A. Additional Figures

Here, we show additional qualitative results.

Successive translations In the main text, we discussed that sinusoidal embedding enables non-integer coordinates. Thus MS-PE generates consistent shapes at locations successively shifted by a pixel. Fig. A show digits generated at locations shifted vertically and horizontally.

Effect of 2D noise SS-PE rely on 2D noise input of StyleGAN [4, 5], as shown in Fig.4 of the main text. Fig. B exhibits additional results with models trained on FFHQ [4]. In Fig. B(b), the hairstyle is modified by a 2D noise map and shows the largest standard deviation.

GAN Inversion Our method facilitates robust GAN inversion. Additional GAN inversion results are presented in Fig. C.

Multi-scale generation MS-PE is effective in multi-scale generation with a single model. To further improve visual quality, we randomly resized (MS-PE w/ Random Resizing) the explicit positional encoding at each training iteration. Additional samples with "MS-PE w/ Random Resizing" are presented in Fig. D.

DDPM Reconstruction In the main text, we demonstrated the appliance of our method to denoising diffusion probabilistic models [3, 10]. Fig. E shows additional reconstruction results.

B. Detailed Introduction to DDPM

Here, we provide an additional description on Denoising Diffusion Probabilistic Models (DDPM) [3, 10]. DDPM consists of two processes: fixed *diffusion process* and learned *reverse process*. The diffusion process is a sequence that adds Gaussian noise to an image x_0 with a fixed variance schedule $\beta_1, ..., \beta_T$. Latent variables $x_1, ..., x_T$ are sampled from the diffusion process:

$$q(x_t|x_{t-1}) := N(x_t; \sqrt{1-\beta_t}x_{t-1}, \beta_t\mathbf{I}). \quad \text{(A)}$$

To generate images from random Gaussian noise, DDPM learns the reverse of the diffusion process, which is also a sequence of Gaussian translation:

$$p_\theta(x_{t-1}|x_t) = N(x_{t-1}; \mu_\theta(x_t, t), \sigma_t^2\mathbf{I}). \quad \text{(B)}$$

Here, variance is a fixed constant [3] and the mean is learned with a neural network θ. Here, θ is a fully-convolutional U-Net [9] with input and output of the same dimensionality. As diffusion (Eq. A) and reverse (Eq. B) process are both Gaussian translations, they are stochastic. Sec.4.5 presents stochastic reconstruction where encoding (diffusion) process and decoding (reverse) process are both stochastic.

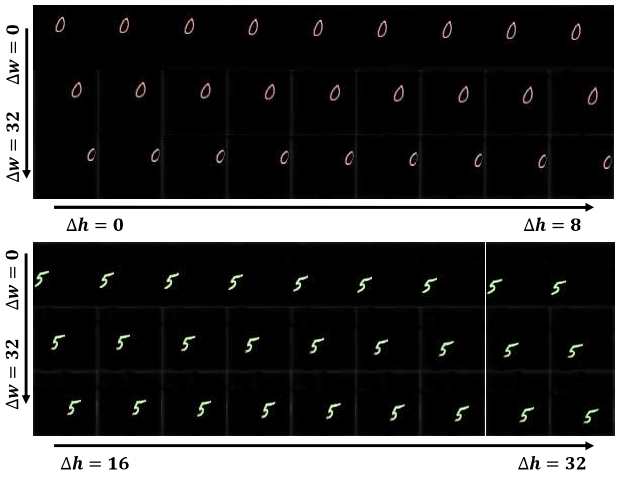

Figure A: **Generation at successively shifted locations.** We generated digits at locations shifted vertically by $\triangle h$ and horizontally by $\triangle w$, sharing the same inputs. Digits show consistent shapes.

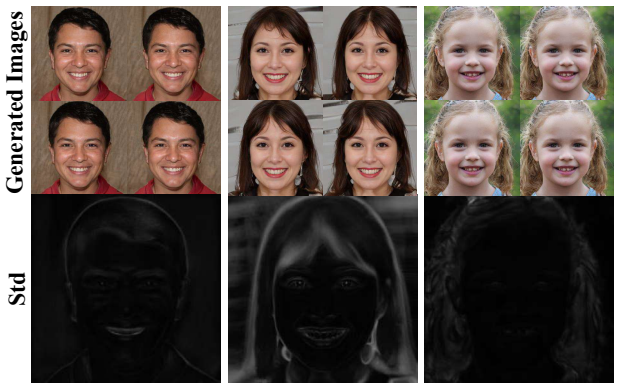

Figure B: **Effect of 2D noise input.** Standard deviation of each pixel over 100 2D noise map instances. SS-PE generates inconsistent hairstyles and larger standard deviations compared to the baseline and MS-PE.

C. Implementation Details

C.1. Dataset

Color-MNIST is a spatially biased dataset we customized for toy experiments presented in Fig.2, Fig.4, Fig.9 of the main text. The dataset consists of 60,000 images at 64×64 resolution. Digits range from 0 to 9, and their numbers are uniform. Digits are located in the 32×32 patch at the upper-left corner.

Flickr Faces HQ [4] consists of 70,000 high-quality face images crawled from Flickr. The images are carefully aligned [6], thus exhibits strong location bias.

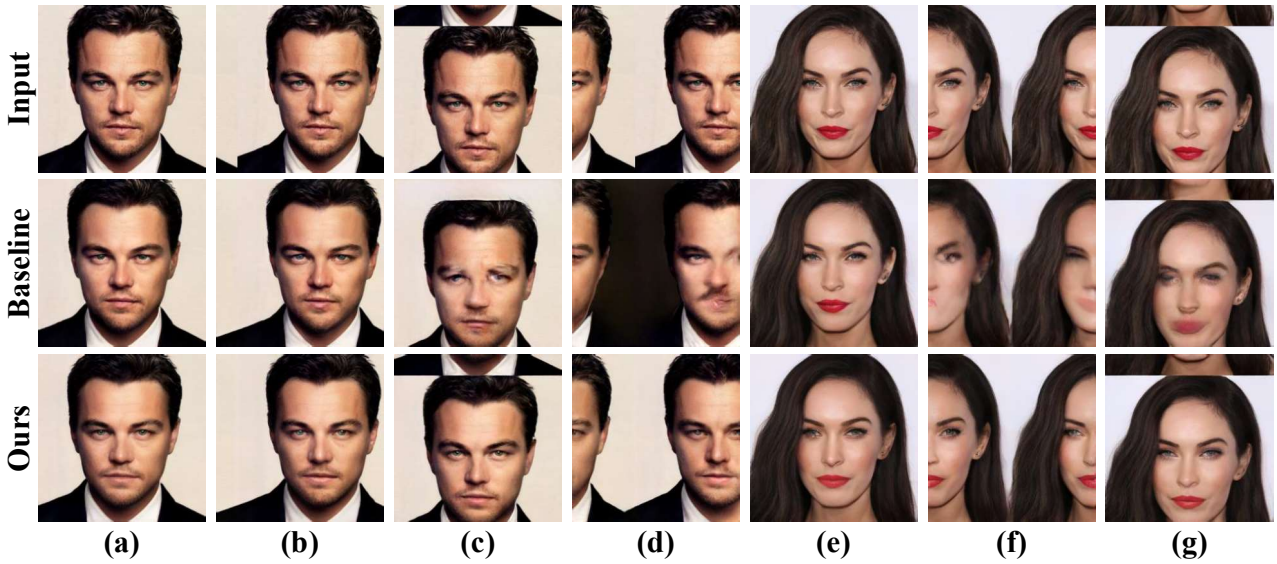

Figure C: **GAN Inversion.** (a) Standard position. (b) Translation of 32 pixels to the right. (c) Translation of 32 pixels to the bottom. (d) Translation of 128 pixels to the right. (e) Standard position. (f) Translation of 128 pixels to the right. (g) Translation of 32 pixels to the bottom. We use circular shift (roll) for translations.

Figure D: **Multi-scale generation.** 256^2, 384^2, 512^2 resolution images generated with a single model designed for 256^2 . Discriminator has seen images at 256^2 resolution only.

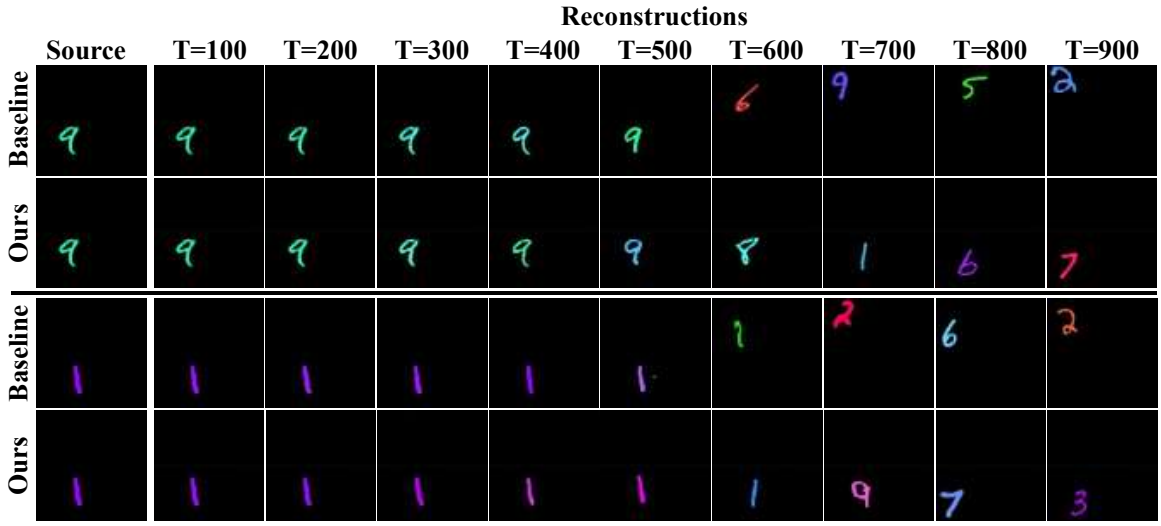

Figure E: **DDPM Reconstruction.** Given source images are encoded to various time steps (T=100 to 900) then decoded with a learned reverse process. Our method preserves location of the given digits.

LSUN Church [13] consists of 126,227 images of churches. Center cropped 256×256 images are used for training models.

C.2. Architecture

StyleGAN2 For experiments except Sec.4.5 of the main text, we used StyleGAN2 [5] trained at 256×256 resolution and inherited most of the architecture details of StyleGAN2. The details include weight demodulation, bilinear up/down sampling [14], noise injection, skip/residual connection in generator/discriminator, equalized learning rate, leaky ReLU activation with slope 0.2, and minibatch standard deviation at the discriminator. These setups correspond to the "baseline" in the main text. We replaced the constant tensor with 2D sinusoidal positional embedding [11]. We also added scale-specific 2D sinusoidal positional embedding at each scale, as described in Eq.3 of the main text. Therefore seven positional encodings in total (from 4×4 to 256×256).

DDPM In Sec.4.5 of the main text, we inherit the architecture details of DDPM [3], including U-Net [9] architecture, group normalization [12], self-attention blocks, linear β_t schedule, and sinusoidal embedding to indicate time step. Our 64×64 model use four feature map resolutions (64×64 to 8×8) and self-attention blocks at 8×8 resolution. We added 2D sinusoidal positional encoding after residual blocks at downscaling layers and after upsampling layers.

C.3. Training

StyleGAN2 For every configuration, we trained models for 6.4M images with batch size 32. We used non-saturating loss [1] with R_1 regularization [8]. We used only random horizontal flip for data augmentation. We used ADAM [7] optimizer with $\beta_1 = 0, \beta_2 = 0.99$. For MS-PE with Random Resizing in Sec.4.2, we randomly selected resolution from $\{256^2, 320^2, 384^2, 448^2, 512^2\}$ with uniform probabilities at each iteration.

DDPM We trained DDPM on our color-MNIST for 9.6M images with batch size 16. We did not use any data augmentations. We used ADAM [7] optimizer with $\beta_1 = 0, \beta_2 = 0.99$.

C.4. Evaluation

Similarity metric In Fig.2(d) of the main text, we measured the similarity of digits during successive translations. To compare the original digit with the shifted digit, we crop 32×32 patch A and B, as shown in Fig F. We then convert them to grayscale and measure similarity as follows:

$$sim = \sum_{i,j} min(A_{i,j}, B_{i,j}) / \sum_{i,j} max(A_{i,j}, B_{i,j}), \quad \text{(C)}$$

where i, j are spatial indexes. This metric is a continuous relaxation of mean Intersection over Union (mIoU).

Shift in the constant tensor. Fig.2(d) of the main text presents successive translations by a pixel. As described in Sec.3.4, a single-pixel shift in the image space corresponds to a 2^{1-L} shift in the constant tensor of the baseline StyleGAN (L-scale). To implement a non-integer shift in the constant tensor, we interpolated features of nearest integer coordinates, as shown in Fig. G.

Fréchet Inception Distance (FID) [2] We calculated FID scores on 50,000 real images and 50,000 generated images using code[1] of the PyTorch framework.

[1] https://github.com/mseitzer/pytorch-fid

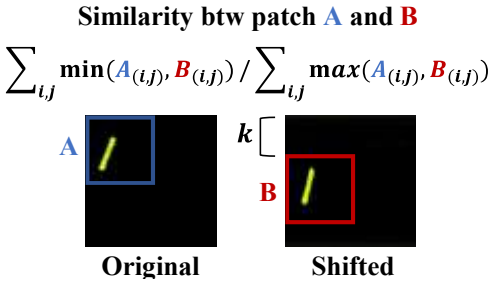

Figure F: **Measuring similarity of patches.** It is a continuous relaxation of mIoU. i, j are spatial indexes where $0 \leq i, j \leq 31$.

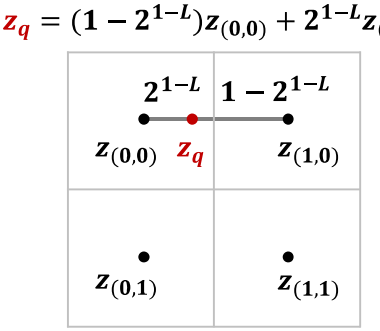

Figure G: **Interpolation at the constant tensor.** To implement non-integer shift in the constant tensor of baseline StyleGAN, we interpolated features at nearest integer coordinates. Zs denote features at each coordinate. We implement a one pixel shift of image by replacing $z_{(0,0)}$ with z_q.

References

[1] Ian J Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks. *arXiv preprint arXiv:1406.2661*, 2014. 3

[2] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *arXiv preprint arXiv:1706.08500*, 2017. 3

[3] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *arXiv preprint arXiv:2006.11239*, 2020. 1, 3

[4] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *CVPR*, 2019. 1

[5] Tero Karras, Samuli Laine, Miika Aittala, Janne Hellsten, Jaakko Lehtinen, and Timo Aila. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8110–8119, 2020. 1, 3

[6] Vahid Kazemi and Josephine Sullivan. One millisecond face alignment with an ensemble of regression trees. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1867–1874, 2014. 1

[7] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. 3

[8] Lars Mescheder, Andreas Geiger, and Sebastian Nowozin. Which training methods for gans do actually converge? In *International conference on machine learning*, pages 3481–3490. PMLR, 2018. 3

[9] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015. 1, 3

[10] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International Conference on Machine Learning*, pages 2256–2265. PMLR, 2015. 1

[11] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention is all you need. *arXiv preprint arXiv:1706.03762*, 2017. 3

[12] Yuxin Wu and Kaiming He. Group normalization. In *Proceedings of the European conference on computer vision (ECCV)*, pages 3–19, 2018. 3

[13] Fisher Yu, Ari Seff, Yinda Zhang, Shuran Song, Thomas Funkhouser, and Jianxiong Xiao. Lsun: Construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 3

[14] Richard Zhang. Making convolutional networks shift-invariant again. In *International Conference on Machine Learning*, pages 7324–7334. PMLR, 2019. 3