obtain the filling margin indirectly. Meanwhile, this strategy is also helpful to other approaches such as SRN [48]. We adopt normalized cross-correlation template matching method [39] for experiments.

3.5 Implementation Details

Network architecture. We adopt encoder-decoder structure similar to CycleGAN [55] for our ImagineGAN's and slice generators. Differently, for slice generator, we replace eight residual blocks to six in bottleneck, and moreover, we insert one AdaIN layer before bottleneck residual blocks to fuse style information, and two SPADE layers before two transposed convolution layers respectively to combine semantic information, yielding a new Encoder-AdaIN-SPADE|Decoder structure. In addition, we use patch discriminator based on pix2pix [17] for ImagineGAN's and our extrapolate discriminators with replacing batch normalization with spectral normalization [31], and Inspired by MUSICAL [46], we adopt a similar structure to DenseNet [14] as our spiral discriminator. See the details in supplementary file.

Training Details. ImagineGAN is trained independently beforehand, whose generator and discriminator are trained jointly using Adam optimizer [22] with the same parameters of learning rate $\alpha = 0.0002$, $\beta_1 = 0.5$, and $\beta_2 = 0.9$. Then, all SliceGANs and spiral/extrapolate discriminators are trained using the same settings of Adam optimizer as those in ImagineGAN.

4 Experiments

To evaluate the performance of our proposed method on image extrapolation, we conduct experiments on eight datasets: CelebA-HQ [19], Stanford Cars [24], CUB [44], Flowers [33], Paris StreetView [8], Cityscapes [6], Place365 Desert Road and Sky [53] , considering the cases of objects (faces, cars, birds and flowers) as well as scenes (streetview, cityscapes, desert road and sky).

For Stanford Cars and CUB, we crop the objects using given bounding box and then resize them to 256×256, also we drop severely distorted objects for extrapolation task. We list training and testing split on eight datasets in Table 1, where we keep default official split on Cityscapes and Place365 datasets, and select samples randomly on other datasets.

We consider three different cases of image extrapolation task for evaluation: (1) four-side extrapolation for $128 \times 128 \rightarrow 256 \times 256$ on CelebA-HQ, Stanford Cars, CUB and Flowers; (2) two-side extrapolation for $256 \times 256 \rightarrow 512 \times 256$ on Cityscapes and Place365 Sky; and (3) one-side extrapolation for $256 \times 256 \rightarrow 512 \times 256$ on Paris StreetView and Place365 Desert Road.

We compare our method with state-of-the-art Boundless [43] in one-side case and SRN [48] in all three cases. Besides, we deal with the case of unknown margin (see Section 3.4, namely SpiralNet-UM) on CelebA-HQ for instance.

Table 1. Training and testing split on eight datasets. The first four are object datasets and the rest are scene datasets.

Dataset	#Train	#Test	#Total
CelebA-HQ [19]	28,000	2,000	30,000
Stanford Cars [24]	4,166	1,000	5,166
CUB [44]	4,200	915	5,115
Flowers [33]	7,000	1,189	8,189
Cityscapes [6]	2,975	1,525	4,500
Place365 Sky [53]	5,000	100	5,100
Paris Street-View [8]	13,000	1,900	14,900
Place365 Desert Road [53]	5,000	100	5,100

Table 2. User study results. Each entry shows percentage of cases where results by SpiralNet are judged more realistic than Boundless and SRN.

SpiralNet	>Boundless	>SRN
CelebA-HQ	-	88.25%
Stanford Cars	-	88.33%
CUB	-	80.00%
Flowers	-	86.67%
Cityscapes	-	77.50%
Place365 Sky	-	80.83%
Paris StreetView	93.33%	71.67%
Place365 Desert Road	63.33%	59.17%

4.1 Quantitative Comparison

Table 3. Quantitative comparison results in different cases.

Dataset (case)	Metrics	Boundless	SRN	ImagineGAN	SpiralNet (SpiralNet-UM)
CelebA-HQ (four-side)	PSNR	-	15.17	15.09	**16.05** (15.82)
	SSIM	-	0.6752	0.6361	**0.6815** (0.6350)
	FID	-	32.25	45.92	**21.17** (23.88)
Stanford Cars (four-side)	PSNR	-	13.34	13.56	**14.31**
	SSIM	-	0.5479	0.5107	**0.5775**
	FID	-	37.11	53.12	**23.64**
CUB (four-side)	PSNR	-	15.31	15.44	**16.22**
	SSIM	-	0.5112	0.4805	**0.5313**
	FID	-	80.13	97.61	**56.50**
Flowers (four-side)	PSNR	-	13.49	14.98	**15.67**
	SSIM	-	0.4660	0.4681	**0.5078**
	FID	-	66.01	75.11	**52.14**
Cityscapes (two-side)	PSNR	-	20.33	20.12	**20.43**
	SSIM	-	0.6980	0.6642	**0.7125**
	FID	-	28.90	114.00	**22.34**
Place365 Sky (two-side)	PSNR	-	21.44	19.90	**21.75**
	SSIM	-	0.7716	0.7479	**0.7834**
	FID	-	52.50	94.41	**51.55**
Paris StreetView (one-side)	PSNR	16.70	16.37	16.39	**17.43**
	SSIM	0.5846	0.5641	0.5377	**0.5970**
	FID	52.02	**30.27**	75.19	35.58
Place365 Desert Road (one-side)	PSNR	19.04	19.45	19.07	**20.22**
	SSIM	0.6825	0.6877	0.6761	**0.7026**
	FID	86.10	85.59	122.50	**80.66**

Following Boundless [43] and SRN [48], we use peak signal-to-noise ratio (PSNR), structural similarity index measure (SSIM) and Frechet Inception Distance (FID) as metrics for evaluating semantic consistency and visual realism (higher is better for PSNR and SSIM, lower is better for FID), and results in Table 3 validate that our SpiralNet outperforms Boundless and SRN in almost all cases. Also note that our ImagineGAN (as a cGAN) performs worse than final SpiralNet and extremely poor in terms of FID, indicating visually unpleasing results.

To compare photorealism and faithfulness of extrapolated outputs, we also conduct a user study of pairwise A/B tests [17,48]. Our settings are similar to SRN [48]. For each dataset, we randomly choose 40 pairwise results extrapolated by SpiralNet vs. Boundless and SpiralNet vs. SRN separately from the same

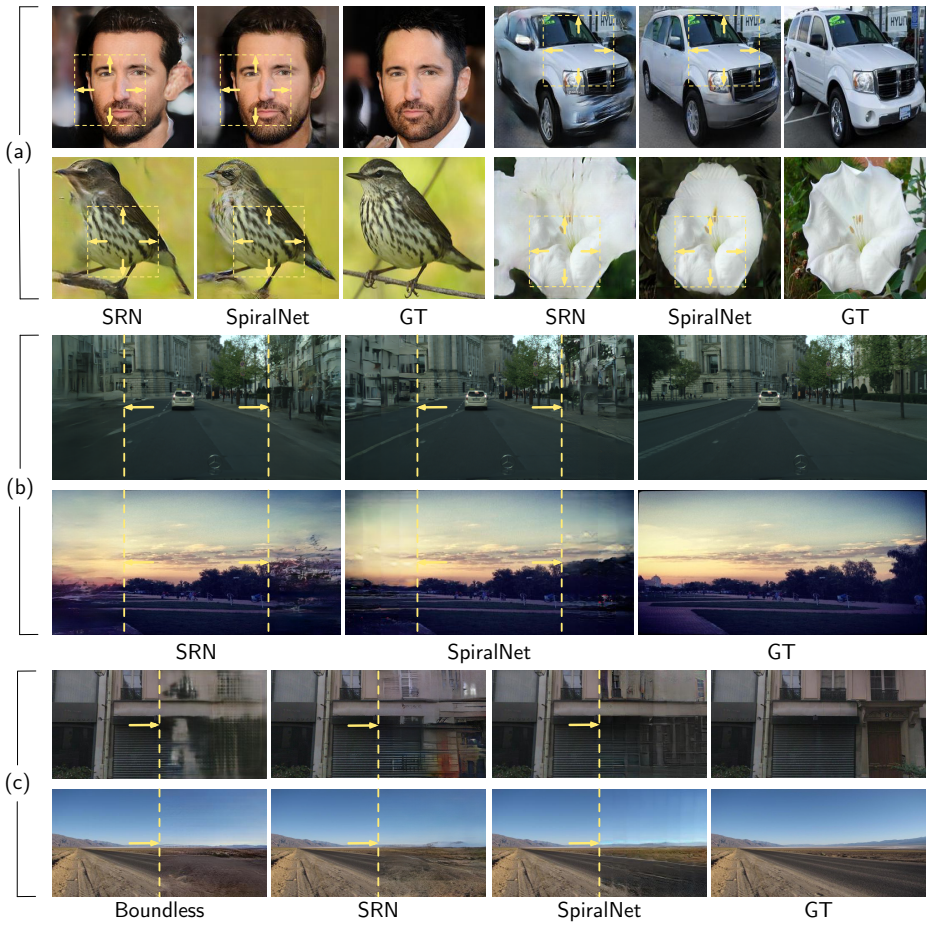

Fig. 4. Qualitative comparison results in different cases. (a) Four-side on CelebA-HQ, Stanford Cars, CUB and Flowers. (b) Two-side on Cityscapes and Place365 Sky. (c) One-side on Paris StreetView and Place365 Desert Road.

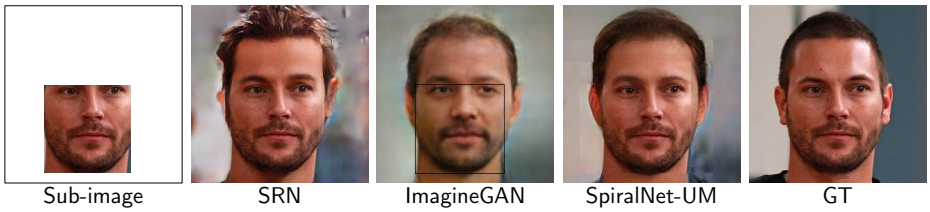

Fig. 5. Case of unknown margin on CelebA-HQ.

inputs. The users are required to select the more realistic image in each pair, and they are given unlimited time to make the decision. Each pair is judged by at least 3 different users. The results shown in Table 2 validate that our SpiralNet performs better than Boundless and SRN on all available datasets.

4.2 Qualitative Comparison

We also show qualitative comparison of Boundless, SRN and our SpiralNet in Fig. 4. Our method extrapolates more reasonable results with semantic consistency and vivid details avoiding meaningless content and cluttered background. Moreover, Fig. 5 and Table 3 (CelebA-HQ) shows that our SpiralNet-UM also works well. More results are shown in *supplementary file* for further reference.

4.3 Why Spiral Is Necessary

Our spiral architecture is necessary for tasks like image extrapolation conditioned on three aspects below, with various cases of ablation study for validating necessity of each one (results are shown in Table 4 and Fig. 6):
A. turn-by-turn extrapolation: (1) one-by-one directional extrapolation (A.one-by-one); (2) horizontal-then-vertical directional extrapolation (A.horizontal-vertical); and (3) vertical-then-horizontal directional extrapolation (A.vertical-horizontal). One exemplar in Fig. 6 shows that, destroying the equilibrium of slice growth in four directions, will lead to inharmonious generators for horizontal small slices and vertical large slices, yielding semantic inconsistency with cluttered content.
B. dependency of directional slices in adjacent turns: (1) without closest slice input (B.w/o closest slice); and (2) replace closest slice with sub-image slice (B.w/ sub-image slice). Figs. 6d and 6e display blurry details and unrealistic textures on parts far away from original sub-image region.
C. correlation between adjacent slices in one turn: (1) generate four directional slices simultaneously (C.simultaneous); (2) horizontal-then-vertical slice generation (C.horizontal-vertical); and (3) vertical-then-horizontal slice generation (C.vertical-horizontal). Figs. 6f, 6g and 6h illustrate that some slice corners are influenced by noncontinuous slice generation in one turn.
While SpiralNets with anticlockwise or clockwise (default) slice generation in one turn, perform better both quantitatively (Table 4) and qualitatively (Fig. 6), indicating that it's effective to do image extrapolation in our spiral way.

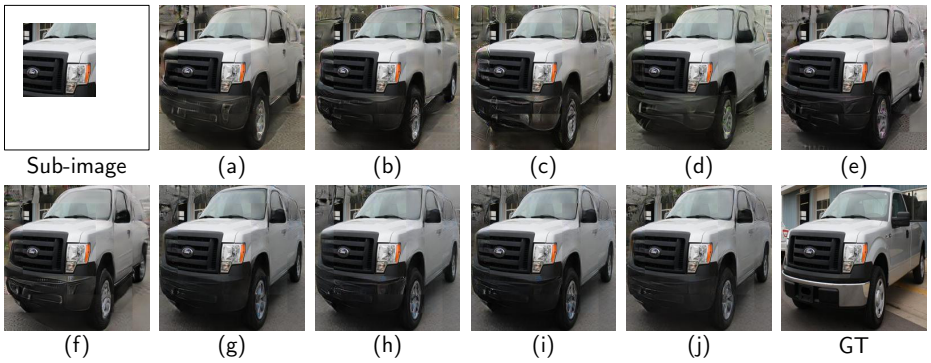

Fig. 6. Qualitative results on why spiral is necessary. (a) A.one-by-one. (b) A.horizontal-vertical. (c) A.vertical-horizontal. (d) B.w/o closest slice. (e) B.w/ sub-image slice. (f) C.simultaneous. (g) C.horizontal-vertical. (h) C.vertical-horizontal. (i) SpiralNet.anticlockwise. (j) SpiralNet.clockwise. Please zoom in for better comparison.

Table 4. Quantitative results on ablation study of why spiral is necessary.

Method	PSNR	SSIM	FID
A.one-by-one	14.04	0.5724	23.80
A.horizontal-vertical	14.12	0.5697	24.00
A.vertical-horizontal	14.09	0.5661	26.79
B.w/o closest slice	14.02	0.5652	24.41
B.w/ sub-image slice	14.08	0.5662	22.77
C.simultaneous	14.18	0.5708	**21.96**
C.horizontal-vertical	14.22	0.5741	24.47
C.vertical-horizontal	14.25	0.5756	24.09
SpiralNet.anticlockwise	14.27	0.5753	24.20
SpiralNet.clockwise	**14.31**	**0.5775**	23.64

Table 5. Quantitative results on ablation study of ternary SliceGAN inputs.

Method	PSNR	SSIM	FID
Baseline	13.95	0.5683	26.10
w/ sub-image	14.02	0.5652	24.41
w/ closest slice	14.18	0.5727	24.64
exchange imaginary & closest slices	13.98	0.5743	26.86
SpiralNet	**14.31**	**0.5775**	**23.64**

Table 6. Quantitative results on ablation study of different slice sizes for SliceGAN.

τ	4	8	16	32	64
PSNR	13.58	13.70	13.80	**14.31**	14.05
SSIM	0.5486	0.5566	0.5632	**0.5775**	0.5694
FID	27.11	26.14	**20.59**	23.64	21.07

4.4 Analysis of SliceGAN

Ternary SliceGAN Inputs. Our SliceGAN (see Section 3.2) is designed with a new Encoder-AdaIN-SPADE|Decoder structure, to encode *imaginary slice* (Encoder) into a latent space, and fuse style information from *sub-image* (AdaIN) with the latent code, then combine with semantic information from *closest slice* (SPADE) when decode composite latent code back to image space (Decoder), resulting in extrapolated slice with consistency of style, semantic, and context.

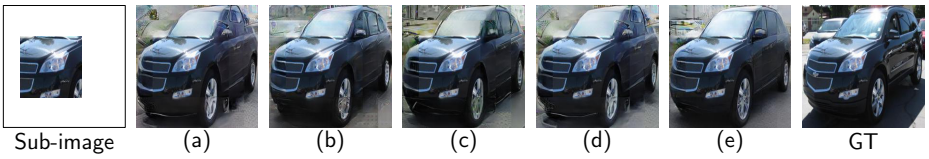

Fig. 7. Qualitative ablation study on ternary SliceGAN inputs. (a) Baseline. (b) w/ sub-image. (c) w/ closest slice. (d) exchange imaginary and closest slices. (e) SpiralNet.

Fig. 8. Qualitative ablation study on different slice sizes. (a) $\tau = 4$. (b) $\tau = 8$. (c) $\tau = 16$. (d) $\tau = 32$. (e) $\tau = 64$.

We thus conduct ablation study on the ternary SliceGAN inputs: imaginary slice, sub-image, and closest slice, for validating efficacy of them together with corresponding structures. We construct an Encoder-Decoder with the only input of imaginary slice as baseline, then add sub-image and closest slice with Encoder-AdaIN-Decoder and Encoder-SPADE|Decoder respectively for comparison. Besides, we also exchange imaginary slice and closest slice for further analysis.

Table 5 and Fig. 7 show results of ablation study on ternary SliceGAN inputs, which validate strength of our structure. Visually, the style between sub-image and generated slices looks more harmonious with sub-image input (Figs. 7a vs. 7b); and generated slices seems more semantically consistent with sub-image via closest slice input (Figs. 7a vs. 7c); also distorted content with inconsistent semantic appears if we exchange imaginary and closest slices (Fig. 7d); while SpiralNet improves over all the others thanks to our ternary inputs (Fig. 7e).

Different Slice Sizes. We then study the impact of slice size τ, and employ four different sizes of $\tau = \{4, 8, 16, 32, 64\}$ for ablation. Table 6 and Fig. 8 report the results, showing that, small slice size might introduce unclear texture (Fig. 8a), and big slice size may result in a more obvious stitching block effect (Fig. 8d). Considering effectiveness and efficiency, we set $\tau = 32$ for balance.

4.5 Efficacy of Hue-Color Loss

We finally analyze efficacy of hue-color loss. For convenience, we conduct ablative experiments using ImagineGAN on Flowers and Stanford Cars, by removing hue-color loss as baseline and replacing it with color loss [47] and L1 loss [17] for comparison. Results in Table 7 indicate that hue-color loss is more helpful in terms of PSNR, SSIM and FID. Figs. 9a and 9b illustrate step-by-step extrapolation in training with different losses, from which we observe that it appears

Table 7. Quantitative results on ablation study of ImagineGAN with different losses.

Model	Flowers			Stanford Cars		
	PSNR	SSIM	FID	PSNR	SSIM	FID
Baseline	10.49	0.0832	136.55	10.63	0.1005	120.33
w/ $\mathcal{L}^{\mathbf{I}}_{color}$	9.45	0.1294	250.30	10.75	0.2882	186.46
w/ $\mathcal{L}^{\mathbf{I}}_{L1}$	12.50	0.1266	109.36	11.78	0.2129	104.09
w/ $\mathcal{L}^{\mathbf{I}}_{hue}$	**14.98**	**0.4681**	**75.11**	**13.56**	**0.5107**	**53.12**

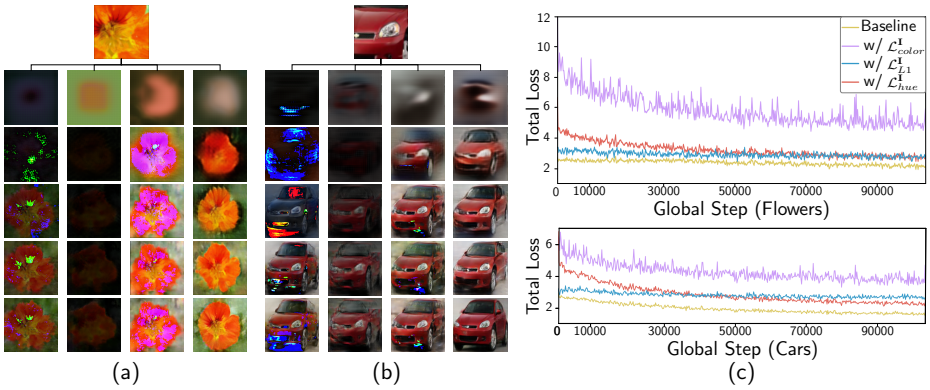

Fig. 9. (a) and (b): Qualitative results on Flowers and Stanford Cars: baseline, with color loss, with L1 loss, and with hue-color loss, from left to right. (c) Corresponding total loss curves on Flowers and Stanford Cars in training by steps.

dark and bright color spots exist in baseline, L1 loss and color loss may alleviate one of these issues, while our hue-color loss handle both issues very well (Figs. 9a and 9b). Corresponding loss curves in Fig. 9c demonstrate that, by using hue-color loss, total loss descends very quickly at the beginning, thereby the model identifies right color to stabilize training process.

5 Conclusion and Limitations

We propose a novel generative framework, SpiralNet, to extrapolate an image by evolving along a spiral curve, and input image grows a little bit slice in four directions after each spiral turn. With the help of our design, both extensive experiments and ablation study demonstrate the superiority of our method. However, it still has limitation that the results inevitably have trivial block effects. We hope to further explore SpiralNet for this limitation, as well as more extension in general image generation tasks.

Acknowledgements

This work was supported by the National Natural Science Foundation of China under Grants 61771440 and 41776113.

References

1. Arjovsky, M., Chintala, S., Bottou, L.: Wasserstein generative adversarial networks. In: ICML. pp. 214–223 (2017)
2. Avidan, S., Shamir, A.: Seam carving for content-aware image resizing. ACM TOG **26**(3), 10 (2007)
3. Barnes, C., Shechtman, E., Finkelstein, A., Goldman, D.B.: PatchMatch: A randomized correspondence algorithm for structural image editing. ACM TOG **28**(3), 24 (2009)
4. Brock, A., Donahue, J., Simonyan, K.: Large scale GAN training for high fidelity natural image synthesis. arXiv preprint arXiv:1809.11096 (2018)
5. Cho, W., Choi, S., Park, D.K., Shin, I., Choo, J.: Image-to-image translation via group-wise deep whitening-and-coloring transformation. In: CVPR. pp. 10639–10647 (2019)
6. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: CVPR. pp. 3213–3223 (2016)
7. Dekel, T., Gan, C., Krishnan, D., Liu, C., Freeman, W.T.: Sparse, smart contours to represent and edit images. In: CVPR. pp. 3511–3520 (2018)
8. Doersch, C., Singh, S., Gupta, A., Sivic, J., Efros, A.A.: What makes paris look like paris? ACM TOG **31**(4), 101 (2012)
9. Efros, A.A., Leung, T.K.: Texture synthesis by non-parametric sampling. In: CVPR. pp. 1033–1038 (1999)
10. Fairchild, M.D.: Color appearance models. John Wiley & Sons (2013)
11. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: NIPS. pp. 2672–2680 (2014)
12. Gulrajani, I., Ahmed, F., Arjovsky, M., Dumoulin, V., Courville, A.C.: Improved training of wasserstein GANs. In: NIPS. pp. 5767–5777 (2017)
13. Hanbury, A.: Constructing cylindrical coordinate colour spaces. Pattern Recognition Letters **29**(4), 494–500 (2008)
14. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR. pp. 4700–4708 (2017)
15. Huang, X., Belongie, S.: Arbitrary style transfer in real-time with adaptive instance normalization. In: ICCV. pp. 1501–1510 (2017)
16. Huang, X., Liu, M.Y., Belongie, S., Kautz, J.: Multimodal unsupervised image-to-image translation. In: ECCV. pp. 172–189 (2018)
17. Isola, P., Zhu, J.Y., Zhou, T., Efros, A.A.: Image-to-image translation with conditional adversarial networks. In: CVPR. pp. 1125–1134 (2017)
18. Johnson, J., Alahi, A., Fei-Fei, L.: Perceptual losses for real-time style transfer and super-resolution. In: ECCV. pp. 694–711 (2016)
19. Karras, T., Aila, T., Laine, S., Lehtinen, J.: Progressive growing of GANs for improved quality, stability, and variation. In: ICLR (2018)
20. Karras, T., Laine, S., Aila, T.: A style-based generator architecture for generative adversarial networks. In: CVPR. pp. 4401–4410 (2019)
21. Kihlstrom, J.F.: The cognitive unconscious. Science **237**(4821), 1445–1452 (1987)
22. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization. In: ICLR (2015)
23. Kosslyn, S.M., Ganis, G., Thompson, W.L.: Neural foundations of imagery. Nature Reviews Neuroscience **2**(9), 635 (2001)

24. Krause, J., Stark, M., Deng, J., Fei-Fei, L.: 3D object representations for fine-grained categorization. In: ICCVW. pp. 554–561 (2013)
25. Lee, D., Kim, J., Moon, W.J., Ye, J.C.: CollaGAN: Collaborative GAN for missing image data imputation. In: CVPR. pp. 2487–2496 (2019)
26. Li, C., Wand, M.: Precomputed real-time texture synthesis with markovian generative adversarial networks. In: ECCV. pp. 702–716 (2016)
27. Liu, G., Reda, F.A., Shih, K.J., Wang, T.C., Tao, A., Catanzaro, B.: Image inpainting for irregular holes using partial convolutions. In: ECCV. pp. 85–100 (2018)
28. MacEvoy, B.: Color Vision. handprint.com (2010)
29. Mao, X., Li, Q., Xie, H., Lau, R.Y., Wang, Z., Paul Smolley, S.: Least squares generative adversarial networks. In: ICCV. pp. 2794–2802 (2017)
30. Mirza, M., Osindero, S.: Conditional generative adversarial nets. arXiv preprint arXiv:1411.1784 (2014)
31. Miyato, T., Kataoka, T., Koyama, M., Yoshida, Y.: Spectral normalization for generative adversarial networks. In: ICLR (2018)
32. Nazeri, K., Ng, E., Joseph, T., Qureshi, F., Ebrahimi, M.: EdgeConnect: Generative image inpainting with adversarial edge learning. arXiv preprint arXiv:1901.00212 (2019)
33. Nilsback, M.E., Zisserman, A.: Automated flower classification over a large number of classes. In: ICVGIP. pp. 722–729 (2008)
34. Park, T., Liu, M.Y., Wang, T.C., Zhu, J.Y.: Semantic image synthesis with spatially-adaptive normalization. In: CVPR. pp. 2337–2346 (2019)
35. Pérez, P., Gangnet, M., Blake, A.: Poisson image editing. ACM TOG **22**(3), 313–318 (2003)
36. Pessoa, L., Thompson, E., Noë, A.: Finding out about filling-in: A guide to perceptual completion for visual science and the philosophy of perception. Behavioral and brain sciences **21**(6), 723–748 (1998)
37. Radford, A., Metz, L., Chintala, S.: Unsupervised representation learning with deep convolutional generative adversarial networks. In: ICLR (2016)
38. Sajjadi, M.S., Scholkopf, B., Hirsch, M.: EnhanceNet: Single image super-resolution through automated texture synthesis. In: ICCV. pp. 4491–4500 (2017)
39. Sarvaiya, J.N., Patnaik, S., Bombaywala, S.: Image registration by template matching using normalized cross-correlation. In: ICACCTT. pp. 819–822. IEEE (2009)
40. Shan, Q., Curless, B., Furukawa, Y., Hernandez, C., Seitz, S.M.: Photo uncrop. In: ECCV. pp. 16–31 (2014)
41. Simonyan, K., Zisserman, A.: Very deep convolutional networks for large-scale image recognition. arXiv preprint arXiv:1409.1556 (2014)
42. Slossberg, R., Shamai, G., Kimmel, R.: High quality facial surface and texture synthesis via generative adversarial networks. In: ECCV. pp. 498–513 (2018)
43. Teterwak, P., Sarna, A., Krishnan, D., Maschinot, A., Belanger, D., Liu, C., Freeman, W.T.: Boundless: Generative adversarial networks for image extension. In: ICCV. pp. 10521–10530 (2019)
44. Wah, C., Branson, S., Welinder, P., Perona, P., Belongie, S.: The Caltech-UCSD Birds-200-2011 Dataset. Tech. Rep. CNS-TR-2011-001, California Institute of Technology (2011)
45. Wang, M., Lai, Y., Liang, Y., Martin, R.R., Hu, S.M.: BiggerPicture: data-driven image extrapolation using graph matching. ACM TOG **33**(6), 173 (2014)
46. Wang, N., Li, J., Zhang, L., Du, B.: MUSICAL: multi-scale image contextual attention learning for inpainting. In: IJCAI. pp. 3748–3754 (2019)

47. Wang, R., Zhang, Q., Fu, C.W., Shen, X., Zheng, W.S., Jia, J.: Underexposed photo enhancement using deep illumination estimation. In: CVPR. pp. 6849–6857 (2019)
48. Wang, Y., Tao, X., Shen, X., Jia, J.: Wide-context semantic image extrapolation. In: CVPR. pp. 1399–1408 (2019)
49. Xian, W., Sangkloy, P., Agrawal, V., Raj, A., Lu, J., Fang, C., Yu, F., Hays, J.: TextureGAN: Controlling deep image synthesis with texture patches. In: CVPR. pp. 8456–8465 (2018)
50. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Generative image inpainting with contextual attention. In: CVPR. pp. 5505–5514 (2018)
51. Yu, J., Lin, Z., Yang, J., Shen, X., Lu, X., Huang, T.S.: Free-form image inpainting with gated convolution. In: ICCV. pp. 4471–4480 (2019)
52. Zhang, Y., Xiao, J., Hays, J., Tan, P.: FrameBreak: Dramatic image extrapolation by guided shift-maps. In: CVPR. pp. 1171–1178 (2013)
53. Zhou, B., Lapedriza, A., Khosla, A., Oliva, A., Torralba, A.: Places: A 10 million image database for scene recognition. IEEE TPAMI pp. 1452–1464 (2017)
54. Zhu, J.Y., Krähenbühl, P., Shechtman, E., Efros, A.A.: Generative visual manipulation on the natural image manifold. In: ECCV. pp. 597–613 (2016)
55. Zhu, J.Y., Park, T., Isola, P., Efros, A.A.: Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV. pp. 2223–2232 (2017)