10.2352/EI.2022.34.15.COLOR-158
© 2022, Society for Imaging Science and Technology

Mimicking DBS halftoning via a deep learning approach

***Baekdu Choi*[1] *and Jan P. Allebach*[1]**
[1] ***Electronic Imaging Systems Laboratory, School of Electronic and Computer Engineering, Purdue University, West Lafayette, IN 47907, US; choi504, allebach@purdue.edu***

Abstract

In this paper we investigate applying two deep generative models to digital halftoning with the aim of generating halftones with comparable quality to those generated with the direct binary search (DBS) algorithm. For the first framework, we apply conditional generative adversarial networks (cGANs) using two discriminators with different receptive field size and a generator consisting of densely connected blocks. For the second framework, deep autoregressive (AR) models, we propose mapping input images into a feature space using a single forward pass of a deep neural network and then applying a shallow autoregressive model at the end output. Our methods show promising results; halftones generated with our algorithms are less noisy than those generated with DBS screen and do not contain artifacts commonly associated with error diffusion type algorithms.

Introduction

Recent advances in deep learning for image processing have brought incredible achievements in diverse areas, but not much work has been done in applying deep learning to the area of digital halftoning. Furthermore, most of the work in halftoning using deep learning has been focused on inverse halftoning, such as in [1, 2, 3, 4, 5, 6], and to our knowledge, only a limited amount of work has been on halftoning, such as [5, 7]. Therefore, in this paper, we aim to investigate applying deep learning, specifically deep generative models, to digital halftoning.

Deep generative models are an area of research using deep learning where the goal is to find the mapping from some prior distribuion to the distribution of the space which the training set represents. Commonly used generative models include variational autoencoders (VAEs) [8, 9, 10], generative adversarial networks (GANs) [11, 12, 13, 14, 15, 16, 17, 18], normalizing flow models [19, 20], deep autoregressive (AR) models [21, 22, 23], and most recently, stochastic diffusion models [24, 25]. We choose GANs and deep AR models as the models to work with since they are more straightforward to apply to halftoning.

This paper is organized as follows. First, we briefly introduce the related works for our paper, namely digital halftoning, GANs, and deep AR models. Next, we discuss our algorithms applying GANs and deep AR models to halftoning, and describe our contributions. Lastly, we show the halftone results illustrating that our algorithms are capable of generating halftones with comparable quality to those generated with the DBS algorithm [26], which is known to generate the best quality aperiodic, dispersed-dot halftones.

Related work

Digital halftoning

Digital halftoning algorithms aim to convert a continuous-tone image into an image with a limited number of tone values while preserving the image quality as much as possible. Halftoning algorithms can be classified based on whether the dot placement is periodic or aperiodic, and whether they are clustered-dot or dispersed-dot halftoning algorithms. In this paper, we focus on implementing aperiodic dispersed-dot halftoning algorithms.

Commonly used halftoning algorithms can also be classified into three categories: screening, error diffusion, and search-based algorithms. A screening algorithm such as [27, 28] compares each pixel to a corresponding index of a halftone screen to determine the halftone pixel value at the pixel location. While these types of algorithms are fast, they tend to generate noisy halftones. Error diffusion type algorithms such as [29] quantize the pixel first, and then the quantization error is diffused to nearby pixels, so that the error can be taken into account for quantizing the pixels afterwards. Lastly, search-based algorithms iterate over the image to optimize some cost function, such as the direct binary search (DBS) algorithm in [26]. While search-based algorithms generate the highest quality halftones, they also tend to be computationally expensive due to their iterative nature.

Generative adversarial networks

Generative adversarial networks (GANs), first proposed in [11], learn to generate realistic images by adversarially training two networks, a generator and a discriminator. The goal of discriminator is to learn to distinguish between real samples from the training set and generated samples from the generator. On the other hand, generators take a random vector, typically called a latent vector, as input and generate an image. They are trained with the goal of fooling the discriminator. To train a GAN, a two time-scale update rule [30] is typically used, where the generator and the discriminator are alternately updated while fixing the weights of the other one. Upon convergence, ideally, the generator generates images that are indistinguishable from those in the training set.

Due to the unstable nature of its training, a large amount of research on GANs was done on the objective (loss function) for its training. Wasserstein GAN [12] proposes using Earth-Mover (EM) distance as the metric for computing disparity between the training set distribution and the generator output distribution, which can be used to update the network weights via its dual form when searching within the set of 1-Lipschitz functions for the discriminator. WGAN-GP [31] proposes using a gradient penalty loss term instead of clipping the weights, as in [12], for imposing the Lipschitz constraint. LSGAN [13] proposes using a least-squares adversarial loss, which is widely adopted due to its relatively stable nature and simple form. Another widely used adversarial loss, hinge GAN loss, is proposed in [14] where the authors incorporate the concept of maximizing margins of the separating hyperplane from support vector machines (SVMs) [32] into GAN training. Relativistic GAN [15] proposes using relativistic versions of the widely-used adversarial losses to stabilize the training further.

There also has been a large amount of research on improving quality of the images generated by GANs. [33] proposes imposing the Lipschitz constraint on the discriminator by applying spectral normalization to the convolution layers, which since has become a common practice for GANs. Self-attention GAN [16] proposes using self-attention [34] in the generator to promote global structural consistency in the generated images. [17] generates images in high resolution by gradually increasing the resolution, via incrementally adding layers in the generator and the discriminator. State-of-the-art GANs such as StyleGANv2 [18] are able to produce high-quality images in high resolution such as 1024×1024 without visible artifacts. Since the works mentioned above are conducted with continuous-tone images, these improvements are not necessarily applicable to halftones. In fact, we did not observe any notable improvements in halftone quality when we added spectral normalization or self-attention to our GAN-based halftoning module.

While in its original form, GANs learn to map a random vector into an image, GANs can also be used for image-to-image translation tasks such as style transfer [35, 36], semantic image synthesis [37, 38], image super-resolution [39, 40], image inpainting [41], and so on. When used for image-to-image translation, these GANs are often referred to as conditional GANs. The terminology comes from treating the input images as conditional inputs in addition to the latent vectors, but often the latent vector is not used since the GAN learns to ignore it [35]. We also adopt the conditional GAN for developing a halftoning algorithm since halftoning can be thought of as an image-to-image translation task.

Deep autoregressive models

Deep autoregressive (AR) models assume a dependency of each pixel on the previous pixels under some scan order, where the dependency is modeled via a deep neural network. Compared to other generative models such as GANs, they are known to have the following benefits [42]. First, AR models can capture dependencies between the neighboring pixels and promote local consistency. Second, training AR models is stable compared to other generative models.

The first deep AR models proposed are PixelCNNs and PixelRNNs in [21], where the causal relation between pixels is modeled by a convolutional neural network (CNN) or a recurrent neural network (RNN). Since causality of the model is crucial for AR models, the authors of [21] propose using masked convolution, where the uncausal pixels are masked out prior to convolution. Originally PixelRNNs outperformed PixelCNNs, but [22] proposed using gated activation and adding horizontal convolution stacks to get rid of convolution blind spots, resulting in the gated PixelCNNs outperforming PixelRNNs. More recently, AR models were incorporated with transformers [34] in works such as [23].

While AR models are useful in modeling image priors, they are inherently computationally expensive. For training AR models, the ground truth images from the training dataset are fed as input, and since the networks are built to be causal, a single forward and backward pass is sufficient to compute the gradients and update the network weights. On the other hand, for inference, since each pixel has to be inferred following the scan order, parallelization is not possible. Thus, the inference requires a large amount of time. Because of this, papers on AR models mentioned earlier mostly perform experiments in low-resolution datasets such as CIFAR-10 [43] and a downscaled version of ImageNet [44].

Conditional GANs for halftoning

Our implementation of conditional GANs is loosely based on the cGAN framework in [35]. That is, we learn the forward mapping from continuous-tone images to corresponding halftones via supervision. Instead of using only continuous-tone images as inputs, we add the DBS-screened [28] halftones of the continuous-tone images as a secondary set of input images. We found that adding screened images as inputs improves halftone quality slightly.

For the generator, we stack densely connected blocks [45]. Originally proposed as part of a classification network in [45], they are known to be useful in image processing tasks, such as in [40] where densely connected blocks are used in a residual manner. We use four convolution layers in each densely connected blocks. And in each convolution layer, convolution is followed by ReLU activation [46] and instance normalization [47]. A densely connected block is illustrated in Figure 1. Our generator overall includes two convolution layers, followed by six densely connected blocks and two convolution layers at the end, as illustrated in Figure 2. We do not perform any downscaling/upscaling since we found that it degrades the output halftone quality.

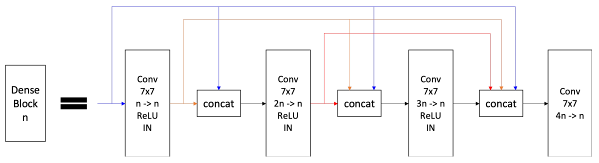

Figure 1. *A densely connected block. IN stands for instance normalization [47].*

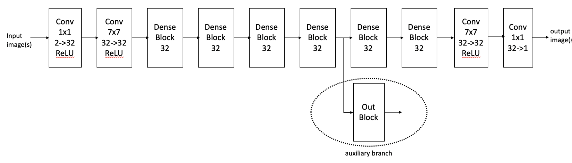

Figure 2. *The generator network structure for our cGAN based halftoning algorithm.*

During training, we also add an auxiliary output from the generator after the fourth densely connected block, followed by two convolution layers. Training using an auxiliary branch is adapted from [48], which is known to enhance gradient flow to the lower layers and increase the correlation between the intermediate feature outputs and final outputs of the network. The auxiliary branch is strictly for training and is discarded during inference.

For the discriminator, we are using Markovian discriminators (PatchGAN), as in [35]. Instead of using a single discriminator, we use two independent discriminators as shown in Figure 3. Similar ideas of using multiple discriminators can be seen in recent papers such as [37]. The first discriminator has a receptive field size of 4×4, whereas the second discriminator has a receptive field size of 94×94. Intuitively, we can think of this as the smaller discriminator learning to see local pixel values to be close to the binarized result, whereas the larger discriminator learns to see the overall dot distribution. We found that using two discriminators stabilizes training of the overall framework.

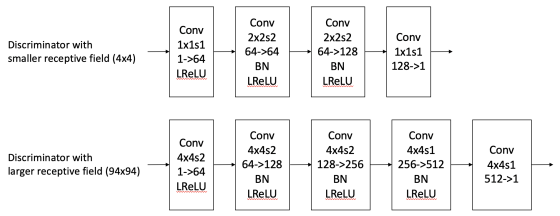
Discriminator with smaller receptive field (4x4)
Conv 1x1s1 1->64 LReLU
Conv 2x2s2 64->64 BN LReLU
Conv 2x2s2 64->128 BN LReLU
Conv 1x1s1 128->1
Discriminator with larger receptive field (94x94)
Conv 4x4s2 1->64 LReLU
Conv 4x4s2 64->128 BN LReLU
Conv 4x4s2 128->256 BN LReLU
Conv 4x4s1 256->512 BN LReLU
Conv 4x4s1 512->1

Figure 3. *The network structures of the two discriminators for our cGAN based halftoning algorithm. After each activation, dropout [49] with $p = 0.5$ is applied to prevent discriminator overfitting. The notation 4x4s2 indicates a convolution kernel of size 4×4 is applied with a stride of 2, 1$->$64 indicates that the input tensor has 1 channel and the output tensor has 64 channels, and BN and LReLU stands for batch normalization [50] and leaky ReLU activation [51], respectively.*

We train the cGAN with supervised training. The training dataset of continuous-tone and halftone image pairs are generated with the DIV2K dataset [52], where we first convert the images to grayscale, then we divide them into nonoverlapping 256 by 256 patches, which then are converted to halftones using the DBS [26] algorithm. This results in around 28,000 image pairs in the training dataset.

For training the cGAN, we use the least-squares GAN [13] loss as the adversarial loss ($\mathscr{L}_{adv}$). For the generator update, we add the L1 loss between the ground truth image and the network output ($\mathscr{L}_{GT}$), which is computed from both the end output and the auxiliary branch output. We also add another loss term, which we call the human visual system loss, defined as

$$\mathscr{L}_{HVS} = \frac{1}{NM} \sum_{n=0}^{N-1} \sum_{m=0}^{M-1} \{(h_{HVS} ** I)[m,n] - (h_{HVS} ** H)[m,n]\}^2 \tag{1}$$

where I denotes the continuous-tone image, H the halftone image, and h_{HVS} the human visual system filter proposed in [53]. The overall loss for the generator update is given as

$$\mathscr{L} = \mathscr{L}_{adv} + 0.5\mathscr{L}_{GT} + 30.0\mathscr{L}_{HVS} \tag{2}$$

For the discriminator update, we only use the adversarial loss. Since the discriminators easily overfit, we scale down the learning rate for updating the discriminators. For the smaller discriminator, we scale the learning rate by 0.2, and for the larger discriminator, we scale the learning rate by 0.04. Also, instead of updating the discriminators using generated images from the current batch, we use a pool of generated images following [54], which is known to prevent the discriminators from un-learning to distinguish image artifacts from generated images that they previously learned. We train the networks for 60 epochs with the Adam optimizer [55] with hyperparameters $(\beta_1, \beta_2) = (0.5, 0.999)$ and batch size of 8. For the learning rate, we start with 5.0×10^{-5}, and after 30 epochs it is scaled by 0.9 every epoch.

Deep AR models for halftoning

As mentioned earlier, naive application of deep AR models can be highly time-consuming at inference since it requires a number of forward passes of the network equal to the number of pixels in the image. While for continuous-tone images this can be bypassed by generating a downscaled version of the image and upscaling, e.g., as in [42], it is not a solution for halftone images.

To reduce runtime at inference, we propose using a deep feedforward network, which we call a feature extractor, to map the continuous-tone images to a feature space prior to AR inference. Jointly training both the feature extraction network and the AR network will result in the features closely related to the output halftones, which will allow us to perform AR inference with the AR network consisting of only a relatively small number of masked convolution layers. The overall framework is illustrated in Figure 4.

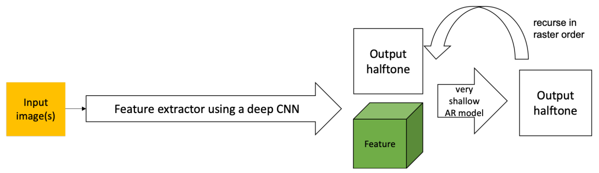

Figure 4. *The proposed framework for deep AR model based halftoning.*

For our halftoning algorithm, we use the generator from the cGAN halftoning algorithm as the feature extractor, except that we remove the output convolution layers and replace them with a single 1×1 convolution layer. For the AR network, we use four masked convolution layers with gated activation [22] followed by a 1×1 convolution layer with sigmoid activation.

For training the deep AR model, we jointly train the feature extractor and the AR network from scratch. Since the output is a binary halftone, we use pixel-wise binary cross entropy loss for training, and the same dataset used for training cGAN is used. We again use the Adam optimizer [55] with hyperparameters $(\beta_1, \beta_2) = (0.5, 0.999)$, with learning rate of 5.0×10^{-5} and batch size 4. The deep AR model is trained for 30 epochs and we do not use any learning rate scheduling.

Results

We will show some halftone results here by using the lake image in Figure 5. This image is of size 2040×1356, and we will zoom into two patches in the image marked with red and blue rectangles in the figure. The red patch is of size 200×133 and the blue patch is of size 300×200.

Figure 6 shows halftone results on a textured region of the image (blue patch in Figure 5). Comparing the halftones generated by our algorithm to the halftone generated by the DBS screen, we can see that our algorithms generate less noisy halftones. On the other hand, our algorithms generated halftones

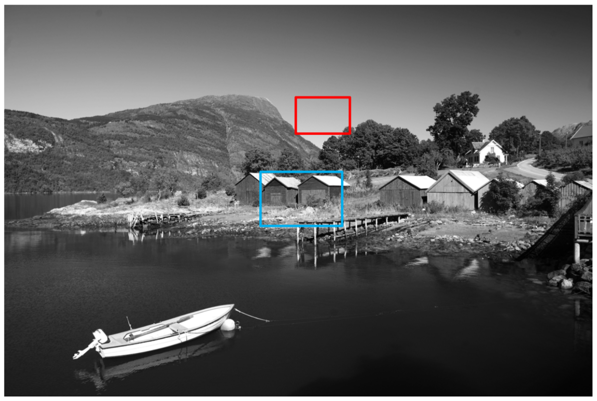

Figure 5. *The original continuous-tone image to be used for showing halftoning results.*

that are comparable to the halftone generated by the DBS algorithm in this area. The error diffusion result shows more detail in the siding of the buildings than all the other halftone results due to the known edge enhancement effect of error diffusion algorithm [56].

Figure 7 shows halftone results on a smooth region of the image (red patch in Figure 5). Here, we can see that error diffusion contains visible vertical and maze-like artifacts, whereas the halftone images generated by the other algorithms do not show such artifacts. Again, the halftone generated by the DBS screen is noisier than the halftones generated by our deep learning based algorithms.

While the deep learning based halftoning algorithms are showing promising results, deep learning is inherently computationally expensive, so it is worth discussing the runtimes it took for running each algorithm on the lake image. The deep learning based algorithms are implemented using PyTorch [57], and we performed experiments with a machine that runs on an Intel i7-9700K CPU with 3.6GHz clock, 16GBs of RAM, and a single core of NVIDIA GeForce RTX2070 super GPU. To perform inference on the entire lake image, cGAN took around 11s on average, whereas the deep AR model took around 2500s. In comparison, the DBS algorithm, which is implemented in C, took around 800s on average.

Conclusions

In this paper, we discussed applying two widely used deep generative models, mainly conditional GANs and deep AR models, for halftoning. For the conditional GAN model, we implemented the generator network using a stack of densely connected blocks, and used two independent discriminators to stabilize the training. For the deep AR model, we proposed using a deep feed-forward convolutional neural network to convert the input image into a feature space, and then performing AR inference using a relatively small AR network. Our halftoning algorithms produced halftones less noisy compared to DBS-screened halftones, especially in textured regions. On the other hand, halftones in smooth image regions showed that our halftoning algorithms do not generate the artifacts commonly associated with error diffusion algorithms. Overall, our halftoning algorithms show great promise in that they are able to generate halftones with comparable quality to those generated with the DBS algorithm.

References

[1] M. Xia and T.-T. Wong, “Deep inverse halftoning via progressively residual learning,” in *Asian Conference on Computer Vision (ACCV)*, 2018.

[2] Y. Xiao, C. Pan, Y. Zheng, X. Zhu, Z. Qin, and J. Yuan, “Gradient-guided DCNN for inverse halftoning and image expanding,” in *Asian Conference on Computer Vision (ACCV)*, 2018.

[3] J. Yuan, C. Pan, Y. Zheng, X. Zhu, Z. Qin, and Y. Xiao, “Gradient-guided residual learning for inverse halftoning and image expanding,” *IEEE Access*, vol. 8, 2019.

[4] H. P. A. Wicaksono, H. Prasetyo, and J.-M. Guo, “Deep learning based inverse halftoning via stationary wavelet domain,” in *27th IEEE International Conference on Telecommunications (ICT)*, 2020.

[5] J.-M. Guo and S. Sankarasrinivasan, “H-GAN: Deep learning model for halftoning and its reconstruction,” in *IEEE International Conference on Consumer Electronics (ICCE)*, 2020.

[6] C.-H. Son, “Inverse halftoning through structure-aware deep convolutional neural networks,” *Signal Processing*, vol. 173, p. 107591, 2020.

[7] M. Xia, W. Hu, X. Liu, and T.-T. Wong, “Deep halftoning with reversible binary pattern,” in *IEEE International Conference on Computer Vision (ICCV)*, 2021.

[8] D. P. Kingma and M. Welling, “Auto-encoding variational Bayes,” *arXiv preprint arXiv:1312.6114*, 2013.

[9] I. Higgins, L. Matthey, A. Pal, C. Burgess, X. Glorot, M. Botvinick, S. Mohamed, and A. Lerchner, “beta-VAE: Learning basic visual concepts with a constrained variational framework,” in *International Conference on Learning Representations (ICLR)*, 2017.

[10] A. van den Oord, O. Vinyals, and K. Kavukcuoglu, “Neural discrete representation learning,” in *Advances in Neural Information Processing Systems*, 2017.

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, “Generative adversarial nets,” in *Advances in Neural Information Processing Systems*, 2014.

[12] M. Arjovsky, S. Chintala, and L. Bottou, “Wasserstein generative adversarial networks,” in *International Conference on Machine Learning (ICML)*, 2017.

[13] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. P. Smolley, “Least squares generative adversarial networks,” in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[14] J. H. Lim and J. C. Ye, “Geometric GAN,” *arXiv preprint arXiv:1705.02894*, 2017.

[15] A. Jolicoeur-Martineau, “The relativistic discriminator: a key element missing from standard GAN,” in *International Conference on Learning Representations (ICLR)*, 2019.

[16] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, “Self-attention generative adversarial networks,” in *International Conference on Machine Learning (ICML)*, 2019.

[17] T. Karras, T. Aila, S. Laine, and J. Lehtinen, “Progressive growing of GANs for improved quality, stability, and variation,” in *International Conference on Learning Representations (ICLR)*, 2018.

[18] T. Karras, S. Laine, M. Aittala, J. Hellsten, J. Lehtinen, and T. Aila, “Analyzing and improving the image quality of StyleGAN,” in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2020.

[19] D. P. Kingma, T. Salimans, R. Jozefowicz, X. Chen, I. Sutskever, and M. Welling, “Improved variational inference with inverse au-

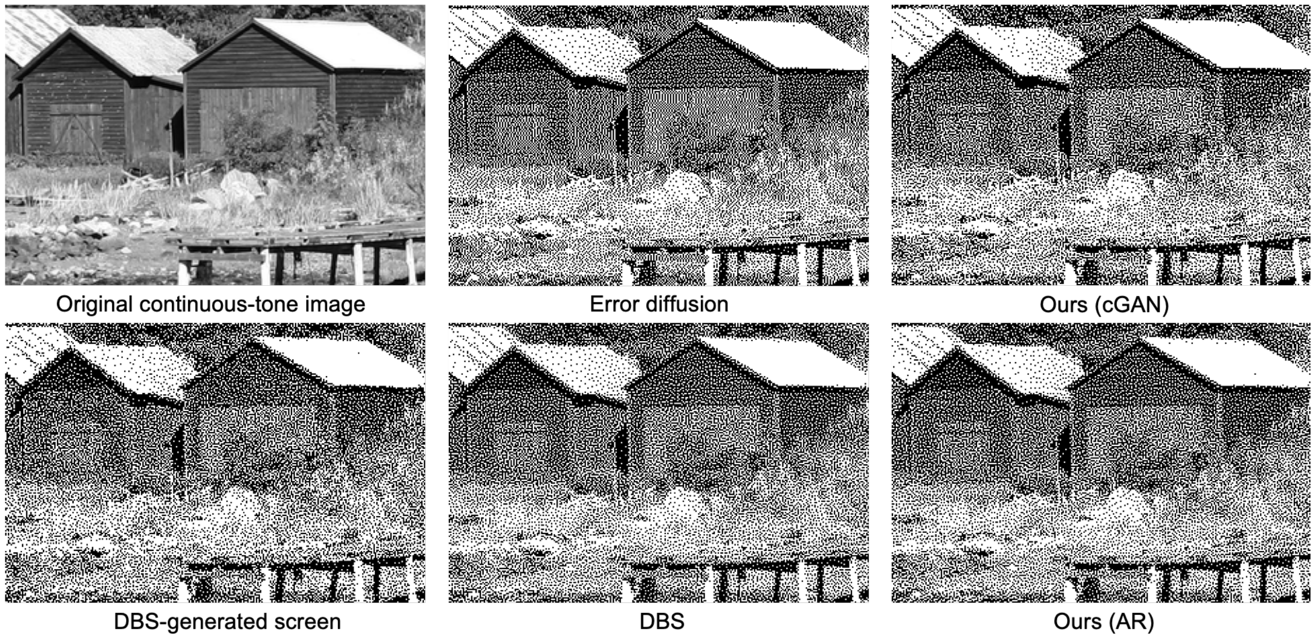

Figure 6. *Halftone results on a textured region. Error diffusion is implemented using the algorithm in [29] with serpentine raster scan, the DBS screen is generated using the algorithm in [28], and the DBS algorithm is from [26]. The continuous-tone and halftone images are zoomed by 200%.*

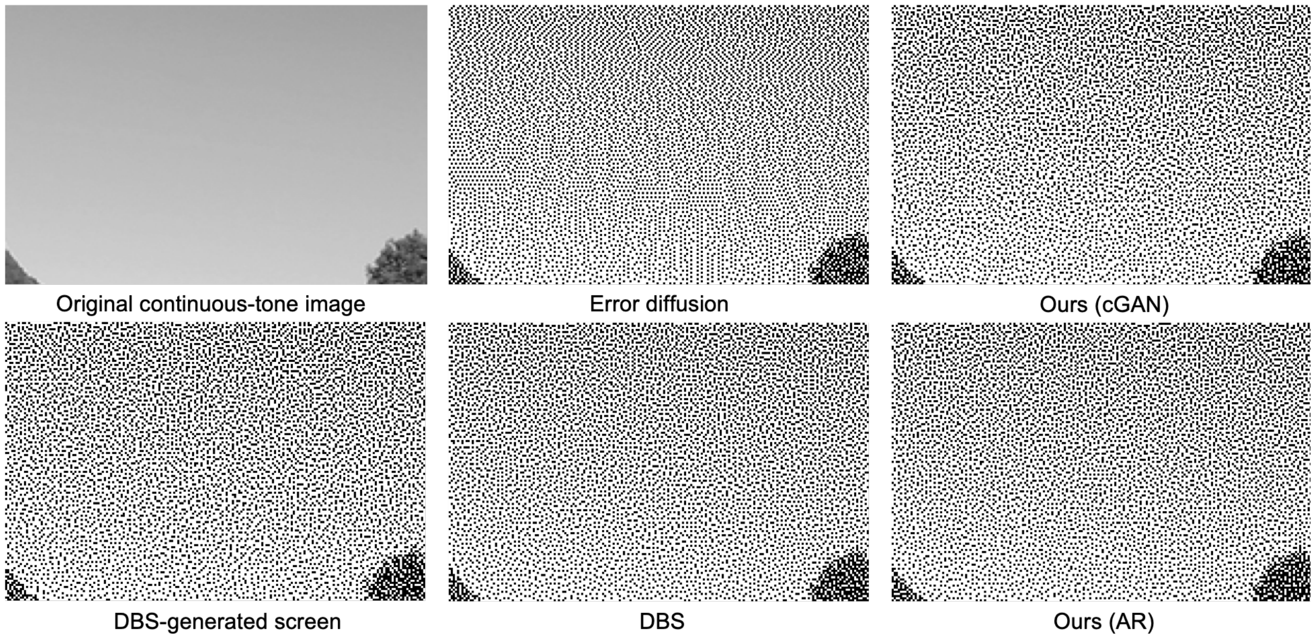

Figure 7. *Halftone results on a smooth region. Error diffusion is implemented using the algorithm in [29] with serpentine raster scan, the DBS screen is generated using the algorithm in [28], and the DBS algorithm is from [26]. The continuous-tone and halftone images are zoomed by 300%.*

toregressive flow," in *Advances in Neural Information Processing Systems*, 2016.

[20] D. P. Kingma and P. Dhariwal, "Glow: Generative flow with invertible 1x1 convolutions," in *Advances in Neural Information Processing Systems*, 2018.

[21] A. V. Oord, N. Kalchbrenner, and K. Kavukcuoglu, "Pixel recurrent neural networks," in *International Conference on Machine Learning (ICML)*, 2016.

[22] A. van den Oord, N. Kalchbrenner, L. Espeholt, K. Kavukcuoglu, O. Vinyals, and A. Graves, "Conditional image generation with PixelCNN decoders," in *Advances in Neural Information Processing Systems*, 2016.

[23] X. Chen, N. Mishra, M. Rohaninejad, and P. Abbeel, "PixelSNAIL: An improved autoregressive generative model," in *International Conference on Machine Learning (ICML)*, 2018.

[24] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems*, 2020.

[25] P. Dhariwal and A. Nichol, "Diffusion models beat GANs on image synthesis," in *Advances in Neural Information Processing Systems*, 2021.

[26] D. J. Lieberman and J. P. Allebach, "A dual interpretation for direct binary search and its implications for tone reproduction and texture quality," *IEEE Trans. Image Process.*, vol. 9, no. 11, pp. 1950–1963, 2000.

[27] R. Ulichney, "The void-and-cluster method for dither array generation," in *Proc. SPIE*, vol. 1913, San Jose, CA, USA, 1993, pp. 332–343.

[28] J. P. Allebach and Q. Lin, "FM screen design using DBS algorithm," in *Proc. IEEE Int. Conf. Image Process.*, Lausanne, Switzerland, 1996, pp. 549–552.

[29] R. W. Floyd and L. Steinberg, "An adaptive algorithm for spatial greyscale," in *Proc. S.I.D.*, vol. 17, no. 2, 1976, pp. 75–77.

[30] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "GANs trained by a two time-scale update rule converge to a local nash equilibrium," in *Advances in Neural Information Processing Systems*, 2017.

[31] I. Gulrajani, F. Ahmed, M. Arjovsky, V. Dumoulin, and A. Courville, "Improved training of Wasserstein GANs," in *Advances in Neural Information Processing Systems*, 2017.

[32] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[33] T. Miyato, T. Kataoka, M. Koyama, and Y. Yoshida, "Spectral normalization for generative adversarial networks," in *International Conference on Learning Representations (ICLR)*, 2018.

[34] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017.

[35] P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros, "Image-to-image translation with conditional adversarial networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[36] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[37] T.-C. Wang, M.-Y. Liu, J.-Y. Zhu, A. Tao, J. Kautz, and B. Catanzaro, "High-resolution image synthesis and semantic manipulation with conditional GANs," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[38] T. Park, M.-Y. Liu, T.-C. Wang, and J.-Y. Zhu, "Semantic image synthesis with spatially-adaptive normalization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[39] C. Ledig, L. Theis, F. Huszár, J. Caballero, A. Cunningham, A. Acosta, A. Aitken, A. Tejani, J. Totz, Z. Wang, and W. Shi, "Photo-realistic single image super-resolution using a generative adversarial network," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[40] X. Wang, K. Yu, S. Wu, J. Gu, Y. Liu, C. Dong, Y. Qiao, and C. C. Loy, "ESRGAN: Enhanced super-resolution generative adversarial networks," in *The European Conference on Computer Vision Workshops (ECCVW)*, September 2018.

[41] J. Yu, Z. Lin, J. Yang, X. Shen, X. Lu, and T. S. Huang, "Generative image inpainting with contextual attention," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[42] S. Guadarrama, R. Dahl, D. Bieber, J. Shlens, M. Norouzi, and K. Murphy, "PixColor: Pixel recursive colorization," in *Proceedings of the British Machine Vision Conference (BMVC)*, 2017.

[43] A. Krizhevsky, "Learning multiple layers of features from tiny images," Canadian Institute for Advanced Research, Tech. Rep., 2009.

[44] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, Z. Huang, A. Karpathy, A. Khosla, M. Bernstein, A. C. Berg, and F.-F. Li, "ImageNet Large Scale Visual Recognition Challenge," *International Journal of Computer Vision (IJCV)*, vol. 115, no. 3, pp. 211–252, 2015.

[45] G. Huang, Z. Liu, L. van der Maaten, and K. Q. Weinberger, "Densely connected convolutional networks," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[46] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *International Conference on Machine Learning (ICML)*, 2010.

[47] D. Ulyanov, A. Vedaldi, and V. Lempitsky, "Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[48] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2015.

[49] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, no. 56, pp. 1929–1958, 2014.

[50] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International Conference on Machine Learning (ICML)*, 2015.

[51] A. L. Mass, A. Y. Hannun, and A. Y. Ng, "Rectifier nonlinearities improve neural network acoustic models," in *International Conference on Machine Learning (ICML)*, 2013.

[52] E. Agustsson and R. Timofte, "NTIRE 2017 challenge on single image super-resolution: dataset and study," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, 2017.

[53] R. Näsänen, "Visibility of halftone dot textures," *IEEE Trans. on Syst., Man, and Cybernetics*, vol. SMC-14, no. 6, pp. 920–924, 1984.

[54] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[55] D. P. Kingma and J. Ba, “Adam: a method for stochastic optimization,” in *International Conference on Learning Representations (ICLR)*, 2015.

[56] R. Eschbach and K. T. Knox, “Error-diffusion algorithm with edge enhancement,” *Journal of the Optical Society of America A*, vol. 8, no. 12, pp. 1844–1850, 1991.

[57] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, “PyTorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems*, 2019.

Author Biography

Baekdu Choi received his B.Sc. in electrical and computer engineering from Seoul National University, Seoul, South Korea in 2017 and is currently working on a Ph.D. in electrical and computer engineering at Purdue University, West Lafayette, IN, USA. His research mainly focuses on digital image processing, digital halftoning and color management for inkjet printers.

Jan P. Allebach is Hewlett-Packard Distinguished Professor of Electrical and Computer Engineering at Purdue University. Allebach is a Fellow of the IEEE, the National Academy of Inventors, the Society for Imaging Science and Technology (IS&T), and SPIE. He was named Electronic Imaging Scientist of the Year by IS&T and SPIE, and was named Honorary Member of IS&T, the highest award that IS&T bestows. He has received the IEEE Daniel E. Noble Award, the IS&T/OSA Edwin Land Medal, and the IS&T Johann Gutenberg Prize. He is a member of the National Academy of Engineering.