High Perceptual Quality Image Denoising with a Posterior Sampling CGAN
Supplementary Material

Guy Ohayon, Technion
Theo Adrai, Technion
Gregory Vaksman, Technion
Michael Elad, Google Research
Peyman Milanfar, Google Research

Appendices

A. Generator (Denoiser) Architecture and Full Framework Schematic

Inspired by StyleGAN2 [5] and UNet [8], our conditional generator (denoiser) is an encoder-decoder deep neural network, as shown in Figure 5. The decoder builds the output image scale by scale by adding residual information at each stage to the up-sampled RGB output of the previous one. This approach was proposed in StyleGAN2 to sidestep the shortcomings of progressive growing [4, 5], a training methodology in which the number of layers in the generator and the discriminator increases during training. In the newly proposed scheme, both are trained end-to-end with all resolutions included, which significantly eases the training procedure but still enforces the decoder to progressively synthesize the output image stage-by-stage by adding resolution-specific details at each level. The encoder is analogous to a drip irrigation system. It consists of a main pipeline that has several exits to independent convolutional neural networks (CNNs), denoted as *Drips*, each of which reinforces its neighboring decoder block with low receptive field information. The main pipeline is a deep CNN with high receptive field that progressively encodes the input image. This method alleviates the task of the decoder, especially at higher scales where original pixel locations are crucial for distortion performance.

Since our denoiser is stochastic and meant to sample from the posterior distribution, it can be considered as a mapping from the latent distribution of the random noise to the posterior, as in all GAN based sampling solutions. Instead of injecting a single noisy tensor to the first layer of our model, we inject noise at each scale of the decoder. These convolutional layers operate as follows: for a given layer with c input channels $\{x_i\}_{i=1}^{c}$ and a random noise input z of the same size, the resulting output of the layer (before the activation function is applied) is

$$\sum_{i=1}^{c} h_i * x_i + h_{c+1} * z, \tag{8}$$

where $\{h_i\}_{i=1}^{c+1}$ are the convolutional kernels of the layer (considering only one block of kernels that leads to one output feature map). If one further assumes that $h_{c+1} = \alpha$ (a 1-by-1 kernel) for some scaling factor α, this boils down to the noise injection scheme of StyleGAN [6] (each resulting feature map corresponds to a different scaling factor). In addition, if one forces the scaling factors of all feature maps to be equal, this becomes the noise injection scheme of StyleGAN2 [5]. Thus, our scheme enlarges the hypothesis set of the convolution operating on z. We incorporate this idea by concatenating each noise injection to the next convolutional layer's input. Consult Figure 5 for clarifications on the denoiser's architecture, and Figure 6 for a full framework schematic diagram.

B. Latent Adversarial Generator Implementation Details

Expression (6) is the originally proposed optimization task of LAG [2].Yet, the SISR results presented in LAG's paper are achieved with a tweaked version,

$$\min_{\theta} \sup_{f\in L_1} \mathbb{E}_{\mathbf{x}}\left[f(\mathbf{x},0)\right] - \mathbb{E}_{\mathbf{g}_\theta,\mathbf{y}}\left[f(\mathbf{g}_\theta, R(\mathbf{g}_\theta,\mathbf{y}))\right] \tag{9}$$
$$+\lambda\mathbb{E}_{\mathbf{x},\mathbf{y}}\left[\|P(\mathbf{x},0) - P(G_\theta(0,\mathbf{y}), R(G_\theta(0,\mathbf{y}),\mathbf{y}))\|_2^2\right],$$

in which, instead of $\mathbf{y}$, the critic receives $R(\mathbf{g}_\theta, \mathbf{y})$ as a second input, the pixel-wise absolute difference between the degraded image $\mathbf{y}$ and the corresponding degraded version of the generated image $\mathbf{g}_\theta|\mathbf{y}$. Such a tweak could also be adopted in the case of image denoising, for instance by defining $R(\mathbf{g}_\theta, \mathbf{y})$ to be the absolute difference between $\mathbf{x}$, the clean source of $\mathbf{y}$, and the corresponding denoised image $\mathbf{g}_\theta|\mathbf{y}$. However, such a revision deviates the posterior sampling goal, since the Wasserstein-1 distance [1] would consequently measure the deviation between the distributions $\mathbb{P}_{\mathbf{x}}$ and $\mathbb{P}_{\mathbf{x}|R(\mathbf{g}_\theta,\mathbf{y})}$. We leave this, possibly beneficial, approach for future research.

In PSCGAN the critic receives $\mathbf{y}$ as its second input, and thus, for a fair comparison, we implement LAG in the same fashion instead of applying the aforementioned tweak. Our choice also aligns with optimization task (6) (since the critic

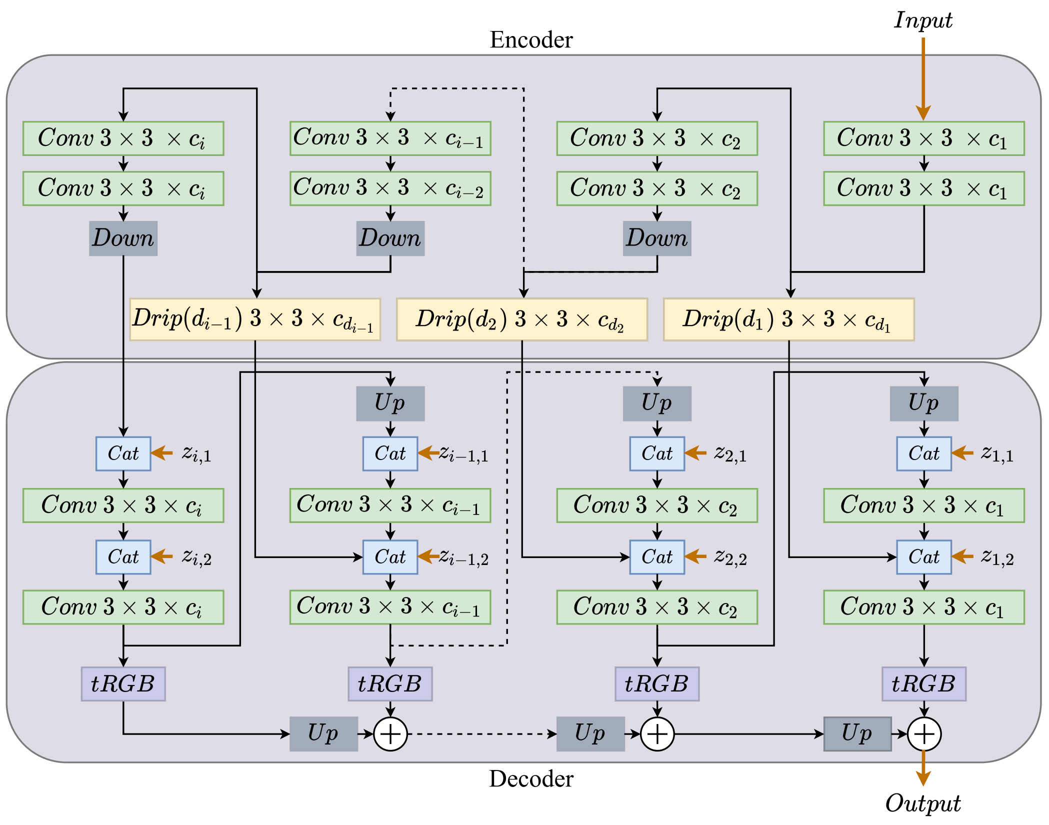

Figure 5: Our proposed generator architecture. An input noisy image is passed through an encoder of i doubly-blocked convolutional layers and downsampled after each (except for the first block). The downsampling operation is performed by a stride of 2 in the preceding convolution layer. The result of each doubly-blocked layer is then passed through a Drip, which is a feed-forward CNN (in the figure, d_k and c_{d_k} denote the number of layers and the number of output channels of each layer in drip k, respectively). Each of these drips extracts features that are limited to a certain receptive field, which are then passed to the neighboring decoder block through concatenation. This further assists in reconstructing the RGB result of the corresponding scale, especially at higher scales. The decoder builds the reconstructed image scale by scale, using features aggregated from previous layers of the decoder and from the drip injections. Noise injections are performed in the decoder's pipeline, where a noisy "image", denoted as $z_{k,1}$ and $z_{k,2}$ for each $1 \leq k \leq i$, is concatenated as another feature map of the next layer's input. All convolutional layers, except for *tRGB*, are coupled with Leaky ReLU activation functions with a slope of $\alpha = 0.2$ for negative values. *tRGB* is a simple convolution operation with output channels being equal to the number of channels of the input image (3 for RGB images). All up-sampling operations are performed with nearest-neighbor interpolation.

receives $\mathbf{y}$) and therefore leads to a more direct evaluation of it.

It is important to note that expression (6) is highly dependent on the choice of $P(\cdot,\cdot)$.While many choices are possible, we find that in our case choosing $P(x,y) = x$ leads to superior results, both in the FID and the PSNR performance measures. $P(x,y) = x$ means that we measure the distortion between $\mathbf{x}$ and $G_\theta(0,\mathbf{y})$ in expression (6) with the MSE loss, operating in the RGB pixel space of the image instead of operating in an intermediate feature space.While the sampled images attained at $\sigma_{\mathbf{z}} = 1$ should achieve higher perceptual quality (regardless of the choice of $P(\cdot,\cdot)$), this choice, quite conveniently, also allows for a fair PSNR comparison between PSCGAN and LAG, since the images produced by LAG at $\mathbf{z} = 0$ are now directly aimed to optimize the MSE. We refer to this version of LAG as *Ours-LAG* in the experimental evaluations, so as to emphasize that our choices deviate quite significantly from the original implementation of LAG (a different inverse problem, different architectures, and different loss).

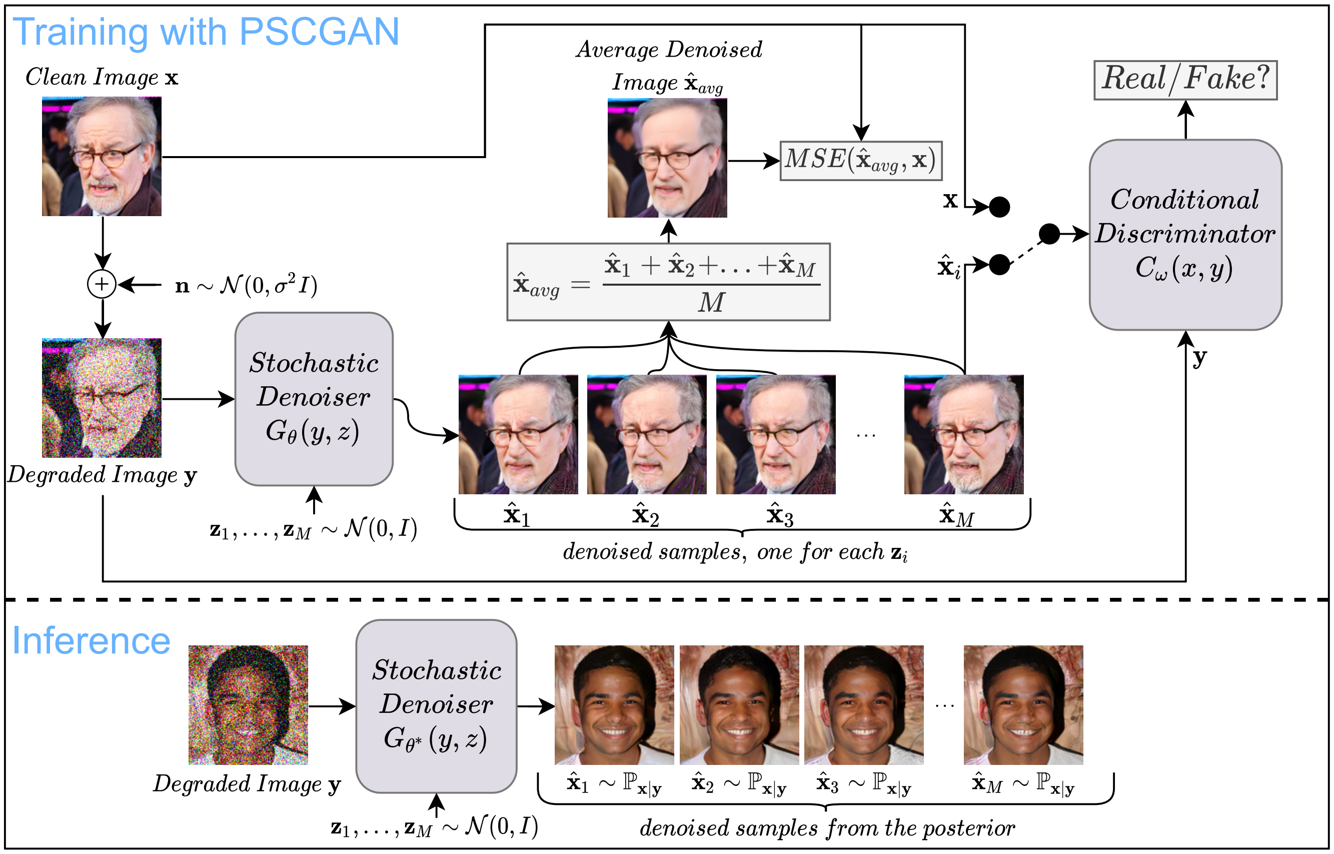
Training with PSCGAN
Clean Image $\mathbf{x}$
$\mathbf{n} \sim \mathcal{N}(0, \sigma^2 I)$
Degraded Image $\mathbf{y}$
Stochastic Denoiser $G_\theta(y, z)$
$\mathbf{z}_1, \ldots, \mathbf{z}_M \sim \mathcal{N}(0, I)$
Average Denoised Image $\hat{\mathbf{x}}_{avg}$
$MSE(\hat{\mathbf{x}}_{avg}, \mathbf{x})$
$\hat{\mathbf{x}}_{avg} = \frac{\hat{\mathbf{x}}_1 + \hat{\mathbf{x}}_2 + \ldots + \hat{\mathbf{x}}_M}{M}$
$\hat{\mathbf{x}}_1$ $\hat{\mathbf{x}}_2$ $\hat{\mathbf{x}}_3$ $\hat{\mathbf{x}}_M$
denoised samples, one for each $\mathbf{z}_i$
$\mathbf{x}$
$\hat{\mathbf{x}}_i$
$\mathbf{y}$
Conditional Discriminator $C_\omega(x, y)$
Real/Fake?
Inference
Degraded Image $\mathbf{y}$
Stochastic Denoiser $G_{\theta^*}(y, z)$
$\mathbf{z}_1, \ldots, \mathbf{z}_M \sim \mathcal{N}(0, I)$
$\hat{\mathbf{x}}_1 \sim \mathbb{P}_{\mathbf{x}|\mathbf{y}}$ $\hat{\mathbf{x}}_2 \sim \mathbb{P}_{\mathbf{x}|\mathbf{y}}$ $\hat{\mathbf{x}}_3 \sim \mathbb{P}_{\mathbf{x}|\mathbf{y}}$ $\hat{\mathbf{x}}_M \sim \mathbb{P}_{\mathbf{x}|\mathbf{y}}$
denoised samples from the posterior

Figure 6: A schematic diagram describing our proposed method at both training time and inference time. During training, the denoiser receives a noisy image $\mathbf{y}$ and outputs many possible denoised candidates $\hat{\mathbf{x}}_1, \hat{\mathbf{x}}_2, \ldots, \hat{\mathbf{x}}_M$. One of these candidates is being fed to a conditional discriminator, so as to drive the denoiser to produce images with high perceptual quality. At the same time, all of the denoised candidates are being averaged, and the result, $\hat{\mathbf{x}}_{avg}$, is forced to be close to the clean image $\mathbf{x}$ (in MSE). During inference, the denoiser receives a noisy input and produces many *different* denoised candidates with high perceptual quality, as many as the number of provided latent noises $\mathbf{z}_i$.

C. Training

C.1. Data Splits

- FFHQ [6] thumbnails contains 70,000 images. We use images 3000-4999 for testing and the rest for training.
- LSUN Bedroom [9] contains 3,037,042 images. We randomly pick 100,000 and 4,000 non-overlapping images for training and testing, respectively.
- LSUN Church [9] contains 126,227 images. We randomly pick 100,000 and 4,000 non-overlapping images for training and testing, respectively.

C.2. Preprocessing

Our model assumes an input image of size 128×128, and since the images in both LSUN data sets are of larger size in both axes, we first center crop each image while keeping the smaller dimension fixed, and then resize the resulting square image to the desired size through interpolation. All images in the FFHQ thumbnails data set are already of size 128×128, and therefore do not require augmentation. Finally, we use random horizontal flip during training in all data sets.

C.3. Hyperparameters

PSCGAN (and consequently PSCGAN-A) is trained with the default hyperparameters given in Algorithm 1.Note that we evaluate the penalty term on the first $PB = 8$ samples of each mini-batch of $B = 32$ samples, and approximate $\mathbb{E}\left[G_\theta(\mathbf{z}, \mathbf{y})|\mathbf{y}\right]$ by averaging $M = 8$ generated samples for a given noisy image $\mathbf{y}$. While this choice of M may seem too small to evaluate the expectation of the posterior, it is sufficient to demonstrate the novelty of PSCGAN.

All other algorithms are also trained with a batch size of 32 and the Adam optimizer [7]. LAG is trained with a learning rate of $2.5 \cdot 10^{-4}$, and the Adam hyperparameters are $\beta_1 = 0, \beta_2 = 0.99$ (similar to PSCGAN). DnCNN and Ours-MSE are trained to optimize the MSE loss, the former with a learning rate of 10^{-3} and the latter with a learning rate of $5 \cdot 10^{-4}$. The Adam hyperparameters for both methods are $\beta_1 = 0.9, \beta_2 = 0.99$.

The full implementation of all methods and the checkpoints that reproduce the results reported in this paper are publicly available. Our implementations are based on PyTorch and PyTorch Lightning [3].

D. LSUN Data Sets' Visual Results

In Figure 7 and Figure 8 we illustrate the visual quality of several denoised images produced by our method and by other MSE based methods on the LSUN Church outdoor and the LSUN Bedroom test sets. As can be seen, our model produces denoised images with high perceptual quality, although in these data sets it is harder to notice the perceptual quality difference with the naked eye (since the images are compressed).

References

[1] Martin Arjovsky, Soumith Chintala, and Léon Bottou. Wasserstein generative adversarial networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70, pages 214–223. PMLR, 2017. 1

[2] David Berthelot, Peyman Milanfar, and Ian Goodfellow. Creating high resolution images with a latent adversarial generator. *arXiv preprint arXiv:2003.02365*, 2020. 1

[3] WA Falcon et al. Pytorch lightning. https://github.com/PyTorchLightning/pytorch-lightning, 2019. 4

[4] Tero Karras, Timo Aila, Samuli Laine, and Jaakko Lehtinen. Progressive growing of gans for improved quality, stability, and variation. *arXiv preprint arXiv:1710.10196*, 2018. 1

[5] Tero Karras et al. Analyzing and improving the image quality of stylegan. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2020. 1

[6] Tero Karras, Samuli Laine, and Timo Aila. A style-based generator architecture for generative adversarial networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, June 2019. 1, 3

[7] Diederik P. Kingma and Jimmy Ba. Adam: a method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2017. 3

[8] O. Ronneberger, P.Fischer, and T. Brox. U-net: convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention*, volume 9351 of *LNCS*, pages 234–241. Springer, 2015. 1

[9] Fisher Yu et al. Lsun: construction of a large-scale image dataset using deep learning with humans in the loop. *arXiv preprint arXiv:1506.03365*, 2015. 3

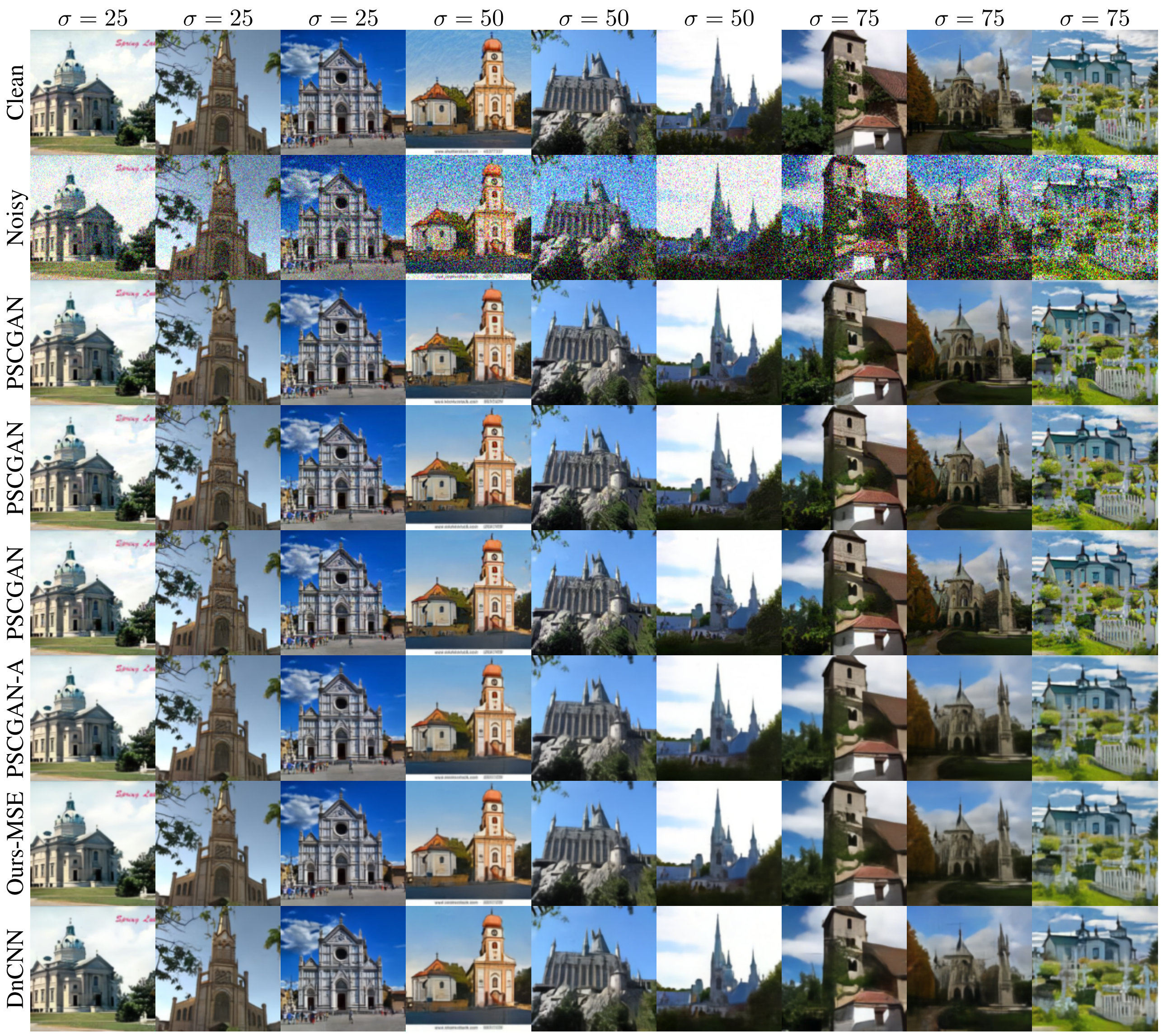

Figure 7: Denoising results on the LSUN Church outdoor test set produced by several methods. For each image we show three different outcomes of PSCGAN, each attained by injecting noise with standard deviation of $\sigma_{\mathbf{z}} = 0.75$. In this case, PSCGAN-A averages 64 instances of PSCGAN, where each instance is attained with $\sigma_{\mathbf{z}} = 1$ at inference time. Each model is trained on the LSUN Church outdoor training set to denoise a specific noise level $(25, 50$ or $75)$.

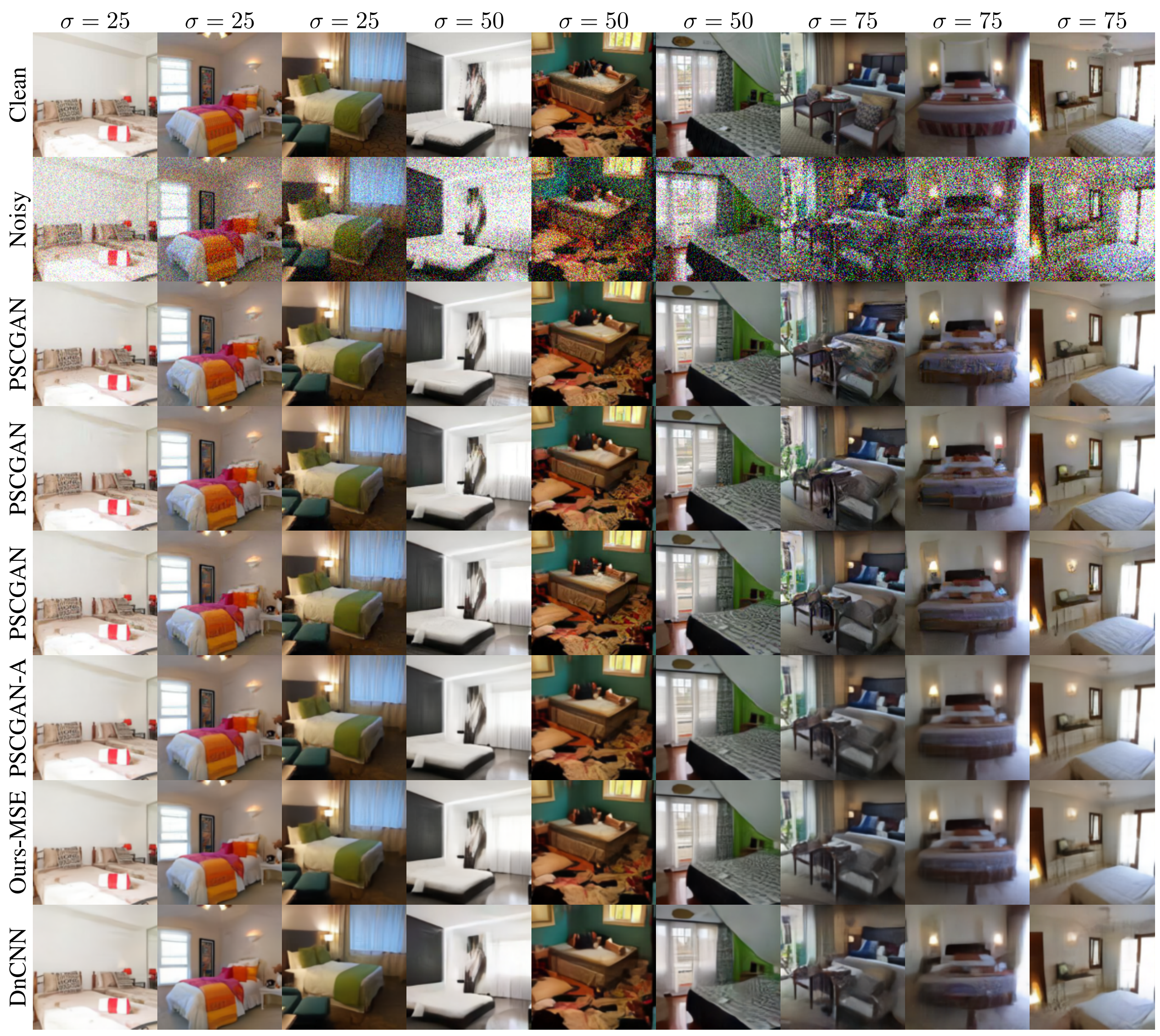

Figure 8: Denoising results on the LSUN Bedroom test set produced by several methods. For each image we show three different outcomes of PSCGAN, each attained by injecting noise with standard deviation of $\sigma_{\mathbf{z}} = 0.75$. In this case, PSCGAN-A averages 64 instances of PSCGAN, where each instance is attained with $\sigma_{\mathbf{z}} = 1$ at inference time. Each model is trained on the LSUN Bedroom training set to denoise a specific noise level $(25, 50$ or $75)$.