

Efficient Halftoning via Deep Reinforcement Learning

Haitian Jiang, Dongliang Xiong, Xiaowen Jiang, Li Ding, Liang Chen, and Kai Huang

Abstract—Halftoning aims to reproduce a continuous-tone image with pixels whose intensities are constrained to two discrete levels. This technique has been deployed on every printer, and the majority of them adopt fast methods (e.g., ordered dithering, error diffusion) that fail to render structural details, which determine halftone’s quality. Other prior methods of pursuing visual pleasure by searching for the optimal halftone solution, on the contrary, suffer from their high computational cost. In this paper, we propose a fast and structure-aware halftoning method via a data-driven approach. Specifically, we formulate halftoning as a reinforcement learning problem, in which each binary pixel’s value is regarded as an action chosen by a virtual agent with a shared fully convolutional neural network (CNN) policy. In the offline phase, an effective gradient estimator is utilized to train the agents in producing high-quality halftones in one action step. Then, halftones can be generated online by one fast CNN inference. Besides, we propose a novel anisotropy suppressing loss function, which brings the desirable blue-noise property. Finally, we find that optimizing SSIM could result in holes in flat areas, which can be avoided by weighting the metric with the contone’s contrast map. Experiments show that our framework can effectively train a light-weight CNN, which is 15x faster than previous structure-aware methods, to generate blue-noise halftones with satisfactory visual quality. We also present a prototype of deep multitoning to demonstrate the extensibility of our method.

Index Terms—Halftoning, dithering, deep learning, reinforcement learning, blue noise.

I. INTRODUCTION

DIGITAL halftoning is the technique that converts continuous-tone images into images whose pixel values are limited to two discrete levels (black and white), which is the limitation of some rendering devices like printers. Thanks to the low-pass filtering property of our human visual system (HVS), a halftone image can be perceived as its contone counterpart from a sufficient distance. According to the dot clustering style, halftone images can be classified into clustered-dot and dispersed-dot. Besides, there are two types of halftone textures: periodic and aperiodic [1]. In this paper, we focus on halftones with *dispersed-dot* and *aperiodic* textures, which has been termed the blue-noise property [2], since this kind of pattern gives the best visual pleasure.

Manuscript received xx xx xxxx; revised xx xx xxxx. This work was supported by the National Key R&D Program of China (2021YFB2206200). (Corresponding author: Kai Huang.)

Haitian Jiang, Dongliang Xiong, Xiaowen Jiang and Kai Huang are with the Institute of VLSI Design, Zhejiang University, Hangzhou, China (e-mail: jianghaitian@zju.edu.cn, xiongd@zju.edu.cn, xiaowen_jiang@zju.edu.cn, huangk@zju.edu.cn).

Li Ding and Liang Chen are with Apex Microelectronics Co.,Ltd., Zhuhai, China (e-mail: ding@apexmic.com.cn, liang.chen@apexmic.com.cn).

In halftoning research, image quality and processing efficiency are of paramount concern. With that in mind, we first briefly review three kinds of “traditional” halftoning techniques: (1) Ordered dithering (screening) methods [3]–[6] periodically split the contone image into tiles and compare them with a dither array, which was designed or generated in advance. These methods are the fastest due to the high parallelism and less computation, but the quality of the resultant halftone is usually unsatisfactory. (2) Error diffusion methods [2], [7]–[14] quantize pixels in sequence. At each step, the quantization error is diffused to adjacent unprocessed pixels with the predefined or input-dependent weights. In general, error diffusion yields better halftones than ordered dithering, although the sequential processing style makes it difficult to compute in parallel and may introduce unpleasant artifacts. (3) Search-based methods [15]–[17] regard halftoning as an optimization problem. They first design a halftone quality metric, which usually explicitly considers the HVS model [18], then optimize it with heuristic algorithms like greedy search [15] or simulated annealing [16]. Search-based methods produce the best halftone quality out of the three types because they can directly optimize an elaborate halftone assessing metric (e.g., structural similarity [19]). However, the computational cost of the searching is usually very expensive.

In the wave of deep learning, inverse halftoning has greatly benefitted from the data-driven methodology [20], [21]. However, there are much fewer discussions [20], [22]–[24] with respect to the deep “forward halftoning”. In this work, we propose a new data-driven halftoning method that can generate stochastic halftone images with structural details while remaining efficient. Instead of searching just-in-time for an optimal solution of this high-dimensional combinatorial optimization problem, we train a light-weight fully convolutional neural network (CNN, with parameters θ) in advance by optimizing the *expectation* of the halftone metric. At runtime, a halftone image can be generated quickly by one CNN inference.

In training such a halftoning CNN, we find that the key difficulty of the learning lies in the final thresholding operation, whose derivative is zero almost everywhere that hinders the back-propagation of gradients. In addition, there is no naturally existing ground-truth halftones available for learning. Rather than learning from a prepared halftone dataset [22], [24] or using the empirical straight-through estimator (STE) [25] and the binarization loss as in [23], we formulate halftoning as a one-step multi-agent reinforcement learning (MARL) problem. Specifically, each binary pixel in the output halftone

TABLE I
COMPARISON OF HALFTONING FORMULATIONS. \mathbf{h} AND \mathbf{c} DENOTE HALFTONE AND CONTONE IMAGES, RESPECTIVELY.

Formulation	Offline Phase	Online Phase	Parallelism	Metric Optimizing
Ordered Dithering	Design/generate dither array	Pixel-wise thresholding with the array	full	\times
Error Diffusion	Design/generate diffusion weights	Diffuse binarizing error pixel-by-pixel	limited	\times
Search-based	-	$\mathbf{h} = \mathbf{h}^* = \arg \min_{\mathbf{h}} \text{Metric}(\mathbf{h}, \mathbf{c})$	limited	\checkmark
DRL-based (ours)	$\theta^* = \arg \min_{\theta} \mathbb{E}_{\mathbf{h} \sim \text{Bernoulli}(\text{CNN}(\mathbf{c}; \theta))} [\text{Metric}(\mathbf{h}, \mathbf{c})]$	$\mathbf{h} = \text{Thresholding}(\text{CNN}(\mathbf{c}; \theta^*), 0.5)$	full	\checkmark

is regarded as a stochastic action chosen by a virtual agent at the corresponding position with the shared CNN. To make it possible to render aperiodic halftone patterns, we input the continuous-tone image with a Gaussian noise map [23] to the fully CNN followed by a pixel-wise sigmoid layer. A tailored effective policy gradient estimator is proposed according to the computational characteristics of halftone metrics. The differences between our deep reinforcement learning (DRL) based formulation and previous methods are summarized in Table I. Moreover, in order to achieve the desirable blue-noise property, we design a novel loss function that suppresses the anisotropy of constant grayscale images' halftones in the frequency domain. Last but not least, we find that the structural similarity index measure (SSIM) [19] used in [16] is actually deficient in optimizing halftone, resulting holes in flat areas with tones near 0/1. To address this problem, we suggest optimizing the contrast-weighted SSIM (CSSIM), which weights the SSIM map with the contrast map of the continuous-tone image.

Extensive experimental results show the significance of our method. We demonstrate that the proposed framework can enable a standard and light-weight CNN model, which is fully parallelizable and can be easily accelerated by modern deep learning HW/SW stacks (15x faster than prior structure-aware halftoning methods), to generate halftones with high quality. The trained model achieves the best SSIM/CSSIM scores compared to existing methods, and the generated halftones can well preserve structural details and possess the blue-noise property. Besides, the training converges quickly (about half a day on one consumer GPU). The extensibility of our method is demonstrated by an example of extending to multitoning, in which the quantization level number is more than two.

Our contributions are summarized as follows:

- We propose an efficient halftoning method via deep reinforcement learning. A fully convolutional neural network is trained to generate halftone images with structural details. The training converges fast thanks to the gradient estimator tailored according to the computational property of halftone assessing metrics.
- We propose the anisotropy suppressing loss function, which leads to the desirable blue-noise property.
- We point out that the SSIM metric has drawbacks in assessing halftones. We suggest optimizing CSSIM that weights the SSIM reward map with local contrast values of the continuous-tone image.
- Our method may facilitate more situations similar to the

basic binary halftoning. A demonstration of extending to multitoning is shown in this paper.

The foundation of this work was reported in our preliminary study [26], named HALFTONERS. On that basis, in the present paper, we provide three significant improvements: First, we discuss the necessity of our DRL-based halftoning solution, establish connections to existing works [15], [27]–[29], and elaborate on how we design the framework in a step-by-step manner. Second, we point out a fatal flaw in an existing halftone metric and solve it with a new weighting scheme. Finally, we add more extensive experimental results, including component analyses, hyperparameter analyses, more comparisons [10], [12], [13], [16], [24], and a multitoning prototype demonstrating our framework's extensibility.

II. PRELIMINARIES: DRL FOR IMAGE PROCESSING

In this section, we first discuss why we choose DRL for halftoning. Then, a brief introduction of existing pixel-level DRL works for image processing is given.

A. Why DRL?

Supervised learning (SL), unsupervised learning (UL), and reinforcement learning (RL) are three common paradigms in deep learning. Most image processing tasks that benefit from the data-driven methodology have adopted the former two, learning from a paired or unpaired image dataset. In this work, we investigate how to train an efficient halftoning NN, which can dither an image by one forward computation. However, we find that the common SL and UL are undesirable here: (1) Preparing the “ground truth” halftones can be costly or even infeasible. It's extremely expensive to obtain the optimal halftone image due to its NP-hard nature. Existing search-based methods can only optimize certain metrics [15] or utilize meta-heuristic search methods [16] that needs to be tuned per-instance [23]. (2) It is non-trivial to transfer the (sub-)optimality from the halftone dataset to the neural network. Halftoning is an one-to-many mapping problem in essence [23]. Directly using a pixel-wise distance loss like the cross-entropy to optimize this ill-posed problem will actually learn an average image [30], [31], which violates halftone's discreteness constraint¹. The generative adversarial network (GAN), which is a prevalent framework of UL, learns

¹Theoretically, autoregressive models [30] can tackle this problem by modeling the dependency between discrete pixels. However, such models require a large amount of runtime [24] due to their iterative processing nature.

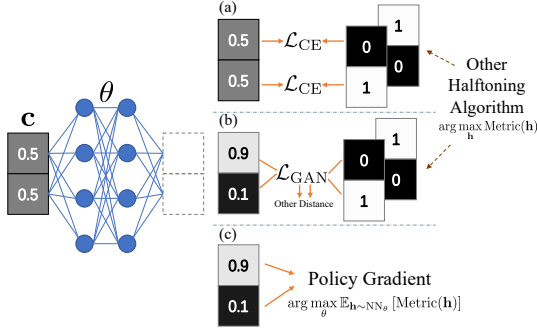


Fig. 1. An illustration of adapting three common learning paradigms to halftoning. Here we dither a continuous-tone “image”: $[0.5; 0.5]$, whose optimal halftones are $[0; 1]$ and $[1; 0]$. (a) Supervised learning. Simply adopting pixel-wise distance loss like the cross-entropy \mathcal{L}_{CE} fails to converge to the two discrete ends. (b) Unsupervised learning. GANs minimize another distance between the generated distribution and a pre-prepared target dataset. (c) Reinforcement learning. The policy network is directly trained to maximize the expectation of the target halftone metric.

from a pre-prepared target halftone dataset [20], [22], [24]. The discriminators in GANs are trained to measure another distance (e.g., the Pearson χ^2 divergence used in LSGAN [24], [32]), which indirectly points to the ultimate halftone metric. Elaborate designs are also needed to stabilize the training [24]. Fig. 1(a) and (b) illustrate these two situations.

Another approach to learn a dither network is to relax the discrete constraint [33] and add a binarizing penalty loss which pushes the output values to their nearest discrete ends, as [23] did. However, such a greedy binarizing rule could damage the optimization from a global perspective.

By contrast, we find that RL is a more appropriate paradigm for halftoning, especially the DRL combined with the powerful deep policy network. In DRL, parameterized *agent(s)* learns to maximize the accumulated *reward* by interacting with the *environment* [34]. The *actions* that agents perform can be discrete, and the expected target metric (reward) can be directly and explicitly optimized by the policy gradient (see Fig. 1(c)). Besides, no additional ground-truth dataset is needed here.

B. Pixel-level DRL

There have recently been works that use DRL to solve image processing problems [27], [35], [36]. While some methods learn to execute global actions on the entire image [35], [37], we are chiefly interested in the one whose actions are formulated at the pixel level: pixelRL, which was proposed by Furuta *et al.* [27]. Herein, we make a brief introduction.

In pixelRL, agents iteratively improve an image by performing actions. Specifically, each pixel has a virtual agent, whose stochastic policy is denoted as $\Pr(h_a^{(t)} | s_a^{(t)}) = \pi_a(h_a^{(t)} | s_a^{(t)})$, where $s_a^{(t)}$ and $h_a^{(t)}$ are the state and action of the a -th agent at the time step t , respectively. There are N agents ($\pi = \{\pi_1, \dots, \pi_N\}$) in total, which is also the number of pixels. All agents share one fully convolutional network (FCN) instead of N individual networks, so the learning can be computationally practical. In the beginning $t = 0$, the state $s^{(0)}$ is set to an initial solution with poor quality (e.g., the original noisy image in the image denoising task). At each time step t ,

agents learn to take actions $\mathbf{h}^{(t)} = \{h_1^{(t)}, \dots, h_N^{(t)}\}$ from a pre-defined toolbox \mathcal{H} consisting of local operators, such as low-pass filtering and pixel value plus one, to improve the current $s^{(t)}$. Then, the agents obtain the next state $s^{(t+1)}$ and rewards $\mathbf{r}^{(t)} = \{r_1^{(t)}, \dots, r_N^{(t)}\}$ from the environment. The ultimate goal of the training is to maximize the expected cumulative rewards $\mathbf{R}^{(t)} = \sum_t \gamma^t \mathbf{r}^{(t)}$ that agents receive in the long run:

$$J(\theta) = \mathbb{E}_{\pi_\theta} \left[\frac{1}{N} \sum_{a=1}^N R_a^{(0)} \right] = \mathbb{E}_{\pi_\theta} \left[\frac{1}{N} \sum_{a=1}^N \sum_{t=0}^{\infty} \gamma^t r_a^{(t)} \right], \quad (1)$$

where γ is the discount rate, and θ denotes FCN’s parameters.

Policy gradient methods are utilized to maximize $J(\theta)$ by taking steps $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$ iteratively (α is the learning rate). According to the policy gradient theorem [34], the gradient of $J(\theta)$ with respect to θ equals:

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\frac{1}{N} \sum_{a=1}^N \sum_{t=0}^{\infty} \nabla_\theta \log \pi_a(h_a^{(t)} | s_a^{(t)}; \theta) R_a^{(t)} \right]. \quad (2)$$

The one-sample Monte Carlo estimation of Eq. (2) is the well-known REINFORCE algorithm [38]. Although it is unbiased, this estimator has a large variance, leading to unstable training. To stabilize the learning, [27] extended the asynchronous advantage actor-critic (A3C) [39] for the pixelRL problem. Specifically, a critic network (a FCN with parameters ϕ) is introduced to predict the value function $V(s^{(t)}) = \mathbb{E}_{\pi_\theta} [R^{(t)} | s^{(t)}]$, which serves as a *baseline* $b(s^{(t)}) = V(s^{(t)})$ to reduce the variance in the sample estimate for the policy gradient:

$$R_a^{(t)} = r_a^{(t)} + \gamma V(s_a^{(t+1)}; \phi), \quad (3)$$

$$A(h_a^{(t)}, s_a^{(t)}) = R_a^{(t)} - V(s_a^{(t)}; \phi), \quad (4)$$

$$\nabla_\theta J(\theta) = \mathbb{E}_{\pi_\theta} \left[\frac{1}{N} \sum_{a=1}^N \sum_{t=0}^{\infty} \nabla_\theta \log \pi_a(h_a^{(t)} | s_a^{(t)}; \theta) A(h_a^{(t)}, s_a^{(t)}) \right]. \quad (5)$$

The critic network is dynamically updated to track the current policy network π_θ :

$$\phi \leftarrow \phi - \alpha \frac{1}{N} \sum_{a=1}^N \nabla_\phi \left(R_a^{(t)} - V(s_a^{(t)}; \phi) \right)^2. \quad (6)$$

The pixelRL setting is interpretable because the agents’ choices in each time step can be observed and analyzed. For more details of pixelRL, please see [27]. Next, we shall formally confront the halftoning problem and analyze our DRL-based method’s benefits.

III. HALFTONING VIA DRL

Given a continuous-tone image $\mathbf{c} \in \mathcal{C}$, a search-based halftoning algorithm [15], [16] generates a halftone image $\mathbf{h} \in \{0, 1\}^N$ by minimizing the designed quantitative visual error $E(\mathbf{h}, \mathbf{c})$, where \mathcal{C} is the contone image dataset, N is the number of pixels, and $E(\cdot, \cdot)$ is a predefined error metric. Binary pixels $h_a \in \{0, 1\}$ compose \mathbf{h} , in which $a \in \{1, \dots, N\}$ identifies the pixel index. Instead of performing individual computationally expensive searching for one image at runtime,

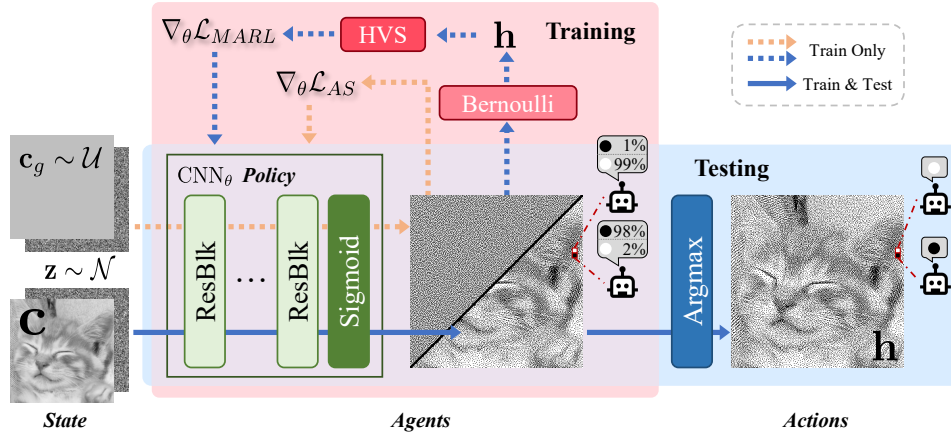


Fig. 2. An overview of our proposed deep half-toning framework. The computation of the half-tone metric and the analysis of anisotropy only exist in the training phase. The half-tone image can be generated by one CNN inference with the trained model.

we *amortize* the cost by training a deep neural network to perform half-toning. Similar to the pixelRL [27], we formulate the half-toning problem as a fully cooperative multi-agent reinforcement learning problem. But here's the key difference: in our formulation, all agents act in just one step, so the half-toning can be done by one NN forward computation. In other words, the first set of actions made by the agents is the final half-tone image, and there is no need for iterative improvements. This single-step setting also enables a simple yet effective training scheme, which we shall show later in this section. In the training phase (offline), agents are trained to generate high quality halftones by the RL algorithm. Then, in the testing phase, the half-tone image is generated by one inference, which can be fast if a light-weight model is adopted (see Fig. 2). Now we cast half-toning into the RL terminology.

State. The environment state s is defined as:

$$s = \text{Concatenate}(\mathbf{c}, \mathbf{z}), \quad (7)$$

where \mathbf{z} is a dynamically sampled white Gaussian noise map enabling a CNN to dither constant grayness images [23].

Agents. A virtual agent with a shared policy π is constructed at each pixel's position, and there are N agents in total. Specifically, we utilize a FCN as the policy network like the pixelRL [27]. After observing the state s_a (the receptive field of an output pixel a) and communicating with other agents during the CNN forward pass, agent at position a gives its probabilities of actions:

$$\Pr(h_a | \mathbf{c}, \mathbf{z}) = \pi_a(h_a | \mathbf{c}, \mathbf{z}; \theta), \quad (8)$$

where θ denotes CNN's parameters. The joint policy is denoted as $\pi = \{\pi_1, \dots, \pi_N\}$. We think that the noise map \mathbf{z} , which is proposed by [20] and [23], plays as a disentangled latent variable or an image-agnostic "plan". It indicates the specific half-tone image that should be predicted in this multimodal problem, which means the half-tone pixels are independent of each other conditioning on \mathbf{c} and \mathbf{z} :

$$\pi(\mathbf{h} | \mathbf{c}, \mathbf{z}; \theta) = \prod_a \pi_a(h_a | \mathbf{c}, \mathbf{z}; \theta). \quad (9)$$

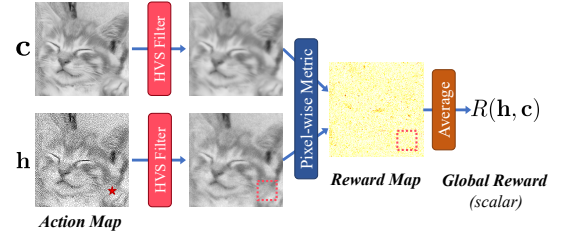


Fig. 3. Illustration of half-tone assessing steps and the reward map. An agent's action (marked by "★") can only affect the rewards in the window around it, whose size is the same as the HVS filter.

In other words, actions (pixels) can be chosen simultaneously by agents in both training and testing. On the contrary, error diffusion and search-based methods process pixels in an autoregressive-like manner [30]: $\Pr(\mathbf{h} | \mathbf{c}) = \prod_{i=1}^N \Pr(h_i | \mathbf{h}_{<i}, \mathbf{c})$, whose parallelism is extremely limited.

Rewards. In half-tone assessment, existing metrics (MSE and SSIM) consist of three steps (shown in Fig. 3): (1) Filter \mathbf{h} and \mathbf{c} using the HVS model. (2) Calculate the pixel-wise error map \mathbf{e} . (3) Calculate the scalar metric by averaging the error map. Following RL terminology, we define the reward, which we would like to maximize, as the negative of the error. So there is a reward map \mathbf{r} , and the *global* reward R equals:

$$R(\mathbf{h}, \mathbf{c}) = \frac{1}{N} \sum_{a=1}^N r_a = -E(\mathbf{h}, \mathbf{c}) = \frac{1}{N} \sum_{a=1}^N -e_a. \quad (10)$$

Strictly speaking, an agent's action will affect all the reward values in the local window around it (see the red dashed box in Fig. 3), whose size is decided by the HVS filter. However, since HVS model is usually a low-pass filter, one agent should bear more of the responsibility for those reward values that are closer to it. This is called the multi-agent credit assignment problem in the RL literature [40]. The scalar performance measure that we want to maximize is:

$$J(\theta) = \mathbb{E}_{\mathbf{c}, \mathbf{z}} [\mathbb{E}_{\mathbf{h} \sim \pi(\mathbf{h} | \mathbf{c}, \mathbf{z}; \theta)} [R(\mathbf{h}, \mathbf{c})]]. \quad (11)$$

Since there is only one decision step in our formulation, Eq. (11) does not involve the integration over time steps, which

is different to Eq. (1). The specific form of the reward function $R(\cdot, \cdot)$ will be discussed in Section V.

Learning. In order to maximize $J(\theta)$ (Eq. (11)), we adopt policy gradient-based methods. The gradient equals:

$$\begin{aligned} \nabla_{\theta} J(\theta) &= \nabla_{\theta} \mathbb{E}_{\mathbf{c}, \mathbf{z}} [\mathbb{E}_{\mathbf{h} \sim \pi(\mathbf{h}|\mathbf{c}, \mathbf{z}; \theta)} [R(\mathbf{h}, \mathbf{c})]] \\ &= \mathbb{E}_{\mathbf{c}, \mathbf{z}} \left[\sum_{\mathbf{h}} \nabla_{\theta} \pi(\mathbf{h}|\mathbf{c}, \mathbf{z}; \theta) R(\mathbf{h}, \mathbf{c}) \right]. \end{aligned} \quad (12)$$

We estimate this expectation by taking B samples (mini-batch) from the dataset \mathcal{C} and the multi-variate Gaussian prior distribution:

$$\nabla_{\theta} J(\theta) \approx \frac{1}{B} \sum_{i=1}^B g(\mathbf{c}^{(i)}, \mathbf{z}^{(i)}), \quad (13)$$

$$g(\mathbf{c}, \mathbf{z}) = \sum_{\mathbf{h}} \nabla_{\theta} \pi(\mathbf{h}|\mathbf{c}, \mathbf{z}; \theta) R(\mathbf{h}, \mathbf{c}), \quad (14)$$

where $\mathbf{c}^{(i)} \sim \mathcal{C}$ and $\mathbf{z}^{(i)} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$. However, the exact integration of $g(\mathbf{c}, \mathbf{z})$ (Eq. (14)) is still intractable (with $O(2^N)$ complexity of reward computation). Therefore, we use a gradient estimator $\hat{g} \approx g$ via Monte Carlo estimation [41].

Next, we will discuss how we design \hat{g} . To keep the notation simple, we leave it implicit that π is a function of $\mathbf{c}, \mathbf{z}, \theta$; R, g are functions of \mathbf{c}, \mathbf{z} ; and the gradient is with respect to θ .

A. REINFORCE with Counterfactual Baseline

The REINFORCE [38] algorithm used in Eq. (2) is still feasible. However, here we do not need the policy gradient theorem [34] due to the single step setting, and the gradient estimator can be directly derived by the log derivative trick:

$$\begin{aligned} g &= \sum_{\mathbf{h}} \nabla \pi(\mathbf{h}) R(\mathbf{h}) \\ &= \sum_{\mathbf{h}} \pi(\mathbf{h}) \nabla \log \pi(\mathbf{h}) R(\mathbf{h}) \\ &= \mathbb{E}_{\mathbf{h} \sim \pi} \left[\nabla \left(\log \prod_a \pi_a(h_a) \right) R(\mathbf{h}) \right] \\ &= \mathbb{E}_{\mathbf{h} \sim \pi} \left[\sum_a \nabla \log \pi_a(h_a) (R(\mathbf{h}) - b_a) \right]. \end{aligned} \quad (15)$$

In pixelRL [27], a critic network [39] is adopted to predict the expected accumulated reward (value function) as the baseline b_a , and this additional model is simultaneously updated within the training procedure (see Eq. (4) and (6)). But here we find that it is possible to directly calculate a more precise baseline. Specifically, we introduce the counterfactual multi-agent (COMA) baseline [28]:

$$b_a = b_a(\mathbf{h}_{-a}) = \sum_{h'_a} \pi_a(h'_a) R(\{h'_a, \mathbf{h}_{-a}\}), \quad (16)$$

where $\mathbf{h}_{-a} = \{h_1, \dots, h_{a-1}, h_{a+1}, \dots, h_N\}$. Intuitively, this baseline $b_a(\mathbf{h}_{-a})$ indicates agent a 's average performance by marginalizing its action h'_a , while keeping other agents' actions \mathbf{h}_{-a} fixed. It is computationally feasible in the halftoning problem, as the dimension of a single agent's action space is only 2 (black and white) and the reward function has locality

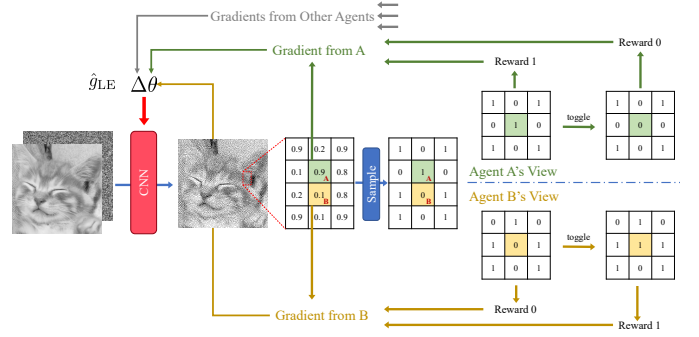


Fig. 4. Illustration of \hat{g}_{LE} (Eq. (19)).

(see the window in Fig. 3). More importantly, it elegantly tackles the credit assignment problem aforementioned by counterfactual thinking: “what if an agent had chosen the other action”. Combining Eq. (15) and (16), the one-sample ($\mathbf{h} \sim \pi$) COMA gradient estimator is:

$$\hat{g}_{COMA} = \sum_a \nabla \log \pi_a(h_a) (R(\mathbf{h}) - b_a(\mathbf{h}_{-a})). \quad (17)$$

B. Local Expectation Gradient Estimator

The gradient estimator \hat{g}_{COMA} (Eq. (17)) is a feasible solution with one sample. However, we notice that the COMA baseline (Eq. (16)) actually has explored N extra non-duplicate data points. It may reduce more variance, if we directly utilize these data points to estimate Eq. (14), rather than regard them as a part of the baseline. Following this idea, we derive a new estimator with $N + 1$ evaluation budgets of $R(\cdot)$:

$$\begin{aligned} g &= \sum_{\mathbf{h}} \nabla \pi(\mathbf{h}) R(\mathbf{h}) \\ &= \sum_{\mathbf{h}} \sum_a \nabla \pi_a(h_a) \pi_{-a}(\mathbf{h}_{-a}) R(\{h_a, \mathbf{h}_{-a}\}) \\ &= \sum_a \sum_{h_a} \sum_{\mathbf{h}_{-a}} \nabla \pi_a(h_a) \pi_{-a}(\mathbf{h}_{-a}) R(\{h_a, \mathbf{h}_{-a}\}) \sum_{h'_a} \pi_a(h'_a) \\ &= \sum_a \sum_{h'_a} \sum_{\mathbf{h}_{-a}} \pi_a(h'_a) \pi_{-a}(\mathbf{h}_{-a}) \sum_{h_a} \nabla \pi_a(h_a) R(\{h_a, \mathbf{h}_{-a}\}) \\ &= \sum_{\mathbf{h}'} \pi(\mathbf{h}') \sum_a \sum_{h_a} \nabla \pi_a(h_a) R(\{h_a, \mathbf{h}'_{-a}\}) \\ &= \mathbb{E}_{\mathbf{h} \sim \pi} \left[\sum_a \sum_{h'_a} \nabla \pi_a(h'_a) R(\{h'_a, \mathbf{h}_{-a}\}) \right]. \end{aligned} \quad (18)$$

This is similar to [29], which was proposed to estimate the gradient of the evidence lower bound in variational inference problems. Despite there being some differences (e.g., our agents are independent conditioning on the shared θ , while the \mathbf{h} was represented as a directed graph in their work), we adopt their name: local expectation gradient, since the spirit under the hood is similar: calculating a local exact expectation while using a single sample from the other variables. The corresponding one-sample estimator is:

$$\hat{g}_{LE} = \sum_a \sum_{h'_a} \nabla \pi_a(h'_a) R(\{h'_a, \mathbf{h}_{-a}\}), \quad (19)$$

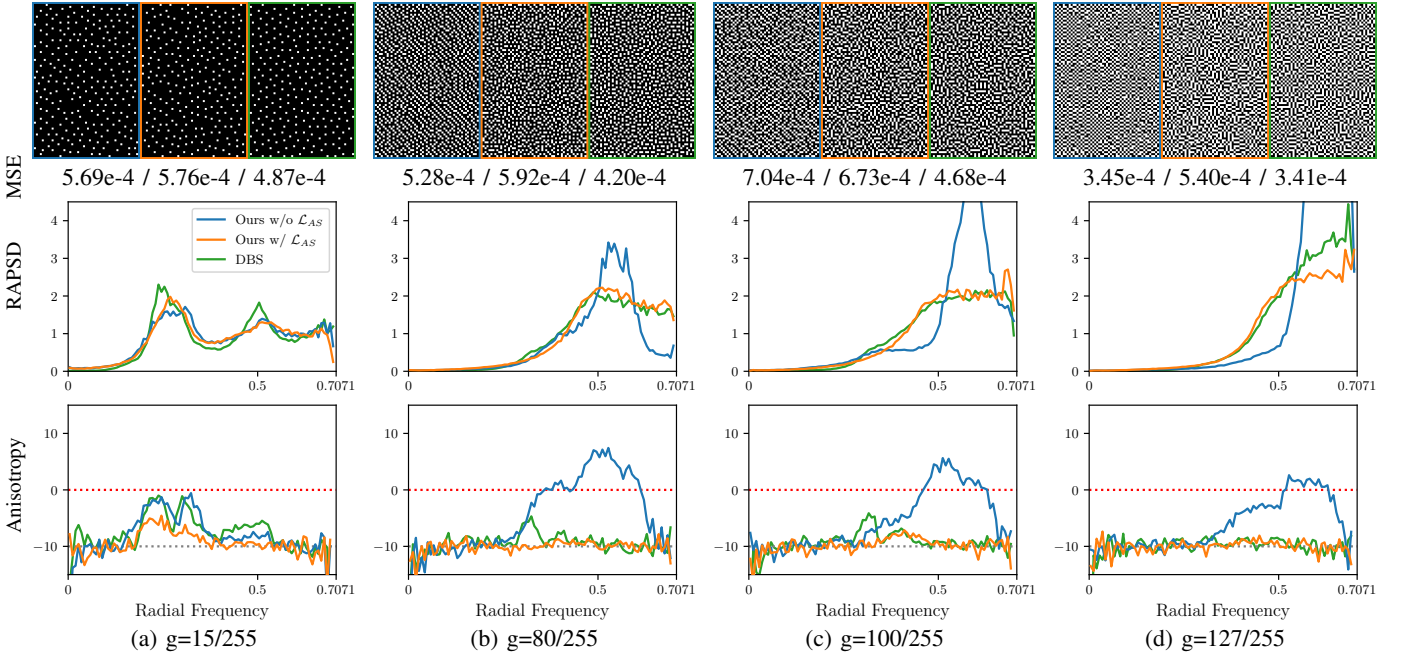


Fig. 5. Spectral analyses on our halftone results generated without \mathcal{L}_{AS} and with \mathcal{L}_{AS} . The corresponding MSE scores measured on the test set are $5.01\text{e-}4$ and $5.51\text{e-}4$, respectively. Results from the DBS algorithm [15] are also included here for reference. Halftone samples in each column: (left) ours w/o \mathcal{L}_{AS} , (middle) ours w/ \mathcal{L}_{AS} , (right) DBS.

where $\mathbf{h} \sim \pi$. While the computational complexity is the same: $O(N + 1)$, we empirically find that \hat{g}_{LE} gets better optimization results than \hat{g}_{COMA} (see Section VI(D)).

Besides, Eq. (19) has an intuitive explanation. It is similar to performing N concurrent “toggle” operations from the DBS algorithm [15] on the sampled halftone image. But instead of updating the halftone in place, we turn the reward differences into gradients to reinforce the NN (illustrated in Fig. 4). The complete formula is given here, as our final proposal:

$$\nabla_{\theta} \mathcal{L}_{MARL} = -\mathbb{E}_{\mathbf{c}, \mathbf{z}, \mathbf{h} \sim \pi} \left[\sum_a \sum_{h'_a} \nabla \pi_a(h'_a) R(\{h'_a, \mathbf{h}_{-a}\}, \mathbf{c}) \right]. \quad (20)$$

We emphasize that no additional NNs, reference halftone datasets, hyper-parameters, or heuristic binarization rules are needed in $\nabla_{\theta} \mathcal{L}_{MARL}$.

C. Efficient Calculation of The Policy Gradient

One may argue that the $N + 1$ times of $R(\cdot)$ in Eq. (20) are too expensive. Here we point out that it can be efficient when considering the characteristics of halftone’s metrics (see Fig. 3). First, all agents can reuse the same HVS filtered map. Second, an agent only needs to be responsible for the local window around it on the reward map. Third, instead of calculating $R(\cdot)$ from scratch, the *opposite* action’s reward window equals the *current* action’s reward window plus/minus the HVS filter, from each agent’s perspective. In a word, the computation of Eq. (20) is doable and parallelizable, and all the training cost only appears in the offline phase.

IV. ANISOTROPY SUPPRESSING LOSS FUNCTION

Although our proposed DRL-based model enables training CNNs to output discrete halftone images, the desirable blue-noise property [2] has not been explicitly guaranteed. Following classic search-based methods [15], we tested the MSE metric (between the HVS-filtered halftone and HVS-filtered contone) as the reward function in the proposed DRL framework. However, we find that a lower MSE score does not necessarily mean the better blue-noise quality (see Fig. 5). Even worse, the parallelism of CNN makes it easy to create globally consistent stripe and checkerboard-like artifacts.

These phenomena were also reported by Xia *et al.* [23]. In response, they proposed to penalize the low frequency components, separated by the discrete cosine transform, on the dithered constant grayness images. However, we observed that the low-frequency components have been minimized well by the proposed policy gradient. The more urgent defect here is the excessive anisotropy, which was neither explicitly considered by MSE nor SSIM. According to [1], the anisotropy of a constant grayness image’s blue-noise halftone should be minimized to -10dB at all frequencies. Thus we intend to explicitly suppress the anisotropy.

Given the definition of the power spectral estimate $\hat{P}(f)$ with one periodogram and the radially averaged power spectrum density (RAPSD) $P(f_{\rho})$ [2]:

$$\hat{P}(f) \approx \frac{1}{N} |\text{DFT}(\mathbf{h})|^2 \quad (21)$$

$$P(f_{\rho}) = \frac{1}{n(r(f_{\rho}))} \sum_{f \in r(f_{\rho})} \hat{P}(f), \quad (22)$$

where $n(r(f_{\rho}))$ is the number of discrete frequency samples in an annular ring with width $\Delta_{\rho} = 1$ around radial frequency

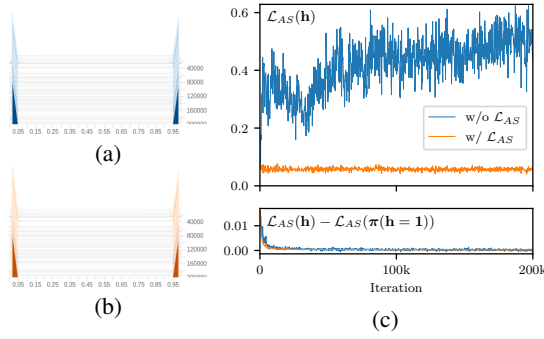


Fig. 6. The histograms of CNN's outputs (analyzed on the Lenna test image) during the training (a) w/o \mathcal{L}_{AS} and (b) w/ \mathcal{L}_{AS} . (c) Curves of \mathcal{L}_{AS} .

f_ρ , the anisotropy is defined as:

$$A(f_\rho) = \frac{1}{n(r(f_\rho)) - 1} \sum_{f \in r(f_\rho)} \frac{(\hat{P}(f) - P(f_\rho))^2}{P^2(f_\rho)}. \quad (23)$$

To achieve the blue-noise property, we propose the anisotropy suppressing loss function:

$$\mathcal{L}_{AS} = \mathbb{E}_{\mathbf{c}, \mathbf{z}} \left[\sum_{f_\rho} \sum_{f \in r(f_\rho)} (\hat{P}(f) - P(f_\rho))^2 \right], \quad (24)$$

which minimizes the numerator of Eq. (23). Note that the calculation of $\hat{P}(f)$ involves the Thresholding(\cdot) operation, which also leads to the gradient vanishing problem. Naturally, one may want to treat Eq. (24) as part of the reward function (Eq. (10)) in the RL framework. But the fatal drawback here is: a single pixel's action switching will lead to changes on the whole spectrum. So the naïve implementation of \hat{g}_{LE} optimizing \mathcal{L}_{AS} will bring $N + 1$ times of anisotropy calculation in every training iteration, which is unacceptable in practice.

To address this problem, we make the following key observation: most of the output values of our network are clustered around the two endpoints in the whole training process (shown in Fig. 6(a)). Therefore the probability map of action “white” should be similar to the real halftone image (after thresholding). According to this observation, we take the differentiable probabilities $\pi(\mathbf{h} = 1)$ as a proxy for \mathbf{h} to estimate the power spectrum in Eq. (21). As the spectral analysis only makes sense for constant grayscale images' halftones, we optimize this loss function on an extra mini-batch of uniformly sampled $\mathbf{c}_g \sim \mathcal{U}(\mathbf{0}, \mathbf{1})$ following [23]. Our experimental results show the artifacts vanish as a result of \mathcal{L}_{AS} . The clustered distribution still holds after adding the new loss function (see Fig. 5(b)).

Now we are able to train a deep blue-noise halftoning model via gradient descending (see Algorithm 1 and Fig. 2):

$$\nabla \mathcal{L}_{total} = \nabla \mathcal{L}_{MARL} + w_a \nabla \mathcal{L}_{AS}, \quad (25)$$

where w_a is a hyper-parameter. The symbols defined in Section III and IV are summarized in Table II. In the next section, we are going to discuss some drawbacks of existing halftone metrics.

Algorithm 1 Halftoning via Deep Reinforcement Learning.

procedure TRAIN(\mathcal{C})

Init θ

repeat

▷ Optimize halftone metrics

Sample $\mathbf{c} \sim \mathcal{C}$, $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$\mathbf{h} \sim \pi(\mathbf{h}|\mathbf{c}, \mathbf{z}; \theta)$

$\mathcal{L}_{MARL} = -\sum_a \sum_{\mathbf{h}'_a} R(\{\mathbf{h}'_a, \mathbf{h}_{-a}\}, \mathbf{c}) \pi_a(\mathbf{h}'_a|\mathbf{c}, \mathbf{z}; \theta)$

▷ Suppress anisotropy

Sample $\mathbf{c}_g \sim \mathcal{U}(\mathbf{0}, \mathbf{1})$, $\mathbf{z}_g \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$\hat{P}_\theta(f) = \frac{1}{N} |\text{DFT}(\pi(\mathbf{h} = 1|\mathbf{c}_g, \mathbf{z}_g; \theta))|^2$

$P_\theta(f_\rho) = \frac{1}{n(r(f_\rho))} \sum_{f \in r(f_\rho)} \hat{P}_\theta(f)$

$\mathcal{L}_{AS} = \sum_{f_\rho} \sum_{f \in r(f_\rho)} (\hat{P}_\theta(f) - P_\theta(f_\rho))^2$

▷ Update weights

$\theta \leftarrow \theta - \alpha(\nabla_\theta \mathcal{L}_{MARL} + w_a \nabla_\theta \mathcal{L}_{AS})$

until Convergence

return θ

procedure TEST(\mathbf{c}, θ)

Sample $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$

$\mathbf{h} = \text{Thresholding}(\text{CNN}_\theta(\mathbf{c}, \mathbf{z}), 0.5)$

return \mathbf{h}

TABLE II
TABLE OF NOTATIONS

Symbol	Meaning
$\Pr(\cdot)$	Probability
\mathbf{h}	Halftone image/actions
\mathbf{c}	Continuous-tone image
\mathcal{C}	Continuous-tone dataset
\mathcal{N}	Gaussian distribution
\mathcal{U}	Uniform distribution
\mathbf{z}	Noise map
\mathbf{s}	State
a	Index of the a^{th} agent/pixel
N	Number of agents/pixels
θ	CNN parameters
π	Policy
$\pi(\mathbf{h} \mathbf{c}, \mathbf{z}; \theta)$	Probability of \mathbf{h} given \mathbf{c} , \mathbf{z} and θ
$E(\cdot, \cdot)$	Error function
$R(\cdot, \cdot)$	Reward function
$J(\theta)$	Performance measure for the policy with θ
b_a	Agent a 's baseline function
\mathcal{L}	Loss function
f	Frequency
$\hat{P}(f)$	Power spectral estimate
f_ρ	Radial frequency
$r(f_\rho)$	Annular rings with center radius f_ρ
$n(r(f_\rho))$	Number of frequency samples in $r(f_\rho)$
$P(f_\rho)$	Radially averaged power spectrum density of f_ρ
$A(f_\rho)$	Anisotropy of f_ρ

V. CONTRAST-WEIGHTED SSIM FOR HALFTONING

To preserve the characteristic look of textured regions in halftone images, Pang *et al.* [16] proposed structure-aware halftoning that takes the structural similarity index measure (SSIM) [19] into consideration. However, we find that the optimizing of the original SSIM metric will cause holes in the smooth area whose values are near 0 or 1 (show in Fig. 7 (a)(b)). On the SSIM index map (Fig. 7(c)), the hole area is assigned with higher scores. By contrast, the area to the left of the hole is given a lower score, which is problematic.

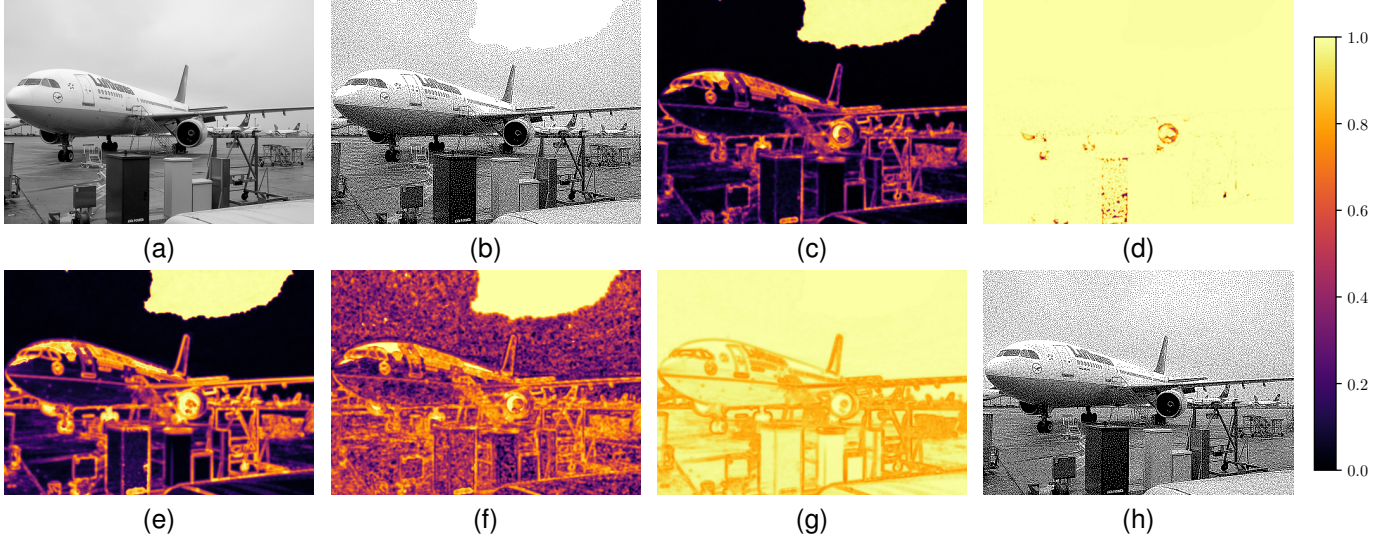


Fig. 7. (a) Continuous-tone image. (b) Halftone image optimizing MSE & SSIM. (c) SSIM index map. (d) Luminance comparison map. (e) Contrast comparison map. (f) Structure comparison map. (g) CSSIM index map. (h) Halftone image optimizing MSE & CSSIM.

To diagnose this problem, we break SSIM into components:

$$\text{SSIM}(\mathbf{h}, \mathbf{c}) = l(\mathbf{h}, \mathbf{c}) \cdot c(\mathbf{h}, \mathbf{c}) \cdot s(\mathbf{h}, \mathbf{c}), \quad (26)$$

where l , c and s denote luminance, contrast and structure comparison maps (see Fig. 7(d)(e)(f)), respectively. It shows that in the flat area, the contrast and structure components of SSIM actually prevent agents from generating minority dots.

As suggested in [19], it is possible to compute a weighted average of the SSIM index map according to the specific application. Recall that the goal of optimizing SSIM in halftoning is to preserve structural details, which should only be performed in the area with abundant textural variation. Motivated by it, we propose to revise the metric by weighting the original SSIM with the continuous-tone image's contrast map σ_c , whose value at position a is:

$$\sigma_a = k \left(\sum_{i \in \text{window}(a)} w_i (c_i - \mu_a)^2 \right)^{\frac{1}{2}}, \quad (27)$$

where $k = 2$ is a normalizing factor, w_i is the i th weight of 11×11 Gaussian kernel with standard deviation of 1.5, and μ_a is the local luminance [19]. So the contrast-weighted SSIM (CSSIM) metric is:

$$\text{CSSIM}(\mathbf{h}, \mathbf{c}) = \sigma_c \cdot \text{SSIM}(\mathbf{h}, \mathbf{c}) + (1 - \sigma_c) \cdot 1. \quad (28)$$

We test this new metric on Fig. 7(b), and it successfully assigns reasonable structural scores to halftones (see Fig. 7(g)).

Finally, we define the reward function $R(\mathbf{h}, \mathbf{c})$ used in \mathcal{L}_{MARL} , which considers both the tone similarity and the structure similarity:

$$R(\mathbf{h}, \mathbf{c}) = -\text{MSE}(\text{HVS}(\mathbf{h}), \text{HVS}(\mathbf{c})) + w_s \text{CSSIM}(\mathbf{h}, \mathbf{c}), \quad (29)$$

where $\text{HVS}(\cdot)$ denotes the low-pass filtering by a HVS model and w_s is a hyper-parameter. Note that our method is not tied to a specific HVS model. One can adopt any HVS model

according to practical requirements, so long as it has a limited filter size. In this paper, without loss of generality, we use Näsänen's HVS model [18] for demonstration since it was recommended in [42]. We use the definition and parameters listed in [42]'s Table I. Besides, the filter size is set to 11×11 . With regard to the scale parameter, which serves as a free parameter [42], we choose $S = 2000$, as it is similar to the situation when looking at a 24-inch 1080p monitor. Fig. 7(h) shows the result generated by our model optimizing the expectation of Eq. (29). One can see that the hole phenomenon has disappeared, and the image presents clear structural details.

VI. EXPERIMENTS

In this section, we are going to show details of our method, compare our work with prior halftoning approaches, analyze the components, and discuss the extensibility.

A. Experiment Settings

Prior Methods. We have selected nine typical halftoning methods from different categories for comparison:

- Ordered dithering. Void-and-cluster (VAC) [5], dither array size = 64×64 .
- Error diffusion. Ostromoukhov's method (OVED) [8]; Structure-aware error diffusion (SAED) [10]; Tone-dependent error diffusion based on an updated blue-noise model (TDED_{BS}) [12]; Simple gradient-based error diffusion (SGED) [13].
- Search-based methods. Direct binary search (DBS) [15]; Structure-aware halftoning (SAH) [16]. For a fair comparison, we use the same optimization objective as in our method (Eq. (29)). 11×11 Näsänen HVS filter with $S = 2000$ when implementing their methods (DBS: MSE; SAH: MSE and CSSIM, $w_s = 0.06$).
- Deep halftoning. To our knowledge, Choi and Allebach's work [24] is the only published paper currently focusing

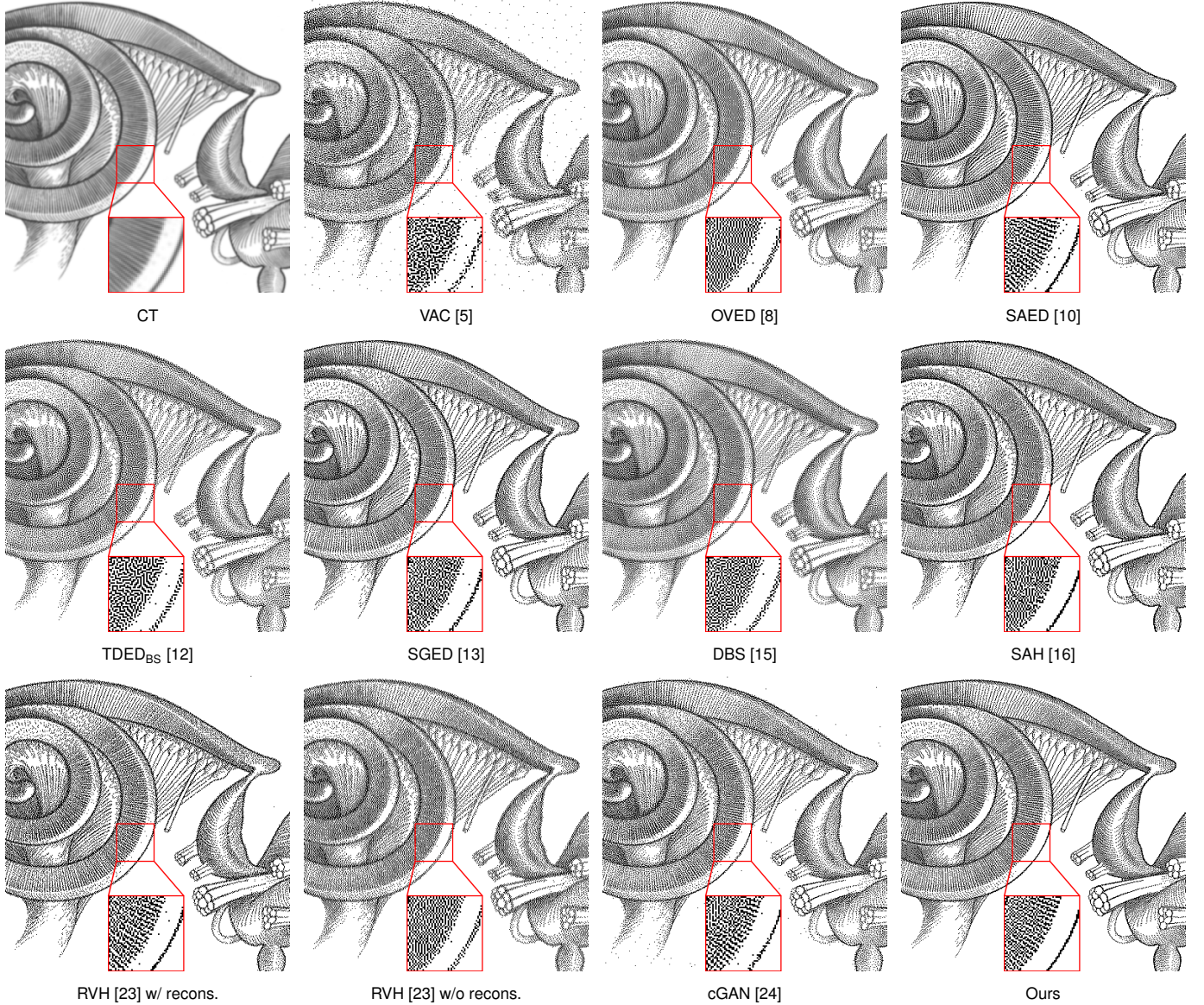


Fig. 8. Halftone samples of image “Snail Shaped Organ”.

on generating aperiodic dispersed-dot halftones with deep models. We implement the conditional GAN (**cGAN**) for comparison since the autoregressive model is, according to their report, very slow. Except for the dataset, all training details strictly follow the settings in [24].

- In addition, we take the reversible halftoning method (**RVH**) [23] for comparison, as it can generate blue-noise halftones. Specifically, we reimplement it by removing the extra reconstruction task that may damage halftones’ quality (only the warm-up stage in their paper). The experimental results of the full version (with reconstruction) are also presented for reference. Except for the warm-up-only version (trained for 45 epochs for convergence), all training details exactly follow their released code.

Dataset. We evaluate all halftoning methods on the VOC2012 dataset [43] following [23]. The test set contains 1,684 images, which are randomly selected from 17,125

images. For learning-based methods (RVH, cGAN and ours), the remaining 13,758 and 1,683 images are reserved as the training set and the validation set, respectively. All images are converted to grayscale in advance. Note that our formulation can be considered as self-supervised, so it should be easy to collect more label-free continuous-tone images.

HW/SW Environment. To take full advantage of the parallelism in some halftoning methods (VAC, SAED, RVH, cGAN, and ours), we implement them on an NVIDIA GeForce RTX 2080Ti GPU with PyTorch 1.11.0, CUDA 11.3 and cuDNN 8.2.0. For those serial methods (OVED, TDDEBS, SGED, DBS and SAH), we implement them in C++ with GCC 6.3.0 and OpenCV 4.6.0 on an Intel Xeon Gold 5215 CPU (2.5GHz).

Details of Our Method. We choose ResNet [44] as the backbone policy network. The CNN has 16 residual blocks and 33 convolutional layers in total. Each convolution kernel has 32 channels, and the strides of all convolution layers

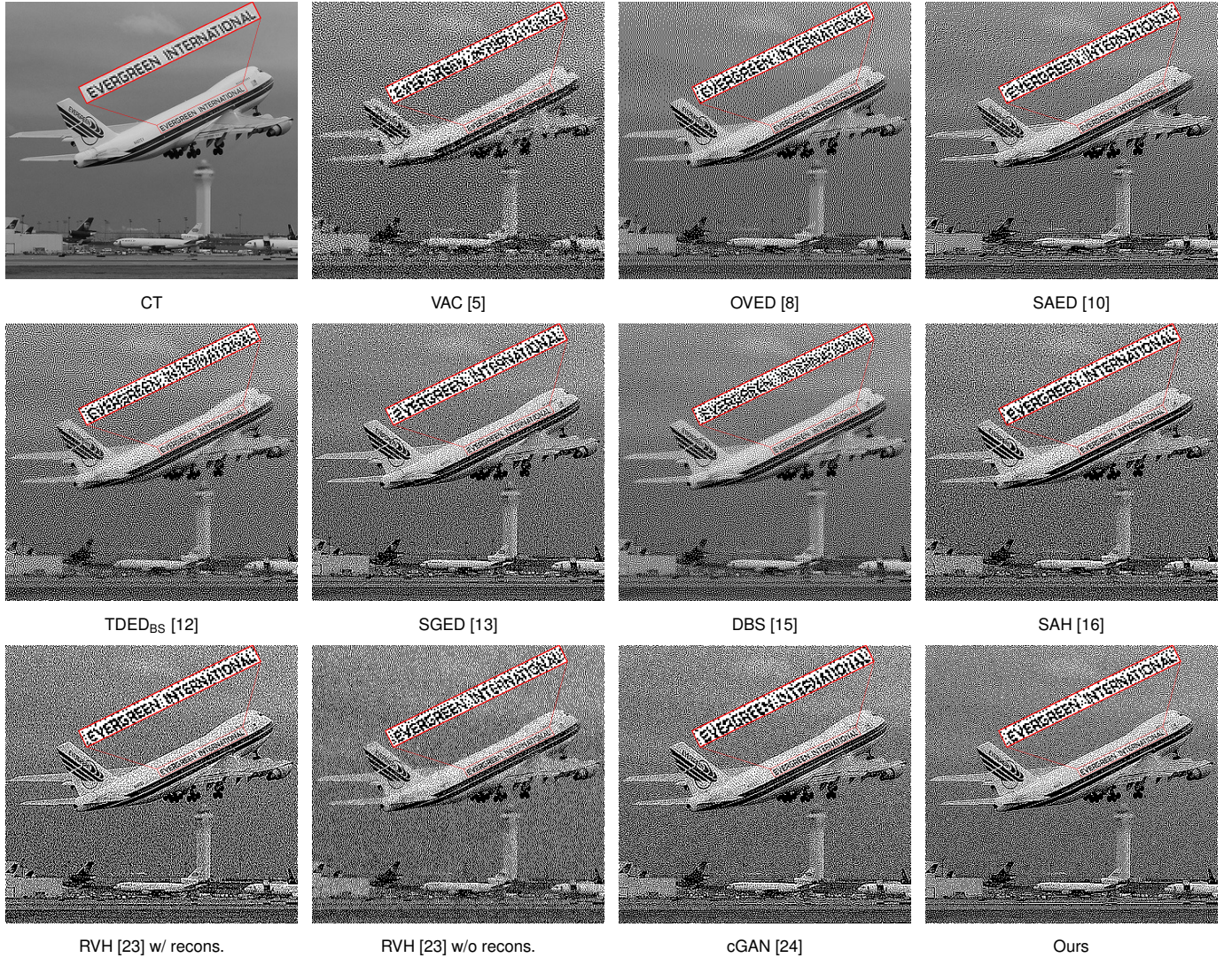


Fig. 9. Halftone samples of image “Plane” (from VOC2012 dataset [43]).

are 1. A pixel-wise sigmoid layer has been appended to the CNN to output the probabilities of white dots $\pi(\mathbf{h} = 1)$. All convolution kernels’ weights are initialized by sampling from the normal distribution $\mathcal{N}(0, 0.01^2)$, and the biases are initialized to zero. This is to set the initial action probabilities around 0.5, so agents can do sufficient exploration rather than be stuck in one random action preference. The batch size is set to 64, training samples are randomly cropped to 64×64 , hyper-parameters $w_s = 0.06$, $w_a = 0.002$, and the learning rate α is adjusted from $3e-4$ to $1e-5$ with the cosine annealing schedule. The model is trained for 200,000 iterations (~ 12 hours on a 2080Ti GPU) with the Adam optimizer [45]. In the testing phase, we process and evaluate images with their original size. We do not pad the images when calculating metrics.

B. Halftone Quality

To compare various methods’ visual quality, we test them on different kinds of images and show three samples here.

Fig. 8 shows the benefit of effectively optimizing halftone metric by our method. Our halftone results keep identi-

fiable structures from the source image. On the contrary, VAC, OVED, TDDED_{BS} and DBS totally destroy the vein pattern because they do not take the structural similarity into consideration. With regard to RVH, we find the halftone with reversibility has better structural details than the one without considering reconstruction. We think this is because the structures have to be maintained in the halftone for the need of reconstruction. The results obtained from cGAN lack structural details, as the dataset used and the loss function employed do not take into account this information. SAED successfully generates halftones with rich structural features. However, there are isolated minority dots near the edges in SAED’s result. This is due to the fact that the accumulated errors are improperly released here by lack of homogeneous receivers. The SAH method, which needs per-instance tuning [23], shows unsatisfactory results in our experiment with the default parameters in [16].

Fig. 9 shows a realistic image and its halftone results. Among all the existing methods, SAED achieves the relatively good quality. However, the text on the plane is not very clear,

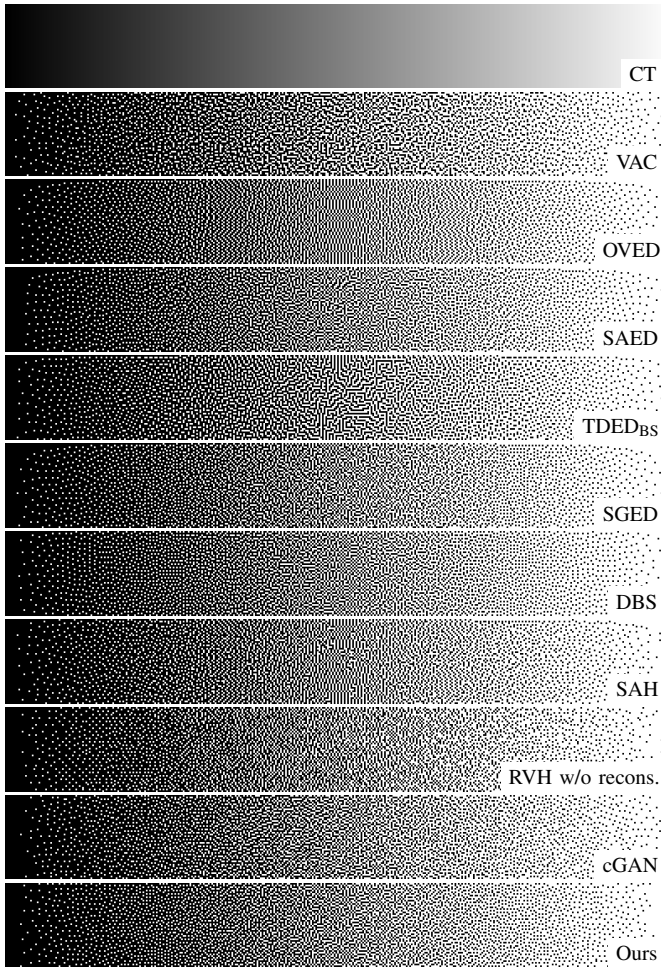


Fig. 10. Halftone samples of grayscale intensity ramp.

which can be blamed on the assumption of only one dominant frequency at each position [10]. In contrast, our method is more adaptive and can render legible structures by learning to optimize the proposed CSSIM metric. Note that all methods except TDED_{BS} have shown a certain degree of checkerboard artifacts in the mid-tone area. In TDED_{BS}, a better noise model has been explicitly considered. For more details of the updated noise model, we refer to [12], [46], [47].

For reference, Fig. 10 shows the results of all methods on the grayscale ramp image. The quality of our halftone result is comparable to the DBS method.

Quantitative quality results of all methods on the test dataset are listed in Table III (mean and standard deviation). The tone consistency is measured by PSNR between the HVS-filtered halftone and the HVS-filtered input contone. In addition to the Näsänen HVS model used in our work, we also report the PSNR scores calculated with the Gaussian filter (sigma=2) for reference. The structural similarity is measured by the SSIM² and the proposed CSSIM (Eq. (28)). Thanks to the effective DRL framework, our model achieves the best SSIM and CSSIM scores among all candidates. Surprisingly, it performs even better than the search-based SAH. With limited searching

budgets, it seems hard for SAH to effectively optimize the target with a generic heuristic searching strategy. With regard to PSNR, DBS achieves the best score by effectively searching [49] for an optimal halftone solution of one contone image on-the-fly. Although halftones with an extremely high PSNR are not necessarily more visually pleasing, it is possible to narrow the gap between DBS and our method by decreasing w_s or increasing the network capacity.

C. Processing Efficiency

The parameter number (predefined dither array, look up tables or trained weights) and the runtime (evaluated on the 512x512 “Lenna” testcase) of all methods are listed in Table III. While the ordered dithering method (VAC) is the fastest, its halftone results are of inferior quality. However, this strategy is still useful in situations where the quality requirements are not strict. Search-based methods (DBS and SAH) are quite time-consuming which impedes its practical application. Compared to them, SAED achieves a better balance between speed and quality. The SGED method is competitive, although it shows limited quality in some complex scenarios (such Fig. 8). RVH’s, cGAN’s and our results prove that it is possible to learn a halftoning network and leave the time-consuming optimization in the offline phase. Furthermore, our method can enable a light-weight CNN to perform halftoning with better image quality.

D. Discussions

Comparison of Gradient Estimators. To make the quantized image differentiable with respect to the θ , Yoo *et al.* [33] introduced the soft projection operation that relaxes the quantization with a small temperature. Later, Xia *et al.* [23] found that a simple relaxation alone was not sufficient for halftones, which has only two discrete levels. They proposed the binarization loss that greedily pushes the predictions to their nearest discrete levels. Besides, rather than stochastic policies, both [33] and [23] utilized deterministic models $\mathbf{h} = \text{Thresholding}(\text{CNN}_\theta(\mathbf{c}))$. Since the gradient of the Thresholding(\cdot) operation vanishes almost everywhere, which impedes the training, the STE [25] that assumes $\nabla_\theta \mathbf{h} \approx \nabla_\theta \text{CNN}_\theta(\mathbf{c}, \mathbf{z})$ was also tested by [23]. Nevertheless, this approximated gradient is generally not the gradient of any function [50]. On the contrary, the unbiased \hat{g}_{COMA} (Eq. (17)) and \hat{g}_{LE} (Eq. (19)) are directly derived from the target (Eq. (12)), which inherently do optimization and discretization at the same time.

To test the effectiveness of these gradient estimators, we conduct training experiments using the same CNN model and optimization objective as in our method. The MSE loss curves are plotted in Fig. 11. We find that, firstly, simply adopting the STE or relaxation neither decreases MSE nor renders blue-noise halftones. Second, while the binarization loss proposed in [23] does help cluster the outputs, this greedy rule could harm the optimization (PSNR (Näsänen/Gaussian), SSIM and CSSIM scores: 30.749/35.974/0.1529/0.9229). By contrast, our estimators \hat{g}_{LE} and \hat{g}_{COMA} result in stable and quick optimization processes. Last but not least, we find that

²Note that it is not appropriate to assess halftones directly using SSIM [48]. We list SSIM scores here for reference.

TABLE III
QUANTITATIVE COMPARISON OF HALFTONING METHODS. METRICS ARE MEASURED ON THE VOC2012 TEST SET. RUNTIMES ARE MEASURED ON THE 512x512 “LENA” TEST IMAGE. “*” INDICATES THIS METHOD IS GPU-ACCELERATED.

Method		PSNR (Näsänen)	PSNR (Gaussian)	SSIM [19]	CSSIM (Eq. (28))	Param. #	Runtime
VAC [5]	(64x64)	29.191±0.836	34.472±1.758	0.0808±0.0584	0.9075±0.0370	4K	0.03ms*
VAC [5]	(512x512)	29.283±0.839	34.490±1.764	0.0808±0.0584	0.9075±0.0370	262K	0.03ms*
OVED [8]		33.052±0.481	42.795±1.259	0.0998±0.0733	0.9117±0.0339	384	2.36ms
SAED [10]		30.942±0.831	37.786±1.683	0.1220±0.0774	0.9175±0.0302	341K	435.25ms*
TDDE _{BS} [12]		31.798±0.519	42.419±0.521	0.0887±0.0679	0.9085±0.0363	818	3.07ms
SGED [13]		31.441±0.675	38.360±1.163	0.1391±0.0849	0.9195±0.0289	-	9.87ms
DBS [15]		33.987 ±0.414	44.365 ±0.534	0.0816±0.0675	0.9055±0.0389	-	932.49ms
SAH [16]		31.214±1.095	38.018±2.160	0.1380±0.0934	0.9210±0.0275	-	162s
RVH [23]	(w/ recons.)	29.447±0.737	34.956±1.074	0.1500±0.0912	0.9220±0.0272	37.8M	49.61ms*
RVH [23]	(w/o recons.)	30.573±0.562	34.658±0.700	0.1360±0.0907	0.9172±0.0298		
cGAN [24]		30.540±0.553	36.167±0.500	0.1133±0.0770	0.9159±0.0316	3.1M	166.67ms*
Ours		31.642±0.670	37.547±0.813	0.1610 ±0.0959	0.9236 ±0.0264	299K	28.38ms*

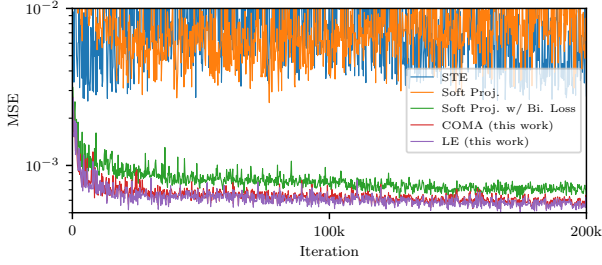


Fig. 11. MSE training loss curves of gradient estimators.

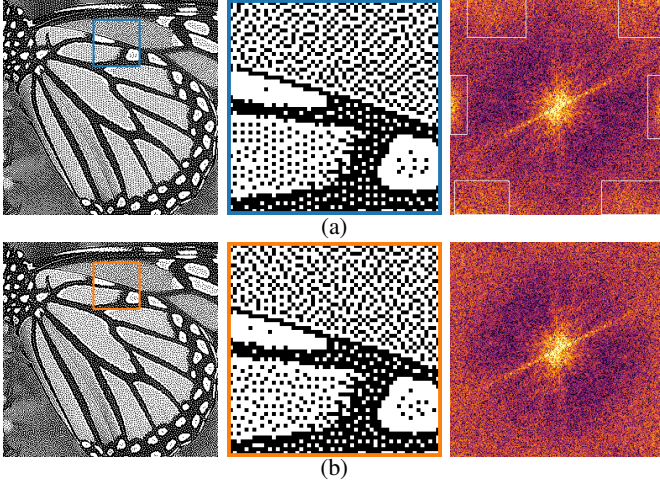


Fig. 12. Halftone samples of image “Butterfly” with Fourier amplitude spectrum: (a) w/o \mathcal{L}_{AS} ($w_a = 0$). (b) w/ \mathcal{L}_{AS} ($w_a = 0.002$).

\hat{g}_{LE} performs better than \hat{g}_{COMA} on PSNR (see Fig. 14) while their computational costs are the same.

Effect of w_s . The hyperparameter w_s determines the contribution of structural similarity in the reward function. To evaluate its effect, we plot the optimizing results of \hat{g}_{LE} and \hat{g}_{COMA} with different values (see Fig. 14). In this work, we pick $w_s = 0.06$ since the structure of its resulting halftone is clear enough without being overly sharp. According to the quality preference, one can trade CSSIM for better PSNR by decreasing w_s , and vice versa.

Effect of \mathcal{L}_{AS} on Real Image. In the halftoning litera-

Noise Type	PSNR	CSSIM
$\mathcal{N}(0, 1)$	31.642	0.9236
$\mathcal{U}(0, 1)$	31.663	0.9236
$\Gamma(1, 10)$	31.636	0.9236
$B(0.5)$	31.573	0.9237
Position Encoding [20]	31.685	0.9237
Dither Array [5]	31.644	0.9236
Screened Halftone [24]	31.616	0.9237



Fig. 13. Left: optimizing results with different input noise type. Right: the “Plane” halftone sample generated with position encoding noise [20].

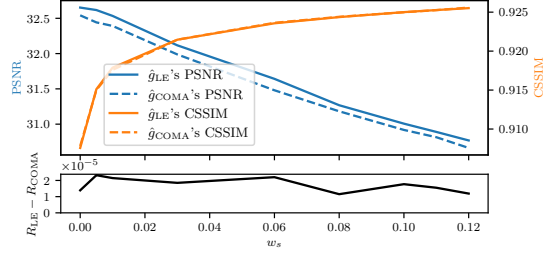


Fig. 14. Optimizing results (PSNR and CSSIM) by \hat{g}_{LE} and \hat{g}_{COMA} with different w_s , as well as their performance comparison $R_{LE} - R_{COMA}$.

ture, the blue-noise property merits particular focus [1]. For realistic images, we show two samples generated without \mathcal{L}_{AS} (Fig. 12(a)) and with \mathcal{L}_{AS} (Fig. 12(b)), as well as their Fourier amplitude spectra. In Fig. 12(a), one can easily tell the regular distribution of dots with orientation preference, which brings peaks at certain phases on the spectrum.

CNN Architecture. The proposed training framework can be applied to any CNN, so long as it keeps the image resolution. We also train a UNet [52] whose architecture strictly follows [23]’s practice. The resultant PSNR (Näsänen/Gaussian), SSIM and CSSIM scores are 31.008/36.104/0.1580/0.9233, respectively. First, we find that the UNet model is inferior to the ResNet in the halftoning task. Second, the RL-based framework successfully activates the model to achieve higher metrics than RVH, while no additional halftone dataset or auxiliary NN is needed here.

Comparison of Input Noise Type. In addition to the white Gaussian noise map $\mathcal{N}(0, 1)$ suggested by [23], we have tested more common distributions including: uniform $\mathcal{U}(0, 1)$,

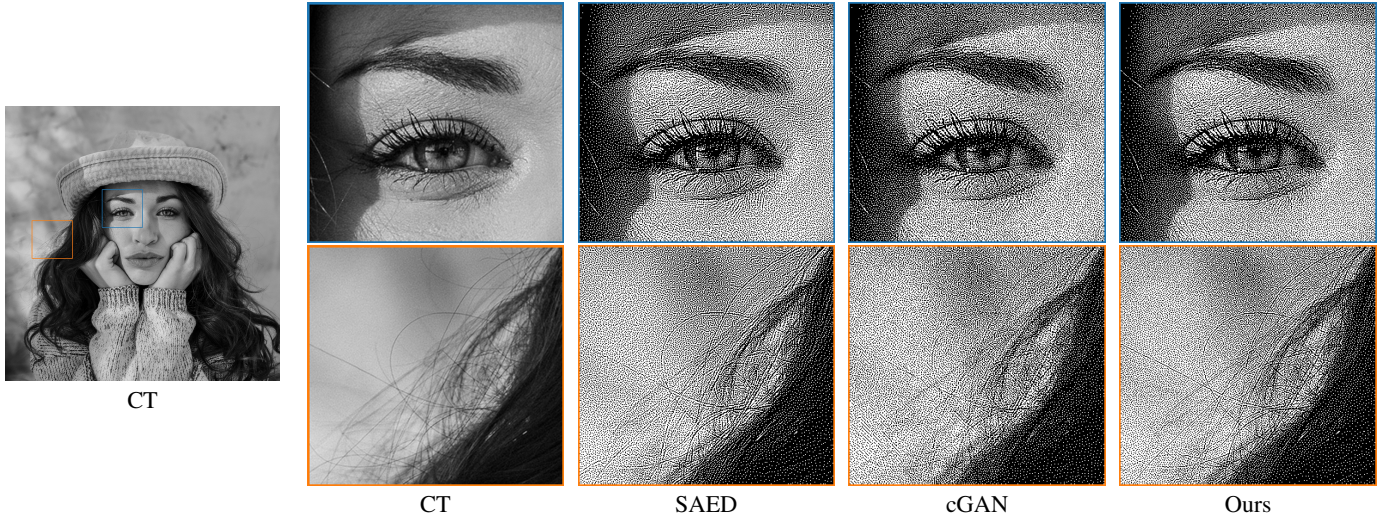


Fig. 15. Half-tone samples of image “Girl” (from DIV2K dataset [51]). SAED and our method are capable of preserving fine structures such as the hairs in this example. The resolution of the full image is 2040x2040. SAED’s runtime: 6,915 ms. cGAN’s runtime: 2,703 ms. Our runtime: 483 ms.

gamma $\Gamma(1, 10)$ and Bernoulli $B(0.5)$. We also tested three special “noise” maps with spatial correlations: the position encoding proposed by [20], the blue-noise dither array generated by [5], and the DBS-screened halftones [6] used in [24]. Experiment results (see Fig. 13) show that they achieve comparable quantitative quality. For visual quality, the position encoding “noise” used in [20] can only generate periodic textures in the flat areas. It would be an interesting topic to discuss how to render other halftone patterns [53] or other noise models [47].

Evaluation on DIV2K Dataset. In order to demonstrate the generalization ability of the proposed method, we also evaluated it on the DIV2K dataset [51], which is split into 800 images for training and 100 images for testing. The PSNR and CSSIM scores are 31.668 and 0.9334, respectively. Fig. 15 shows a sample from this dataset [51]. We select SAED [10] and cGAN [24] (also retrained on DIV2K) for comparison here as the former shows ability to produce structural details, and the latter is currently the only published deep learning approach that focuses on generating aperiodic halftones. One can see that our method can effectively capture details and demonstrate a significant increase in speed. In addition, we tested the new model trained here on the VOC2012 [43] test set, and the PSNR/CSSIM scores are 31.535/0.9235, which is comparable to the original results (31.642/0.9236).

E. An Example of Extending to Multitoning

The framework presented above is exemplified by solving the basic binary halftoning problem, but it can be extended to more complex situations, such as color halftoning [54] and multitoning [55]. To show the extensibility of our method, here we present a *prototype* of deep multitoning, which reproduces a continuous-tone image with dots of more than two discrete ink intensities. A straightforward solution is to let CNN generate multi-channel images by replacing the sigmoid layer with the softmax operation. However, it not only brings a larger action space that we need to traverse (now the complexity is $O(N(L-1)+1)$, where N is the

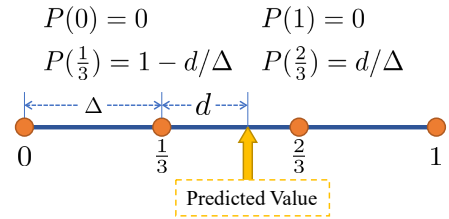


Fig. 16. Illustration of casting CNN’s outputs to the probabilities of multitone pixel’s actions (levels=4).

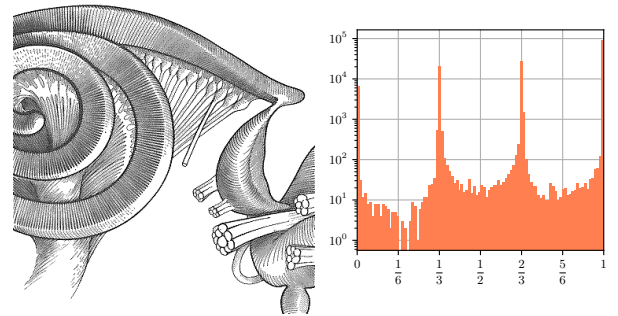


Fig. 17. Left: a multitone example generated by our deep multitoning model. Right: the histogram of the predicted values (note the logarithmic scale).

pixel number and L is the number of available ink levels), but also disables the directly applying of the proposed anisotropy estimation solution. Motivated by some schemes from gradient quantization works [56], [57], we continue using the binary output model, but define the probabilities of actions as:

$$P(h_a) = \begin{cases} (\text{ceil}(v_a) - v_a)/\Delta & h_a = \text{floor}(v_a) \\ (v_a - \text{floor}(v_a))/\Delta & h_a = \text{ceil}(v_a) \\ 0 & \text{otherwise,} \end{cases} \quad (30)$$

where $\text{ceil}(v_a)$ and $\text{floor}(v_a)$ denotes the nearest two discrete levels around v_a predicted by the CNN, and $\Delta = 1/(L-1)$ is the distance between levels. A 4-level case is illustrated in Fig. 16. The dimension of each agent’s action space is still two, regardless of L , and we can directly extend the

proposed algorithm to here with the same optimization target and training procedure. A multitone sample and the predicted values' histogram (see Fig. 17) show that the majority of output values gather around the discrete levels. As the number of available levels L increases, finally the CNN learns an identical mapping.

Note that there are extra considerations in the multitoning problem, such as the different blue noise model [55] and the banding artifact [58]. While the extended solution presented here is computationally feasible, it can be further improved by taking these aspects into account. It would not be trivial to conduct a systematical study of the multi-level situation. Consequently, we leave them for future work.

VII. CONCLUSION

An efficient halftoning method with a data-driven methodology is proposed in this paper. First, we propose to formulate halftoning as a reinforcement learning problem, in which an effective gradient estimator is tailored to train a light-weight CNN as the policy network. Second, to achieve the blue-noise property, the anisotropy of constant grayscale images' halftone is suppressed by a new loss function in the training phase. Finally, we suggest weighting the original SSIM metric by the contrast map of the input continuous-tone image to avoid the hole problem. While existing halftoning methods use heuristics or conduct expensive search strategies, our trained model not only generates halftones with structural details but also stays efficient (15x faster than prior structure-aware method), as shown in the experiments. Our framework can also be extended to address related problems such as multitoning.

We hope this work will motivate more colleagues to consider the potential benefits of deep learning for halftoning. Future work could include:

- A more domain-specific neural model to narrow the run-time gap between the present work and classic methods, such as error diffusion [13].
- Extending to related problems, including, but not limited to, multitoning [58], color halftoning [59], video halftoning [60], and other noise models [47], [61].
- Merging the deep halftoning model into real-time printer image processing pipelines.

ACKNOWLEDGMENTS

The authors would like to thank Aiguo Yin from Pantum Electronics and Dr. Menghan Xia from Tencent AI Lab for their support in this work.

REFERENCES

- [1] D. L. Lau and G. R. Arce, *Modern Digital Halftoning*. CRC Press, 2008.
- [2] R. A. Ulichney, "Dithering with blue noise," *Proc. IEEE*, vol. 76, no. 1, pp. 56–79, 1988.
- [3] B. E. Bayer, "An optimum method for two-level rendition of continuous tone pictures," in *Proc. IEEE Int. Communication Conf.*, 1973, pp. 2611–2615.
- [4] J. Sullivan, L. Ray, and R. Miller, "Design of minimum visual modulation halftone patterns," *IEEE Trans. Man-Mach. Syst.*, vol. 21, no. 1, pp. 33–38, 1991.
- [5] R. A. Ulichney, "Void-and-cluster method for dither array generation," in *Proc. SPIE*, 1993, pp. 332–343.
- [6] J. P. Allebach and Q. Lin, "Fm screen design using dbs algorithm," in *Proc. IEEE Int. Conf. Image Process.*, vol. 1, 1996, pp. 549–552.
- [7] R. W. Floyd and L. Steinberg, "An adaptive algorithm for spatial grayscale," in *Proc. Soc. Inf. Disp.*, vol. 17, 1976, pp. 75–77.
- [8] V. Ostromoukhov, "A simple and efficient error-diffusion algorithm," in *Proc. SIGGRAPH*, 2001, pp. 567–572.
- [9] P. Li and J. P. Allebach, "Tone-dependent error diffusion," *IEEE Trans. Image Process.*, vol. 13, no. 2, pp. 201–215, 2004.
- [10] J. Chang, B. Alain, and V. Ostromoukhov, "Structure-aware error diffusion," *ACM Trans. Graph.*, vol. 28, no. 5, pp. 1–8, 2009.
- [11] H. Li and D. Mould, "Contrast-aware halftoning," *Comput. Graph. Forum*, vol. 29, no. 2, pp. 273–280, 2010.
- [12] Y.-H. Fung and Y.-H. Chan, "Tone-dependent error diffusion based on an updated blue-noise model," *J. Electron. Imag.*, vol. 25, no. 1, p. 013013, 2016.
- [13] X. Y. Hu, "Simple gradient-based error-diffusion method," *J. Electron. Imag.*, vol. 25, no. 4, p. 043029, 2016.
- [14] Y. Mao, L. Abello, U. Sarkar, R. A. Ulichney, and J. P. Allebach, "4-row serpentine tone dependent fast error diffusion," in *Proc. IEEE Int. Conf. Image Process.*, 2018, pp. 3973–3977.
- [15] M. Analoui and J. P. Allebach, "Model-based halftoning using direct binary search," in *Proc. SPIE*, vol. 1666, 1992, pp. 96–108.
- [16] W.-M. Pang, Y. Qu, T.-T. Wong, D. Cohen-Or, and P.-A. Heng, "Structure-aware halftoning," *ACM Trans. Graph.*, vol. 27, no. 3, pp. 1–8, 2008.
- [17] A. Chatterjee, B. Tudu, and K. C. Paul, "Towards optimized binary pattern generation for grayscale digital halftoning: A binary particle swarm optimization (bpso) approach," *J. Vis. Commun. Image Represent.*, vol. 23, no. 8, pp. 1245–1259, 2012.
- [18] R. Näsänen, "Visibility of halftone dot textures," *IEEE Trans. Man-Mach. Syst.*, vol. SMC-14, no. 6, pp. 920–924, 1984.
- [19] Z. Wang, A. Bovik, H. Sheikh, and E. Simoncelli, "Image quality assessment: from error visibility to structural similarity," *IEEE Trans. Image Process.*, vol. 13, no. 4, pp. 600–612, 2004.
- [20] T.-H. Kim and S. I. Park, "Deep context-aware descreening and rescreening of halftone images," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–12, 2018.
- [21] M. Xia and T.-T. Wong, "Deep inverse halftoning via progressively residual learning," in *Proc. Asian Conf. Comput. Vis.*, 2019, pp. 523–539.
- [22] J.-M. Guo and S. Sankarasrinivasan, "H-gan: Deep learning model for halftoning and its reconstruction," in *Proc. IEEE Int. Conf. Consum. Electron.*, 2020, pp. 1–2.
- [23] M. Xia, W. Hu, X. Liu, and T.-T. Wong, "Deep halftoning with reversible binary pattern," in *Proc. IEEE/CVF Int. Conf. Comput. Vis.*, 2021, pp. 13 980–13 989.
- [24] B. Choi and J. P. Allebach, "Mimicking dbs halftoning via a deep learning approach," in *Proc. Color Imag. XXVII, Displaying, Process., Hardcopy, Appl., Part IS&T Electron. Imag.*, 2022, pp. 158–1–158–7.
- [25] Y. Bengio, N. Léonard, and A. Courville, "Estimating or propagating gradients through stochastic neurons for conditional computation," 2013. [Online]. Available: <https://arxiv.org/abs/1308.3432>
- [26] H. Jiang, D. Xiong, X. Jiang, A. Yin, L. Ding, and K. Huang, "Halftoning with multi-agent deep reinforcement learning," in *Proc. IEEE Int. Conf. Image Process.*, 2022, pp. 641–645.
- [27] R. Furuta, N. Inoue, and T. Yamasaki, "Pixelrl: Fully convolutional network with reinforcement learning for image processing," *IEEE Trans. Multimedia*, vol. 22, no. 7, pp. 1704–1719, 2020.
- [28] J. N. Foerster, G. Farquhar, T. Afouras, N. Nardelli, and S. Whiteson, "Counterfactual multi-agent policy gradients," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, 2018, pp. 2974–2982.
- [29] M. Titsias RC AUEB and M. Lázaro-Gredilla, "Local expectation gradients for black box variational inference," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 28, 2015, pp. 2638–2646.
- [30] R. Dahl, M. Norouzi, and J. Shlens, "Pixel recursive super resolution," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017, pp. 5449–5458.
- [31] S. Menon, A. Damian, S. Hu, N. Ravi, and C. Rudin, "Pulse: Self-supervised photo upsampling via latent space exploration of generative models," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020, pp. 2434–2442.
- [32] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley, "Least squares generative adversarial networks," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2017.
- [33] I. Yoo, X. Luo, Y. Wang, F. Yang, and P. Milanfar, "Gifnets: Differentiable gif encoding framework," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2020.

- [34] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT press, 2018.
- [35] Y. Hu, H. He, C. Xu, B. Wang, and S. Lin, "Exposure: A white-box photo post-processing framework," *ACM Trans. Graph.*, vol. 37, no. 2, may 2018.
- [36] C. Wang, R. Zhang, S. Ravishankar, and B. Wen, "Repnp: Plug-and-play with deep reinforcement learning prior for robust image restoration," in *Proc. IEEE Int. Conf. Image Process.*, 2022, pp. 2886–2890.
- [37] D. Li, H. Wu, J. Zhang, and K. Huang, "A2-rl: Aesthetics aware reinforcement learning for image cropping," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, 2018, pp. 8193–8201.
- [38] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, no. 3, pp. 229–256, 1992.
- [39] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu, "Asynchronous methods for deep reinforcement learning," in *Proc. Int. Conf. Mach. Learn.*, vol. 48, 2016, pp. 1928–1937.
- [40] S. Gronauer and K. Diepold, "Multi-agent deep reinforcement learning: a survey," *Artif. Intell. Rev.*, vol. 55, no. 2, pp. 895–943, 2022.
- [41] S. Mohamed, M. Rosca, M. Figurnov, and A. Mnih, "Monte carlo gradient estimation in machine learning," *J. Mach. Learn. Res.*, vol. 21, no. 132, pp. 1–62, 2020.
- [42] S. H. Kim and J. P. Allebach, "Impact of hvs models on model-based halftoning," *IEEE Trans. Image Process.*, vol. 11, no. 3, pp. 258–269, 2002.
- [43] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," <http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html>.
- [44] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 770–778.
- [45] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *Proc. Int. Conf. Learn. Represent.*, 2015.
- [46] D. Lau and R. Ulichney, "Blue-noise halftoning for hexagonal grids," *IEEE Trans. Image Process.*, vol. 15, no. 5, pp. 1270–1284, 2006.
- [47] Y.-H. Fung and Y.-H. Chan, "Tone-dependent noise model for high-quality halftones," *J. Electron. Imag.*, vol. 22, no. 2, p. 023004, 2013.
- [48] P. Itoua, A. Beghdadi, and P. Viaris de lesegno, "Objective perceptual evaluation of halftoning using image quality metrics," in *Proc. Int. Conf. Inf. Sci. Signal Process. Appl.*, 2010, pp. 456–459.
- [49] J.-R. Liao, "Theoretical bounds of direct binary search halftoning," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 3478–3487, 2015.
- [50] P. Yin, J. Lyu, S. Zhang, S. J. Osher, Y. Qi, and J. Xin, "Understanding straight-through estimator in training activation quantized neural nets," in *Proc. Int. Conf. Learn. Represent.*, 2019.
- [51] E. Agustsson and R. Timofte, "Ntire 2017 challenge on single image super-resolution: Dataset and study," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2017, pp. 1122–1131.
- [52] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *MICCAI*, 2015, pp. 234–241.
- [53] T. Frank, J. Liu, S. Gat, O. Haik, O. B. Mor, I. Roth, J. P. Allebach, and Y. Yitzhaky, "A machine learning approach to design of aperiodic, clustered-dot halftone screens via direct binary search," *IEEE Trans. Image Process.*, vol. 31, pp. 5498–5512, 2022.
- [54] F. Baqai, J.-H. Lee, A. Agar, and J. P. Allebach, "Digital color halftoning," *IEEE Signal Process. Mag.*, vol. 22, no. 1, pp. 87–96, 2005.
- [55] J. Bacca Rodriguez, G. R. Arce, and D. L. Lau, "Blue-noise multitone dithering," *IEEE Trans. Image Process.*, vol. 17, no. 8, pp. 1368–1382, 2008.
- [56] S. Zhou, Y. Wu, Z. Ni, X. Zhou, H. Wen, and Y. Zou, "Dorefa-net: Training low bitwidth convolutional neural networks with low bitwidth gradients," 2016. [Online]. Available: <https://arxiv.org/abs/1606.06160>
- [57] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, "Qsgd: Communication-efficient sgd via gradient quantization and encoding," in *Proc. Adv. Neural Inf. Process. Syst.*, 2017, pp. 1709–1720.
- [58] J.-M. Guo, J.-Y. Chang, Y.-F. Liu, G.-H. Lai, and J.-D. Lee, "Tone-replacement error diffusion for multitone," *IEEE Trans. Image Process.*, vol. 24, no. 11, pp. 4312–4321, 2015.
- [59] A. U. Agar and J. P. Allebach, "Model-based color halftoning using direct binary search," *IEEE Trans. Image Process.*, vol. 14, no. 12, pp. 1945–1959, 2005.
- [60] W.-C. Kao and J.-L. Ho, "Fast video halftoning for electronic papers," in *Proc. IEEE Int. Conf. Consum. Electron. Taiwan*, 2022, pp. 595–596.
- [61] P. Goyal, M. Gupta, C. Staelin, M. Fischer, O. Shacham, and J. P. Allebach, "Clustered-dot halftoning with direct binary search," *IEEE Trans. Image Process.*, vol. 22, no. 2, pp. 473–487, 2013.