# SUPPLEMENTARY MATERIALS FOR GAN2GAN: GENERATIVE NOISE LEARNING FOR BLIND DENOISING WITH SINGLE NOISY IMAGE

**Sungmin Cha**[1], **Taeeon Park**[1], **Byeongjoon Kim**[2], **Jongduk Baek**[2] **and Taesup Moon**[3*]
Sungkyunkwan University[1], Yonsei University[2], Seoul National University[3], South Korea
{csm9493,pte1236}@skku.edu, bjkim2006@naver.com,
jongdukbaek@yonsei.ac.kr, tsmoon@snu.ac.kr

## 1 PROOF OF THEOREM 1

**Theorem 1** *Consider the single-letter Gaussian setting and $f_{\text{Noisy N2N}}(Z,y)$ obtained in (Eq.(2), manuscript). Also, assume $0 < y = \sigma_0^2/\sigma_X^2 < 1$. Then, there exists some $y_0$ s.t. $\forall y \in (y_0, 1)$, $\mathbb{E}(X - f_{\text{Noisy N2N}}(Z,y))^2 < \sigma_0^2$.*

*Proof:* We consider the following chain of equalities:

$$\begin{aligned}
& \sigma_0^2 - \mathbb{E}(X - f_{\text{Noisy N2N}}(Z,y))^2 && (1)\\
= \;& \sigma_0^2 - \mathbb{E}\Big(X - \frac{\sigma_X^2(1+y)}{\sigma_X^2(1+y)+\sigma_N^2}(X+N)\Big)^2 && (2)\\
= \;& \sigma_0^2 - \mathbb{E}\Big(\frac{\sigma_N^2}{\sigma_X^2(1+y)+\sigma_N^2}X - \frac{\sigma_X^2(1+y)}{\sigma_X^2(1+y)+\sigma_N^2}N\Big)^2 && (3)\\
= \;& \sigma_X^2\Big(y - \frac{\sigma_N^4}{(\sigma_X^2(1+y)+\sigma_N^2)^2} - \frac{\sigma_N^2\sigma_X^2(1+y)^2}{(\sigma_X^2(1+y)+\sigma_N^2)^2}\Big) && (4)\\
= \;& \sigma_X^2\frac{y(\sigma_X^2(1+y)+\sigma_N^2)^2 - \sigma_N^4 - \sigma_N^2\sigma_X^2(1+y)^2}{(\sigma_X^2(1+y)+\sigma_N^2)^2} && (5)
\end{aligned}$$

Now, by denoting the numerator of (5) as $g(y)$, we have

$$\begin{aligned}
g(y) &= y\big(\sigma_X^4(1+y)^2 + \sigma_N^4 + 2\sigma_X^2\sigma_N^2(1+y)\big) - \sigma_N^2\sigma_X^2(y^2+2y+1) - \sigma_N^4 && (6)\\
&= \sigma_X^4y^3 + (2\sigma_X^4 + \sigma_N^2\sigma_X^2)y^2 + (\sigma_X^4+\sigma_N^4)y - \sigma_N^2\sigma_X^2 - \sigma_N^4. && (7)
\end{aligned}$$

Then, we can easily see that

$$\begin{aligned}
g(0) &= -\sigma_N^2\sigma_X^2 - \sigma_N^4 < 0 && (8)\\
g(1) &= 4\sigma_X^4 > 0 && (9)\\
g(y) &= 3\sigma_X^4y^2 + 2(2\sigma_X^4 + \sigma_N^2\sigma_X^2)y + \sigma_X^4 + \sigma_N^4 > 0. && (10)
\end{aligned}$$

Therefore, in $0 < y < 1$, we can see $g(y)$ is an increasing function and has a root $y_0$ in the interval. Hence, the claim of the theorem: for all $y \in (y_0, 1)$, $\mathbb{E}(X - f_{\text{Noisy N2N}}(Z,y))^2 < \sigma_0^2$ holds. ■

## 2 DETAILS ON THE EXPERIMENTAL SETTINGS

### 2.1 SMOOTH NOISY PATCH EXTRACTION

#### 2.1.1 GCBD (CHEN ET AL., 2018) RULE EQ.(3)

The original GCBD paper (Chen et al., 2018) did not provide any source code or training data, hence, we reproduced their noisy patch extraction algorithm. There are six hyperparameters for the rule

*Corresponding author (E-mail: tsmoon@snu.ac.kr)

[Eq.(3), Manuscript], and we used the exact same hyperparameters given in their paper, which are shown in Table 1. $d$ and $h$ denote the size of a patch, $\boldsymbol{p}$, and its sub-patches, $\mathbf{q}_j$, given in [Eq.(3), Manuscript], respectively. $s_p$ and $s_q$ are the stride sizes for extracting the patches, $\boldsymbol{p}$ and $\{\mathbf{q}_j\}$, from a given image. $\mu$ and $\lambda$ are the hyperparameters of the rule for selecting the smooth patches shown in [Eq.(3), Manuscript].

Table 1: Hyperparameters for the patch extraction rule of GCBD

| Hyperparameters | $d$ | $h$ | $s_p$ | $s_q$ | $\mu$ | $\gamma$ |
|---|---|---|---|---|---|---|
| Values | 64 | 16 | 32 | 16 | 0.1 | 0.25 |

### 2.1.2 G2G RULE EQ.(4)

There are three hyperparemeters for our extraction rule [Eq.(4 ), Manuscript], $\lambda$, $d$ (the patch size), and $s_d$ (the stride size for extracting patches from an image). The choices for our experiments are shown in Table 3. Moreover, we stress that we did *not* tune $\lambda$ using clean images, but the different $\lambda$ values in the table are determined by the pre-determined number of extracted patches by applying our rule [Eq.(4), Manuscript]. Moreover, as argued in Section 3.1 (manuscript), we do not require any sub-patches to be extracted, hence, have only half the hyperparameters compared to the GCBD rule.

Table 2: Hyperparameters for the extraction rule of G2G

| | Gaussian Noise | Mixture Noise | Correlated Noise | WF | Medical |
|---|---|---|---|---|---|
| $\lambda$ | 0.03 | 0.1 | 0.15 | 0.42 | 0.015 |
| $d$ | 96 | | | | |
| $s_p$ | 24 | | | | |

### 2.1.3 EFFECT OF $\lambda$

Table 3 shows the effect of $\lambda$ in [Eq.(4), Manuscript] on the final performance of G2G$_2$. Note the smaller the $\lambda$, the less number of patches are extracted, but the homogeneity increases. The table shows $\lambda$ clearly affects the denoising performance of $g_{\boldsymbol{\theta}_2}$, but as the iterative G2G training continues, the performance of G2G$_2$ becomes not very sensitive to $\lambda$. Hence, in our experiments, we did not optimize $\lambda$ based on *any* clean validation set, but just set $\lambda$ based on the number of extracted patches and checking the visual qualities of the patches.

Table 3: Effects of varying $\lambda$ on the denoising performance.

| Gaussian Noise ($\sigma = 25$) | | | | Mixture Noise ($s = 25$) | | | | Correlated Gaussian Noise ($\sigma = 25$) | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| $\lambda$ | # of patches | $g_{\theta_2}$ | G2G$_2$ | $\lambda$ | # of patches | $g_{\theta_2}$ | G2G$_2$ | $\lambda$ | # of patches | $g_{\theta_2}$ | G2G$_2$ |
| 0.03 | 100,000 | 26.30/0.7123 | **28.93/0.8293** | 0.1 | 80,000 | 32.73/0.9478 | **40.30/0.9845** | 0.15 | 100,000 | 25.68/0.7606 | **27.85/0.8185** |
| 0.01 | 61,000 | 27.20/0.7159 | **28.84/0.8045** | 0.005 | 45,000 | 35.84/0.9588 | **40.16/0.9838** | 0.11 | 30,000 | 26.61/0.7566 | **27.67/0.8203** |
| 0.0075 | 32,000 | 26.44/0.7085 | **28.80/0.8060** | 0.025 | 23,000 | 34.20/0.9398 | **40.28/0.9848** | 0.1 | 11,000 | 26.30/0.7440 | **27.65/0.8203** |

## 2.2 TRAINING A W-GAN BASED GENERATIVE MODEL

Here, we elaborate a couple of subtle points for training our generative model as mentioned in Section 3.2 (manuscript).

Firstly, given the overall optimization objective [Eq.(8), manuscript], we use $(\alpha, \beta, \gamma) = (1, 1, 0)$ for the inner maximization for critics, and use $(\alpha, \beta, \gamma) = (5, 1, 10)$ for the outer minimization for generators. The main intuition for using different $(\alpha, \beta, \gamma)$ for training the generators is due to different levels of confidence in the generator loss terms. Namely, we assign the largest weight to [Eq.(7), Manuscript] since it is a deterministic loss and its value has a clear meaning. The generator loss [Eq.(5), Manuscript], which is in the form of the standard W-GAN loss, gets the medium level weight since the meaning of its value is less certain than [Eq.(6), Manuscript]. In contrast, the generator loss in [Eq.(5), Manuscript], which consists of two generators, can become somewhat unstable during training, hence, it gets the least weight. Figure 2.2 compares the performance of

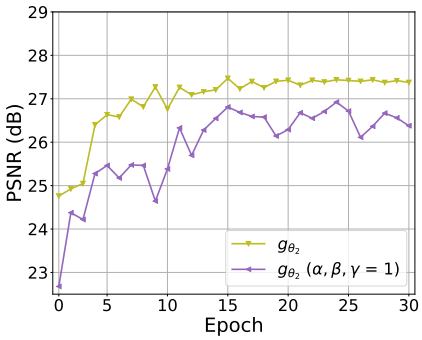



Figure 1: Ablation study on $(\alpha, \beta, \gamma)$

$g_{\boldsymbol{\theta}_2}$'s on BSD68 ($\sigma = 25$) when using $(\alpha, \beta, \gamma) = (5, 1, 10)$, for the outer minimization, as proposed, and using $(\alpha, \beta, \gamma) = (1, 1, 1)$. We observe there is a significant gap between the two.

Secondly, the output layer of $g_{\boldsymbol{\theta}_2}$ *must* have the sigmoid activation function. Note $g_{\boldsymbol{\theta}_2}$ itself can be thought of another denoiser, but since we are not training it with any target, we need to ensure the outputs of $g_{\boldsymbol{\theta}_2}$ have values between $[0, 1]$ to prevent from obvious errors of generating negative or out-of-bound pixel values. Without the sigmoid activation, it turned out all the generators cannot be trained properly at all.

Finally, using the right architectures for the generators and critics, *e.g.*, number of layers and filters, was critical since the training procedure got very sensitive to the architectural variations. Tables 4 shows the details on the architecture of our first generator, $g_{\boldsymbol{\theta}_1}$, which aims to generate noise patches. The dimension of $\boldsymbol{r}$ (the input random vector) was set to 128, and $C$ denotes the channel of the generated noise patch. The architectures of the $g_{\boldsymbol{\theta}_2}$ and $g_{\boldsymbol{\theta}_3}$ in our generative model are equal to that of the DnCNN model (Zhang et al., 2018), however, $g_{\boldsymbol{\theta}_2}$ had 15 layers with sigmoid activation in the output layer, and $g_{\boldsymbol{\theta}_3}$ had 17 layers and linear activation in the output layer. In addition, the architectures of the two critics, $\{f_{\boldsymbol{w}_1}, f_{\boldsymbol{w}_2}\}$, in our generative model are given in Table 5.

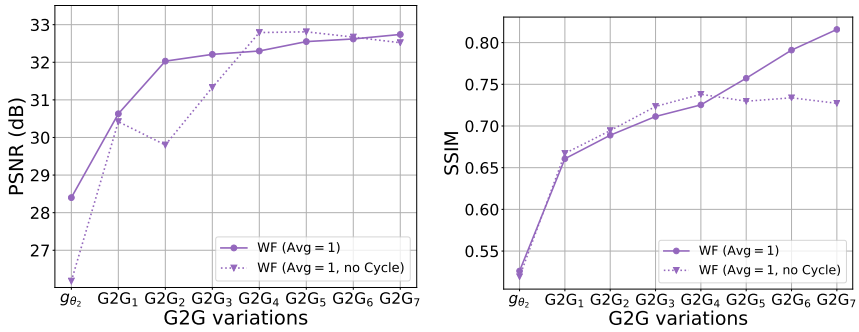
Table 4: Architectural details on $g_{\boldsymbol{\theta}_1}$.

| Input shape : (128, ) | | Details of DeConv layer | | | | |
|---|---|---|---|---|---|---|
| Layer Num | Layer composition | Input channel | Output channel | Kernel size | Stride | Padding |
| 1 | DeConv + BatchNorm + ReLU | 128 | 64 | 4 | 1 | 0 |
| 2 | DeConv + BatchNorm + ReLU | 64 | 32 | 4 | 2 | 1 |
| 3 | DeConv + BatchNorm + ReLU | 32 | 16 | 4 | 2 | 1 |
| 4 | DeConv + BatchNorm + ReLU | 16 | 8 | 4 | 1 | 1 |
| 5 | Conv + Tanh | 8 | $C$ | 4 | 2 | 1 |
| Output shape : (64x64x$C$) | | - | | | | |

Table 5: Architectural details on the critics, $\{f_{\boldsymbol{w}_1}, f_{\boldsymbol{w}_2}\}$.

| Input shape : (64x64x$C$) | | Details of Conv layer | | | | | Details of LeakyReLU |
|---|---|---|---|---|---|---|---|
| Layer Num | Layer composition | Input channel | Output channel | Kernel size | Stride | Padding | $\alpha$ |
| 1 | Conv + BatchNorm + LeakyReLU | C | 128 | 4 | 2 | 1 | 0.2 |
| 2 | Conv + BatchNorm + LeakyReLU | 128 | 256 | 4 | 2 | 1 | |
| 3 | Conv + BatchNorm + LeakyReLU | 256 | 512 | 4 | 2 | 1 | |
| 4 | Conv | 512 | 1 | 4 | 1 | 0 | - |
| Output shape : (64x64x1) | | - | | | | | - |

For training, we carry out the random cropping of the given patches to the size of $64 \times 64$, and the data augmentation was done by flipping the cropped patches horizontally and vertically. For optimization, we used Adam (Kingma & Ba, 2015) optimizer for the three generators and RMSProp (Tieleman & Hinton, 2012) optimizer for the two critics. The initial learning rates were set to $0.0004$ and $0.0005$ for Adam and RMSProp, respectively. Also, the learning rate decay, dropping the learning rate linearly starting from epoch 10, is applied to the Adam optimizer. The parameter clipping was done for the critics and the range was set to $[-0.02, 0.02]$, and the number of training iterations for the critics was 5. The total number of training epochs was 30 and the mini-batch size was 64.




(a) Iterative G2G (PSNR) (b) Iterative G2G (SSIM)

Figure 2: Figure (a) and (b) compares the PSNR and SSIM performances between starting from $g_{\boldsymbol{\theta}_2}$ and $g_{\boldsymbol{\theta}_2}$(No $\mathcal{L}_{cyc}$), respectively.

#### 2.2.1 Ablation study on $\mathcal{L}_{\text{cyc}}$

As shown in the synthetic noise case of Figure 6(b) (manuscript), the iterative G2G training is powerful such that there is a negligible performance difference between the schemes with and without $g_{\boldsymbol{\theta}_2}$, when the number of iterations is sufficiently large. Consequently, the cycle loss $\mathcal{L}_{cyc}$ also does not have significant effect in the final performance for the synthetic noise case. However, for the real noise case, $\mathcal{L}_{cyc}$ becomes more critical. As shown in Figure 2(a) and 2(b), on WF(Avg$=1$) dataset, we observe that when there is no $\mathcal{L}_{cyc}$ in our generative model, the final PSNR or SSIM performances cannot reach the model *with* $\mathcal{L}_{cyc}$ even after many iterations of G2G training. Hence, this result shows the necessity of $\mathcal{L}_{cyc}$.

### 2.3 Iterative GAN2GAN training of a denoiser

We do the same random cropping and data augmentation as in the generative model training. Moreover, for every minibatch in the G2G training, we generated new synthetic noisy image pairs using our trained generators as was done in the noise augmentation of (Zhang et al., 2017). Adam optimizer with an initial learning rate $0.001$ was used, and the learning rate scheduling, which halves the learning rate every 20 epochs, was applied. The total number of training epochs was 50, and the mini-batch size was 4. We also stress that we set the architecture of $\hat{\mathbf{X}}_{\boldsymbol{\theta}}(\mathbf{Z})$ identical to that of 17-layers DnCNN in (Zhang et al., 2017) to make a fair comparison. The pseudo algorithm for training a generative model is in Algorithm 1

**Algorithm 1** Training G2G, all experiments in this paper used the defaults values, $n_{epoch} = 50$, $\alpha_{G2G} = 1e^{-3}$

**Require** $\mathcal{D}$, $g_{\theta_1}$, $g_{\theta_2}$, $\phi$, num_iter, m
**for** $j \leftarrow 1$, num_iter **do**
  **for** $ep \leftarrow 1, n_{epoch}$ **do**
    Sample $\{r^{(i)}_{j,1}, r^{(i)}_{j,2}\}^m_{i=1} \sim N(0, I)$, $\{Z^{(i)}\}^m_{i=1} \sim \mathcal{D}$
    $\boldsymbol{\phi}_j \leftarrow \arg\min_{\boldsymbol{\phi}} \mathcal{L}_{\text{G2G}}(\boldsymbol{\phi}, \hat{\mathcal{D}}_j)$,
  **end for**
**end for**
**return** $\boldsymbol{\phi}_{num_iter}$

### 2.4 Noise2Void (Krull et al., 2019)

We used the publicly available source code of Noise2Void (N2V) (Krull et al., 2019) to obtain the denoising results of N2V. Most of the hyperparameters were set to the default ones, but we changed three things to make a fair comparison with our method.

Firstly, while the CNN architecture for the original N2V was a UNet3, we used the DnCNN (Zhang et al., 2018) with 17 layers such that it has the same structure as our G2G model. Secondly, as also is

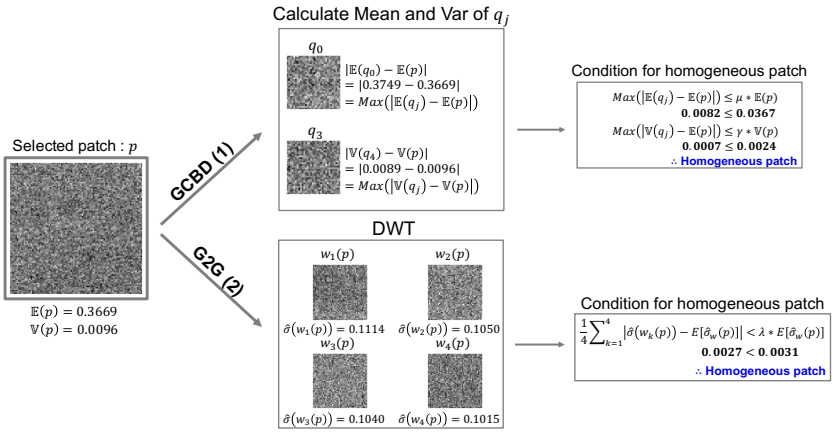
done in (Krull et al., 2019), we had to use a validation set to do a proper model selection for N2V (i.e., the best epoch), while our G2G does not require any validation set (since we always use a model at the last epoch). The reason why N2V needs a validation is that its learning curve is very unstable and a proper model selection greatly affects the final denoising performance. To that end, since we used 20,500 patches with $120 \times 120$ size for training our G2G and other baselines, we divided the 20,500 patches into 18,000 training patches and 2,500 validation patches for training and selecting the best N2V model. Thirdly, we set 'mini_batch_size' to 4 (as our G2G) and 'train_steps_per_epoch' to 'num_of_training_data / mini_batch_size', hence, 4,500. Other hyperparameters are given in Table 6.

Table 6: Hyperparameters for N2V

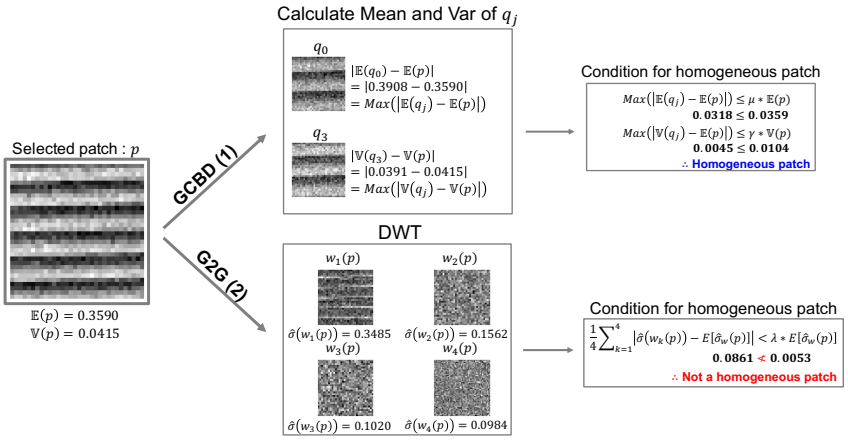
| Hyperparameter | Value |
|---|---|
| train_steps_per_epoch | 4,500 |
| train_loss | 'mse' |
| train_scheme | 'Noise2Void' |
| train_batch_size | 4 |
| n2v_num_pix | 64 |
| n2v_patch_shape | (64,64) |
| n2v_manupulator | 'uniform_withCP' |
| n2v_neighborhood_radious | '5' |

## 3 Comparison of the patch extraction rules

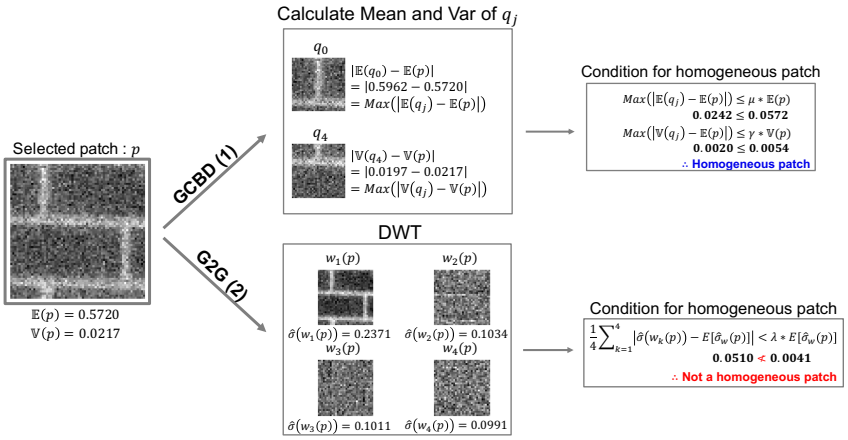
Here, we make a further, thorough comparison between the GCBD smooth patch extraction rule [Eq.(3), manuscript] and ours [Eq.(4), manuscript]. We selected three noisy patches from [Figure 2(b), manuscript] and show the decision criterion of each rule for each image Figure 3. From the figure, we can observe that while our G2G rule correctly excludes the patches in Figure 3(b) and 3(c) as non-homogeneous patches, the GCBD rule wrongly determines them also as homogeneous patches. That is, we note that since the DWT transform used in our rule can successfully disaggregate the high and low frequency components in the patches, the patches with *self-similar repeating patterns* would have significantly varying sub-band coefficient variances as shown in the figures. Hence, our rule can exclude those patches. However, in the GCBD rule, there may exist a sub-patch $\boldsymbol{q}_j$ that has similar empirical mean and variance as the original patch $\boldsymbol{p}$, thus, it may determine the patches with the self-similar repeating patterns as homogeneous as well. We believe these examples clearly show the stark difference between our rule and the GBCD rule for smooth patch extraction.




(a) Example patch determined as homogeneous by the both rules.




(b) Example patch 1 that is wrongly determined as homogeneous by the GCBD rule




(c) Example patch 2 that is wrongly determined as homogeneous by the GCBD rule

Figure 3: Noise patch extractions of GCBD (3) and G2G (4) rules.

### 3.1 RESULT TABLE FOR REAL MICROSCOPY NOISE

We report the detailed experimental results on the real microscopy images in Table 7. We observe that Iterative G2G increases PSNR/SSIM in WF.

Table 7: Experimental results on the real microscopy dataset

| Data Type | Noise Type | DnCNN-S | DnCNN$_B$ | BM3D | N2V (DnCNN) | $g_{\theta_2}$ | G2G$_1$ | G2G$_2$ | G2G$_3$ | G2G$_4$ | G2G$_5$ | G2G$_6$ | G2G$_7$ | N2C (GCBD) | N2C ((Eq.(4)) |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| WF | Raw | 35.39 /0.8738 | 25.43 /0.3702 | 26.32 /0.4012 | 25.31 /0.3411 | 28.40 /0.5261 | 30.63 /0.6407 | 32.03 /0.6889 | 32.21 /0.7114 | 32.30 /0.7253 | 32.56 /0.7672 | 32.62 /0.7910 | 32.74 /0.8158 | 31.16 /0.7493 | 32.26 /0.8205 |
| | Avg = 2 | 36.11 /0.8969 | 28.36 /0.5292 | 29.21 /0.5642 | 28.23 /0.4500 | 29.73 /0.5844 | 31.84 /0.6717 | 32.41 /0.6920 | 32.80 /0.7161 | 32.85 /0.7500 | 32.90 /0.7697 | 32.92 /0.7783 | 32.85 /0.7808 | 31.88 /0.7575 | 33.23 /0.8218 |
| | Avg = 4 | 37.46 /0.9182 | 31.32 /0.6910 | 32.19 /0.7202 | 31.28 /0.6676 | 31.41 /0.6580 | 33.32 /0.7728 | 33.52 /0.7974 | 33.71 /0.8079 | 33.68 /0.8091 | 33.85 /0.8140 | 33.69 /0.8088 | 33.79 /0.8135 | 34.42 /0.8665 | 34.79 /0.8559 |
| | Avg = 8 | 39.81 /0.9374 | 34.63 /0.8218 | 35.76 /0.8444 | 34.85 /0.8097 | 34.81 /0.8084 | 35.16 /0.8315 | 35.16 /0.8315 | 35.27 /0.8325 | 35.21 /0.8321 | 35.27 /0.8333 | 35.25 /0.8330 | 35.22 /0.8316 | 36.92 /0.9126 | 36.86 /0.8800 |
| | Avg = 16 | 42.10 /0.9569 | 37.82 /0.9136 | 39.67 /0.9293 | 38.75 /0.9094 | 36.97 /0.9086 | 38.97 /0.9153 | 38.98 /0.9174 | 38.84 /0.9172 | 38.84 /0.9170 | 38.87 /0.9175 | 38.82 /0.9181 | 38.82 /0.9178 | 38.72 /0.9110 | 38.92 /0.9181 |
| Average | | 38.17 /0.9166 | 31.52 /0.6652 | **32.63 /0.6919** | **31.68 /0.6365** | **32.26 /0.6971** | **33.98 /0.7664** | **34.41 /0.7855** | **34.57 /0.7970** | **34.58 /0.8067** | **34.69 /0.8203** | **34.66 /0.8258** | **34.68 /0.8324** | 34.62 /0.8394 | 35.21 /0.8592 |

### 3.2 DESCRIPTION AND THE RESULT TABLE ON RECONSTRUCTED CT DATASET

The reconstructed CT dataset consists of chest and head parts of 27 pediatric extended cardiac-torso phantoms (Segars et al., 2015), which provide a highly realistic model of the human anatomy. We extracted 60 image slices from each phantom, leading to 1620 image slices in total. The dataset was generated in the following procedure. First, noiseless projection data were acquired in a parallel-beam geometry with Siddon's ray-driven algorithm (Sidky & Pan, 2008). To reduce view aliasing artifacts, the detector quarter-offset was used during a CT scan. Second, Poisson noise was generated and added to the noiseless projection data. Note that the mean number of detected photons was set to 2,500, 5,000, 7,500, and 10,000 to simulate 25%, 50%, 75%, and 100% of a normal dose, respectively. Finally, the images were reconstructed by filtered backprojection (Hsieh, 2003). To preserve fine anatomical structures in the images, the Ram-Lak filter was used as a reconstruction filter. Detailed simulation parameters are summarized in Table 8.

Table 8: Simulation parameters

| Parameters | Values |
|---|---|
| Source to iso-center distance | 595 mm |
| Source to detector distance | mm |
| Detector cell size | 0.7 mm |
| Detector array size | 736 x 1 |
| Data acquisition angle | 360 dares |
| Number of projection views | 736 |
| Reconstructed pixel width | 0.67 mm |
| Reconstructed matrix size | 512x512 |

We divided 27 phantoms into training and test data and the phantom number for each dataset is in Fig 9. Also, We visualized the first image of Female 1 in Fig 4. We can clearly see that each dose has a different noise level, and the noise is source independent and correlated. Finally, Table 10 shows the details of experimental results on Reconstructed CT dataset.

Table 9: Training and test data information of Reconstructed CT dataset

| | Training data | Test data |
|---|---|---|
| Female | 1,3,4,5,6,7,8,9,10,11 | 13,14,15 |
| Male | 1,2,3,4,5,6,7,8,9,10,11 | 12,13 |
| # of images | 21x60 = 1260 | 60x5 = 300 |

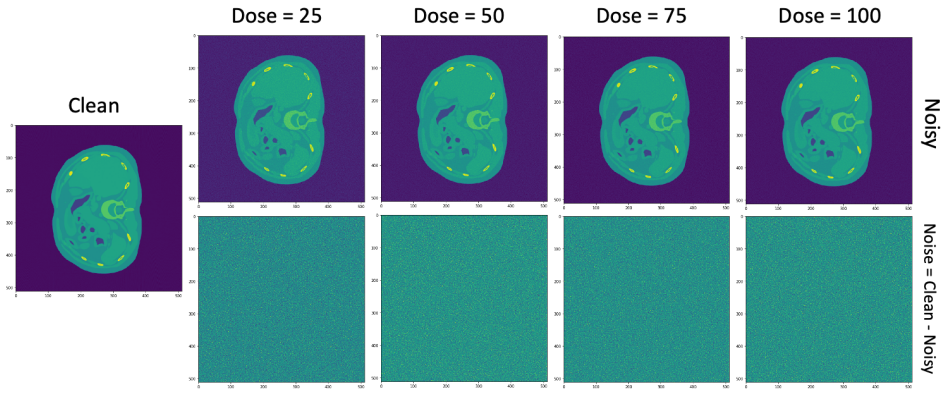



Figure 4: Clean, noisy and noise images from Reconstructed CT

Table 10: Experimental results on the reconstructed CT

| Data Type | Noise Type | N2C (UNnet) | DnCNN$_B$ | BM3D | N2V (UNet) | $g_{\theta_2}$ | G2G$_1$ | G2G$_2$ | G2G$_3$ |
|---|---|---|---|---|---|---|---|---|---|
| Reconstructed CT | Dose 25 | 48.43 /0.9609 | 35.50 /0.6055 | 42.40 /0.7575 | 31.57 /0.5416 | 34.66 /0.5759 | 40.49 /0.8301 | 46.04 /0.9579 | 47.47 /0.9707 |
| | Dose 50 | 49.07 /0.9600 | 38.48 /0.7440 | 45.14 /0.8510 | 34.17 /0.6931 | 37.70 /0.7202 | 43.38 /0.9063 | 47.78 /0.9702 | 48.06 /0.9705 |
| | Dose 75 | 49.45 /0.9591 | 40.09 /0.8111 | 46.55 /0.8929 | 35.90 /0.7756 | 39.49 /0.7919 | 44.96 /0.9320 | 48.80 /0.9744 | 49.20 /0.9733 |
| | Dose 100 | 49.63 /0.9565 | 41.19 /0.8513 | 47.48 /0.9169 | 37.26 /0.8118 | 40.75 /0.9350 | 46.11 /0.9492 | 49.19 /0.9760 | 48.83 /0.9718 |
| Average | | 49.15 /0.9591 | 38.82 /0.5730 | **45.39 /0.8546** | **34.73 /0.7055** | **38.15 /0.7558** | **43.74 /0.9044** | **47.95 /0.9696** | **48.39 /0.9715** |

## 4 Analysis on Real Microscopy Image

In this section, we analyze why our G2G also works well on the real microscopy image dataset (WF) Zhang et al. (2019). although the source-dependent noise does not satisfy our assumption on the noise. The real microscopy image dataset consists of three different types of dataset, which are Wide-Focal(WF), Two-Photon(TP) and Con-Focal(CF), and It is generally known that the real noise follows the Poisson-Gaussian model Zhang et al. (2019),

$$Z_i = x_i + N_i, \quad i = 1, 2, \ldots, \tag{11}$$

in which $N_i \sim \mathcal{N}(0, \sigma_i^2)$ and

$$\sigma_i^2 = \alpha x_i + \sigma^2 \tag{12}$$

with a scaling factor $\alpha > 0$. Thus, the noise variance depends on the underlying clean source pixel value, and $\alpha$ determines the level of the dependence.

In Table 7, we observe that our G2G performs well for WF compared to other baselines, hence, we visualize clean, noisy, noise images from each set and examine if there are any notable difference in the noise distributions. Figure 5 shows two image samples (Avg$=1$ cases) from the Wide-Focal (WF) set. The noise images are obtained by subtracting the clean images from its noisy versions. Note even though the intensities in the source images change significantly among pixels (particularly for the top image), the noise images do not show any source-dependent patterns. Hence, we can deduce that $\alpha$ may be small for the WF images. Also, we could see that there is a correlated pattern in the noise. We belive that these back the good performance of G2G for the WF set.

Figure 6, on the other hand, visualizes an image from TP set for Avg$=\{1, 16\}$ cases. Comparing with Figure 5, we can clearly see the source-dependent patterns in the noise images, particularly severely for the Avg $= 1$ case. Also, CP set showed the similar source-dependent noise patterns. This source-dependent noise is not in our assumption so we did not apply GAN2GAN to TP and CP.

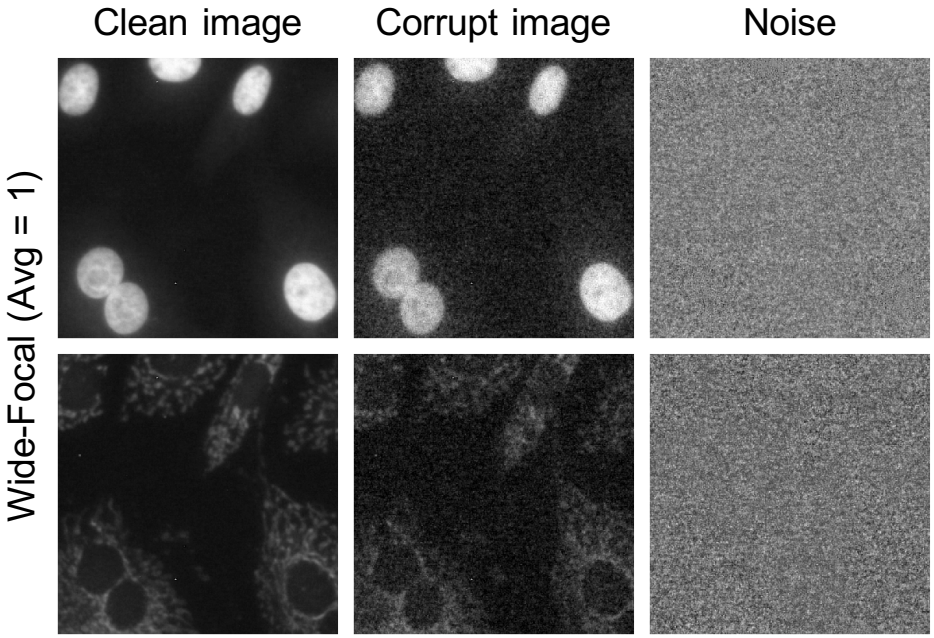



Figure 5: Clean, noisy and noise images from the WF set. (Best viewed in PDF.)

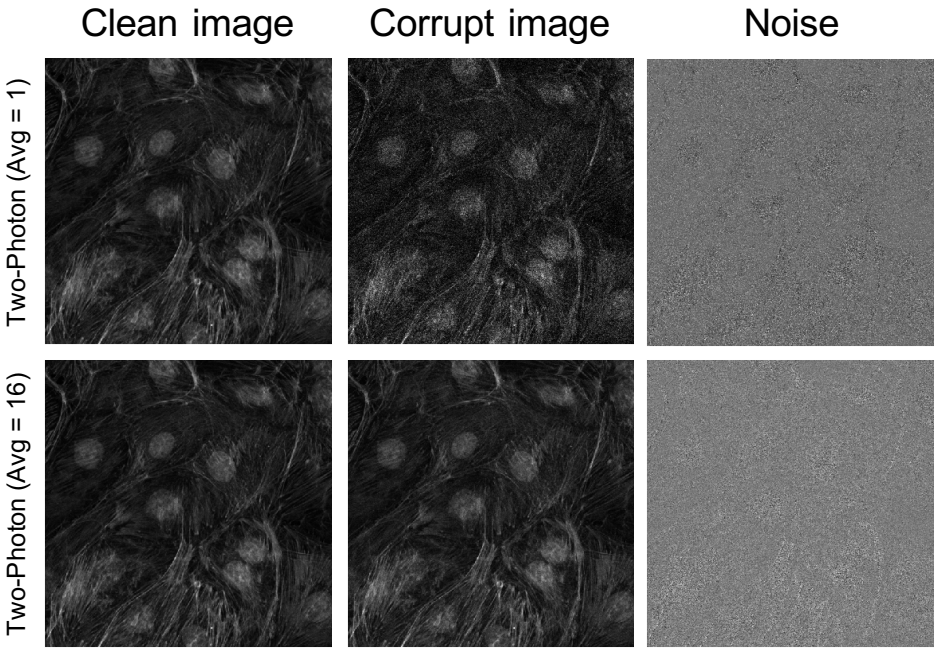



Figure 6: Clean, noisy and noise images from the TP set. (Best viewed in PDF.)

However, we want to stress out that the source independent real noise also exists and GAN2GAN shows the best result compared to any other baselines.

## 5 VISUALIZATIONS

### 5.1 VISUALIZATION OF $\tilde{\mathbf{Z}}$

Figure 7 and 8 visualize the simulated noisy image pairs $(\hat{\mathbf{Z}}_1, \hat{\mathbf{Z}}_2)$, generated from our generative model, for synthetic and real noise cases, respectively. A close examination shows that the images are not simple copies of the original noisy image $\mathbf{Z}$ but are successfully synthesized with the independent noise processes.

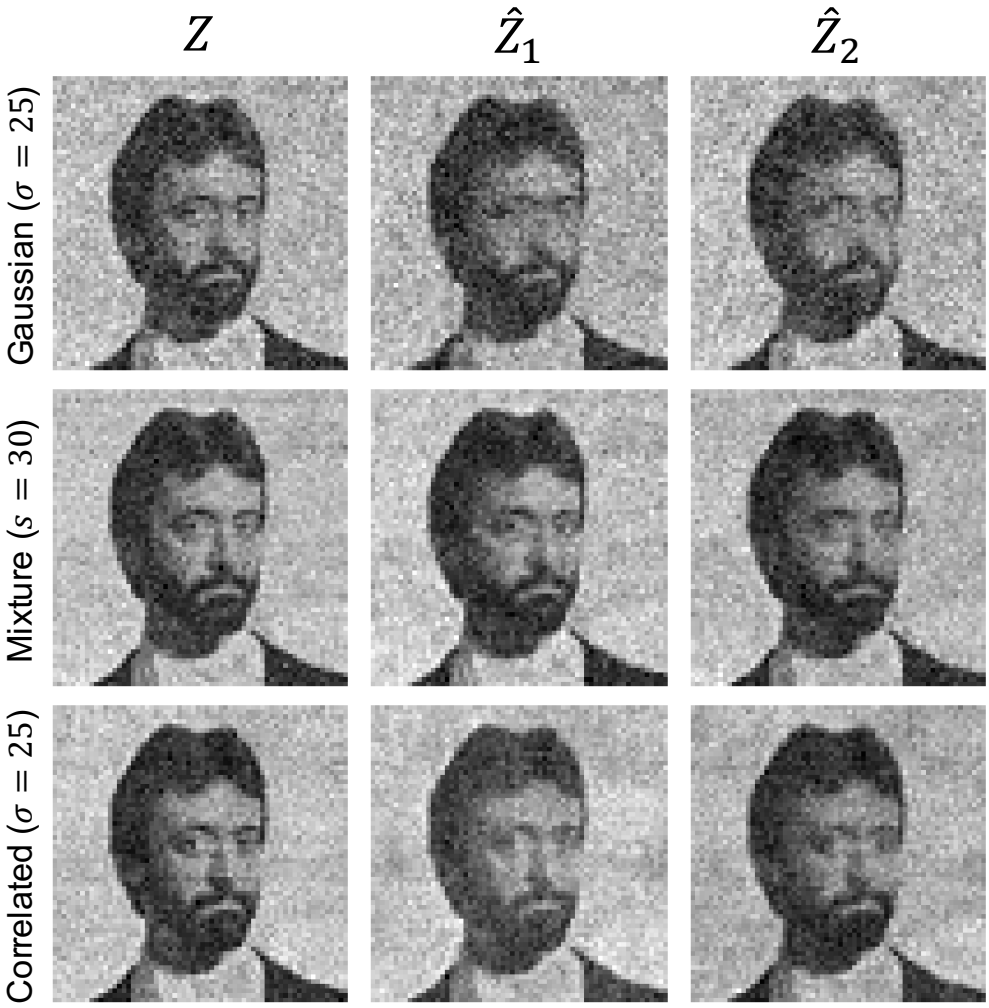



Figure 7: Visualizations of synthesized synthetic noisy image pairs.

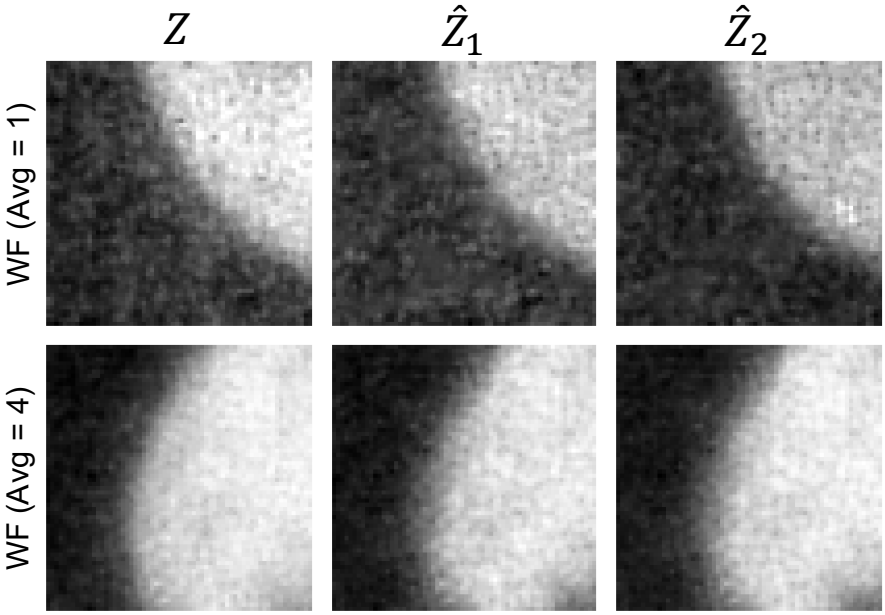



Figure 8: Visualizations of synthesized real noise image pairs.

### 5.2 Visualization of denoised images on BSD68

Figure 9 visualizes the denoising results of a BSD68 image for different types of noise. Note the clear difference in the noise characteristics for Gaussian, mixture, and correlated noises. The visualization of G2G$_3$ certiainly seems better than N2V and BM3D, in line with the PSNR results. DnCNN-B and Noise2Noise use more information than G2G$_3$, but the visualzation as well as the PSNR of G2G$_3$ are comparable to those of the two methods.

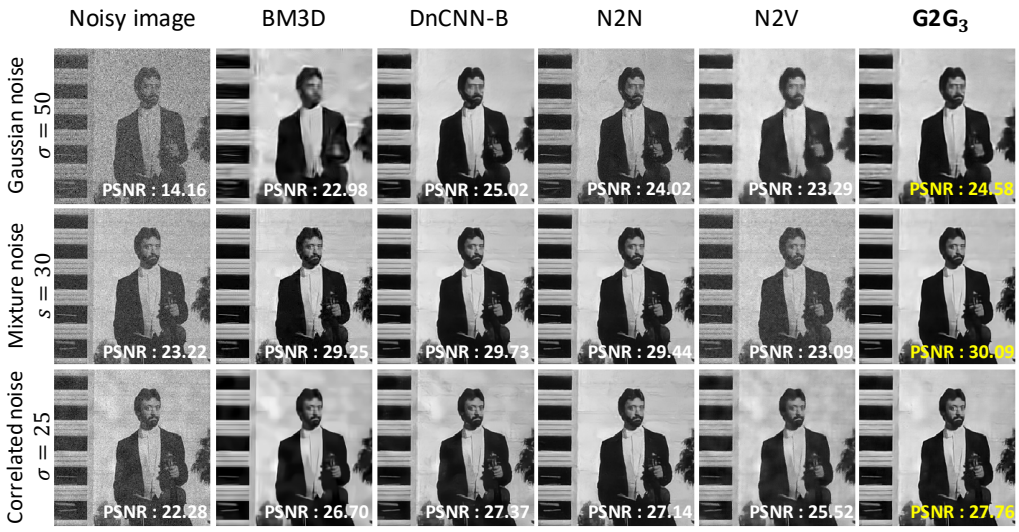



Figure 9: Denoising results on the synthetic noise images.

### 5.3 ADDITIONAL VISUALIZATIONS ON THE REAL MICROSCOPY IMAGES

We also visualize additional denoised images of WF images in Figure 10. We can see that the denoising results of the baselines for WF (Avg = 1) are very noisy, but $G2G_3$ shows relatively clean denoising results than others.

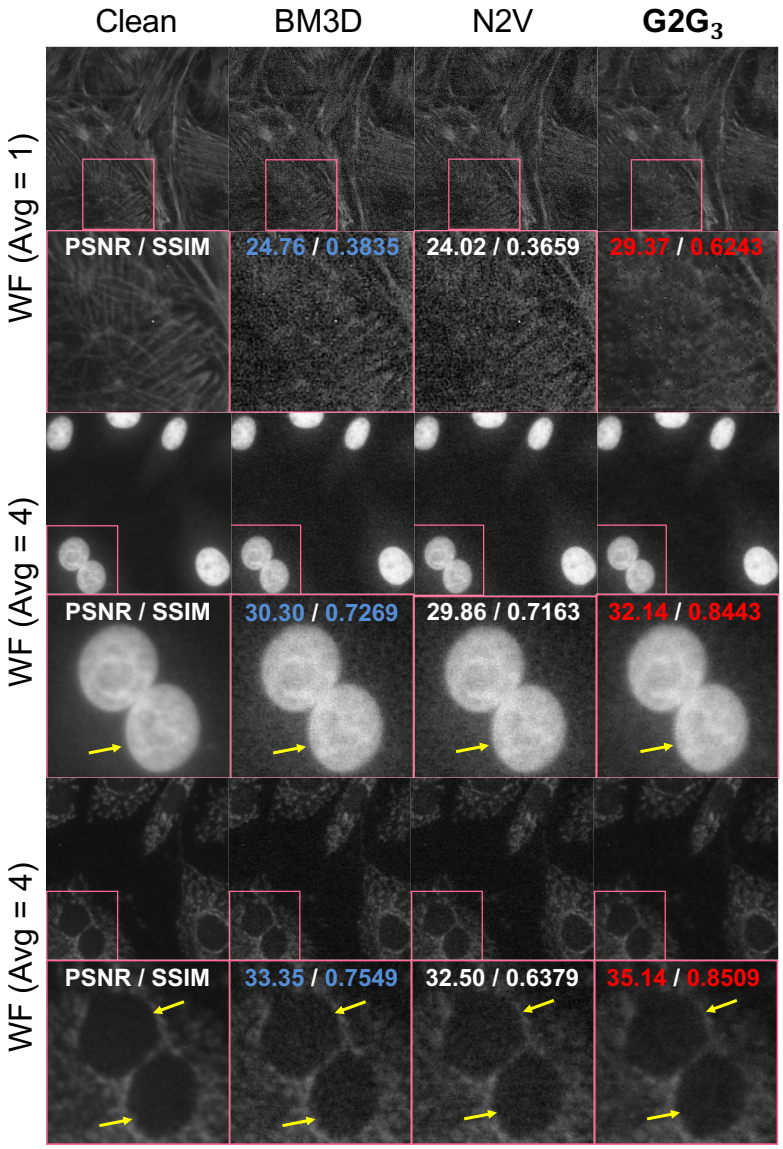



Figure 10: Denoising results on the real noisy microscopy images.

### 5.4 Additional visualizations on the Reconstructed CT

We visualize the denoising result of a Reconstructed CT image in Figure 11. We observed that BM3D and N2V shows a stil noisy result on this image but $G2G_3$ shows a clearly denoised result.

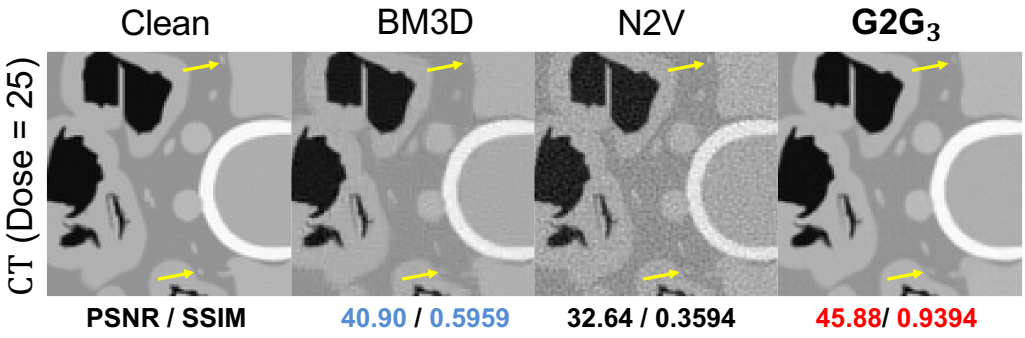



Figure 11: Denoising results on Reconstructed CT images.

## 6 Experimental Results of $G2G_3$ with $(\hat{\mathbf{Z}}_{j1}^{(i)}, \hat{X}_{\phi_{j-1}}(\mathbf{Z}^{(i)}))$

Table 11: Experimental results for Reviewer 3's Q.(2).

| PSNR / SSIM | $G2G_3$ | |
|---|---|---|
| | $(\hat{\mathbf{Z}}_{j1}^{(i)}, \hat{X}_{\phi_{j-1}}(\mathbf{Z}^{(i)}))$ | $(\hat{\mathbf{Z}}_{j1}^{(i)}, \hat{\mathbf{Z}}_{j2}^{(i)})$ |
| Gaussian ($\sigma = 25$) | 29.02 / 0.8153 | 28.96 / 0.8080 |
| Mixture ($s = 30$) | 30.70 / 0.8621 | 30.49 / 0.8538 |
| Correlated ($\sigma = 25$) | 24.04 / 0.8673 | 28.00 / 0.8447 |
| WF (Avg = 1) | 16.24 / 0.4490 | 34.57 / 0.7970 |
| Medical (Dose = 25) | 18.52 / 0.7397 | 47.47 / 0.9707 |

We did the experiments on $G2G_3$ with $(\hat{\mathbf{Z}}_{j1}^{(i)}, \hat{X}_{\phi_{j-1}}(\mathbf{Z}^{(i)}))$ and Table 11 shows the experimental results. From the table, we observe the suggested approach (left column) can in fact achieve slightly improved results for synthetic Gaussian and Mixture noise. However, we observe that the performances of the approach for the Correlated and WF/Medical datasets deteriorate significantly compared to ours.

## REFERENCES


Jingwen Chen, Jiawei Chen, Hongyang Chao, and Ming Yang. Image blind denoising with generative adversarial network based noise modeling. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 1

Jiang Hsieh. *Computed tomography: principles, design, artifacts, and recent advances*, volume 114. SPIE press, 2003. 7

D. Kingma and J. Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations (ICLR)*, 2015. 3

Alexander Krull, Tim-Oliver Buchholz, and Florian Jug. Noise2void-learning denoising from single noisy images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 4, 5

WP Segars, Hannah Norris, Gregory M Sturgeon, Yakun Zhang, Jason Bond, Anum Minhas, Daniel J Tward, JT Ratnanather, MI Miller, D Frush, et al. The development of a population of 4d pediatric xcat phantoms for imaging research and optimization. *Medical physics*, 42(8):4719–4726, 2015. 7

Emil Y Sidky and Xiaochuan Pan. Image reconstruction in circular cone-beam computed tomography by constrained, total-variation minimization. *Physics in Medicine & Biology*, 53(17):4777, 2008. 7

Tieleman and G. Hinton. RMSProp: Divide the gradient by a running average of its recent magnitude. In *Lecture Note 6-5, University of Toronto*, 2012. 3

K. Zhang, W. Zuo, Y. Chen, D. Meng, and L. Zhang. Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising. *IEEE Trans. Image Processing*, 26(7):3142 – 3155, 2017. 4

Yide Zhang, Yinhao Zhu, Evan Nichols, Qingfei Wang, Siyuan Zhang, Cody Smith, and Scott Howard. A poisson-gaussian denoising dataset with real fluorescence microscopy images. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 8

Yulun Zhang, Yapeng Tian, Yu Kong, Bineng Zhong, and Yun Fu. Residual dense network for image restoration. *arXiv preprint arXiv:1812.10477*, 2018. 3, 4