DOI: 10.1111/cgf.70173
Eurographics Symposium on Rendering 2025
B. Wang and A. Wilkie
(Guest Editors)

COMPUTER GRAPHICS *forum*
*Volume 44 (2025), Number 4*

# Differentiable Search Based Halftoning

E. Luci[1], K. T. Wijaya[1], V.Babaei[1]

[1]Max Planck Institute for Informatics, Germany

**Figure 1:** *A halftone image showing the capabilities of our method.* ***Top Right:*** *multilevel RGB color halftoning.* ***Top Left:*** *halftoning with custom color palette.* ***Bottom Left:*** *black and white halftoning.* ***Bottom Right:*** *RGB color halftoning.*

**Abstract**

*Halftoning is fundamental to image reproduction on devices with a limited set of output levels, such as printers. Halftoning algorithms reproduce continuous-tone images by distributing dots with a fixed tone but variable size or spacing. Search-based approaches optimize for a dot distribution that minimizes a given visual loss function w.r.t. an input image. This class of methods is not only the most intuitive and versatile but can also yield the highest quality results depending on the merit of the employed loss function. However, their combinatorial nature makes them computationally inefficient. We introduce the first differentiable search-based halftoning algorithm. Our proposed method can be natively used to perform multi-color, multi-level halftoning. Our main insight lies in introducing a relaxation in the discrete choice of dot assignment during the backward pass of the optimization. We achieve this by associating a fictitious distance from the image plane to each dot, embedding the problem in three dimensions. We also introduce a novel loss component that operates in the frequency domain and provides a better visual loss when combined with existing image similarity metrics. We validate our approach by demonstrating that it outperforms stochastic optimization methods in both speed and objective value, while also scaling significantly better to large images. The code is available at* https:gitlab.mpi-klsb.mpg.de/aidam-public/differentiable-halftoning

**CCS Concepts**

• ***Computing methodologies*** → *Image processing;*

## 1. Introduction

Almost every printed image we encounter in our daily lives is a halftone image. Unlike continuous-tone images, on our displays for example, which contain a wide range of intensities, printers can only reproduce a very limited intensity range at each pixel. In fact, most printers can either mark a pixel with ink or leave it blank. To overcome this hardware limitation and create the visual impression of a continuous-tone images, halftoning algorithms leverage the low-pass filtering property of the human visual system (HVS). They generate a dot pattern that, when printed at high enough res-

olution, blends into a smooth appearance. Thus, the fundamental challenge of halftoning is to find an artifact-free spatial arrangement of dots that best approximates the original continuous-tone image.

There exists a wide range of halftoning methods, from the well-established *smart thresholding* techniques [Bay76] to more recent approaches based on deep learning [KP18]. Among these, the most intuitive, transparent, and controllable are search-based methods, which directly optimize the spatial distribution of dots to best approximate the input image. While these methods achieve the highest quality, they are computationally expensive and difficult to scale. This is because searching for the optimal dot distribution is a combinatorial, NP-hard problem. Existing search-based methods rely on stochastic optimization, limiting their scalability as image resolution increases. Besides, search based methods (as well as deep learning methods) require a halftoning visual loss which is an open problem.

We introduce the first differentiable search-based halftoning method that is fast, scalable, and flexible. Our key insight is that the inherently 2D halftoning problem can be embedded in a 3D space, where each dot is associated with a fictitious vertical coordinate. This additional degree of freedom allows us to formulate a continuous relaxation of the problem, replacing stochastic heuristics with differentiable transformations. This relaxation departs from the randomization approaches used in previous search-based methods [PQW*08] and enables us to model the discrete appearance and disappearance of dots in a differentiable manner. Moreover, our Lagrangian formulation allows efficient multi-color, multi-level halftoning. This is a departure from the traditional color halftoning extensions where each ink layer is independently halftoned and superimposed. Such a superposition makes the resulting color halftone image susceptible to artifacts, such as moiré [Ami09]. Furthermore, our solution treats each dot in parallel, achieving significant speedups. Our method reliably produces high-quality halftoned images with slight dependency on initialization. This is unlike previous search-based methods that depend heavily on starting conditions. Finally, our flexible framework supports any differentiable image quality metric, allowing for tailored optimization objectives which can be tested almost at an interactive rate. We leverage this flexibility by introducing a novel halftoning loss that combines image statistics and perceptual metrics to improve output quality.

To summarize our main contributions:

1. We present the first differentiable search-based halftoning algorithm with a continuous formulation. Our method speeds up the optimization process even for very large images, while maintaining a reasonable memory footprint and surpasses the counterpart stochastic optimizations in terms objective value.
2. Our method works seamlessly and efficiently in multi-level, multi-color halftoning scenarios.
3. We show how different image losses perform with our reliable optimization, propose a new loss component, and almost interactively tune different loss combinations.

## 2. Related work

**Halftoning with Smart Thresholding.** Halftoning is a very established problem with established solutions that with today computing standards are extremely efficient. A main class of these solutions are based on smart thresholding. In one category of smart thresholding a hand-crafted threshold matrix is designed and is used to tile the image. Ordered dither [Bay76] and clustered-dot [Kan99] halftoning are the most well-known of these methods. In error diffusion halftoning [Flo76], each pixel is thresholded individually and the induced error is propagated to the neighboring pixels. This method has been improved upon by Ostromokhouv [Ost01] and Chang *et al*. [CAO09]. Despite its extreme efficiency, halftoning with smart thresholding is known to introduce obvious periodic or other low-frequency artifacts. Having said that, these methods stay very industrially relevant mainly because of their scalability.

**Search Based Algorithms.** Thresholding methods are inherently limited by the hand-crafted design choices for the spatial distribution of threshold values or error propagation. The more intuitive, controllable class of halftoning is search-based methods. These methods treat halftoning as an optimization problem [AA92, PQW*08, CTP12, WWH13, AGKN21]. This means, they require a halftone quality metric, typically inspired by the properties of the human visual system (HVS) model [Näs84]. More importantly, to deal with the combinatorial nature of halftoning, they rely on stochastic algorithms, such as greedy search [AA92] or simulated annealing [PQW*08]. Search-based methods are known to deliver the highest halftone quality. They are also flexible and transparent methods as they directly optimize for a desired quality metric. Unfortunately, these methods constitute the most computationally intensive class of halftoning especially when dealing with high-resolution images. Furthermore, they are sensitive to initialization, cannot escape local minima, and once initialized with a certain number of dots can only change the dot patterns and *not* change the number of dots. As we shall see, our method, which belongs to this class, addresses all these challenges.

**Neural Halftoning.** Unsurprisingly, learning methods have recently affected the area of halftoning [CA22, XHLW21a, GS20, KP18, JM22]. The focus has however been on inverse halftoning where given a halftone image we recover the original, continuous-tone image. The lack of more 'forward' deep halftoning method can be attributed to the lack of a large and, simultaneously, high-quality dataset. A successful deep halftoning method is the work of Xia *et al*. [XHLW21b] where in addtion to typical image similarity metrics they consider invertibility. This means that additional information is encoded in the halftoning pattern to be able to recover the original image. Recently, and in the same vein as in our method, Jiang *et al*. [JZM24] introduce a differentiable encoding of dots by using neural networks. Interestingly, they devise a separate neural network to compute the number of dots given the input image. Our method, does not need such a careful initialization as it can differentiably appear or disappear dots. In general, search-based halftoning methods are significantly more transparent and flexible than methods based on deep learning.

**Physics Inspired Halftoning** Given the importance of halftoning, there are a few interesting physics inspired methods during the

past two decades including methods for generating blue noise samples [Gra06, SGBW10a, WW17, SGW12, JZW*15]. A particularly interesting work is the electrostatic halftoning [SGBW10b] that considers halftoning as an energy minimization problem between charged particles. Unfortunately, these methods resort to stochastic algorithms in case an optimization is involved and are therefore not scalable to higher resolutions.

## 3. Overview

An halftoning function $H : \mathbf{I_c} \mapsto \mathbf{I_d}$ transforms a continuous-tone image $\mathbf{I_c} = [0,1]^{X \times Y \times N}$ of $N$ channels, into a halftone image $\mathbf{I_d} = \{v_0, \dots, v_M\}^{X \times Y \times D}$ such that a pixel can have $M$ different intensities for each of the $D$ different ink types. The *ideal* halftoning can be cast as an integer optimization problem where the objective function to minimize is a *visual loss* $\mathcal{L}$, and the variables are the discrete intensity levels of the individual pixels for each ink:

$$\min_{\boldsymbol{I_d}} \mathcal{L}(\boldsymbol{I_c}, \boldsymbol{I_d}). \tag{1}$$

One can immediately identify two important challenges. First, the above optimization problem falls under the category of integer programming; that quickly make the problem becomes intractable as images can be composed of millions of pixels, that is, millions of variables. Second, constructing an appropriate visual halftoning loss is non-trivial. Such a loss function should specify how close a continuous image is to its discrete approximation. Quantifying how humans perceive similarity between images, even in far less challenging scenarios, is an active topic of research [YLQS23, MSL*25, MEMS14]. A satisfactory loss can be constructed by combining multiple conventional losses, such as MSE and SSIM and assigning different weights to them [PQW*08]. However, finding the appropriate weights is a tedious and not well-defined task, as the choice of weights is not only subjective but also image-dependent.

In this work, we mainly focus on the first challenge of loss minimization. We propose a novel relaxation and its associated projection that allows us to use gradient descent techniques to solve such an optimization. The idea is to treat each *dot* assignment as a *potential* value that represents a dot's tendency to be selected. We model this using a 3D embedding, where each dot from the 2D image is assigned a fictitious distance from the image plane (Figure 2). This function does not need to be Euclidean, allowing flexibility in how dot selection is guided. We then compute the *gradient* of the color of a pixel w.r.t. a dot position. During the projection step we select the dots based on their distance, producing a discrete assignment of dots. We then compute the loss and proceed to update the dots positions given their gradients. This formulation immediately enables many benefits. By utilizing multiple dot *types* per pixel we obtain color halftoning, and we can perform multi-level halftoning, where there are more than two (inked or blank) levels, by having fractional values for a dot assignment.

Related to the second challenge mentioned above, our formulation allows us to solve the weight assignment problem when combining *base* losses. We can explore the landscape of different weights by having the optimizer run continuously as we change the loss function. This allows a human observer to see in real-time how the halftoning result changes while changing the weights and select their preferred ones.

**Table 1:** *List of Symbols*

| Symbol | Description |
|---|---|
| $I_c$ | Continuous Image |
| $I_d$ | Discretized Image |
| $D$ | Number of different dots (inks) |
| $\tau$ | Thickness of the slicing volume |
| $z^d_{x,y}$ | Distance of dot $d$ at position $x,y$ from the limit |
| $\mathbf{t}^d_{x,y}$ | Type of dot $d$ at position $x,y$ |
| $\mathbf{t}_{x,y}$ | Type of pixel at position $x,y$ |
| $\mathbf{s}_{x,y}$ | Assignment (choice of dots that color the pixel) for pixel $x,y$ |
| $\mathbf{c}_{x,y}$ | Color of pixel at position $x,y$ |

## 4. Method

We describe here the details of our method. The description is for a general halftoning, that is, including both color and multilevel halftoning. We use the bold notation to represent tensors. We first introduce the details of our formulation and how the optimization proceeds and then present our novel loss component as well as our loss weights optimization tool. A table of symbols is provided for reference in Table 1.

### 4.1. Moving Dots

In our conceptualization of the problem, the color of a pixel within the halftone image is influenced by the *dots* that are near the image plane. For a clearer mental image, dots can be imagined as the inks of a printer. We can *place* dots on the substrate and that spot will become colored. Similarly, given an image $\mathbf{I_c}$, to each pixel we associate a given number $D > 0$ of dots. $D$ represents the different number of inks at our disposal, e.g., 4 for a CMYK printer. We index each dot by its $x,y$ coordinates within the image, and by an index $1 \leq d \leq D$ representing the ink number. Each dot will have an initial distance $z^d_{x,y}$ from the image plane, and an ink type $\mathbf{t}^d_{x,y} \in \mathbb{R}^K, K \in \mathbb{N}$. The type is a vector that represents the position in the ink space of that particular ink. For maximal generality, an ink space is used to describe inks. For example, a 2D ink space can describe how much of two different pigments are present in a given ink formulation.

### 4.2. Image Formation

A halftone image is formed by *selecting* dots for each pixel, *mixing* their types together and *rendering* the final color of the pixel.

The selection of dots is based on their distance and constitutes the projection step. We call an assignment $\mathbf{s}_{x,y} \in \{v_0, \dots, v_m\}^D$ a potentially fractional selection of up to $D$ dots for the pixel $x,y$. A selection describes which dots are currently contributing to the color of a pixel and can be thought of as which dots are placed on the substrate and in what amount. A general selection strategy, for

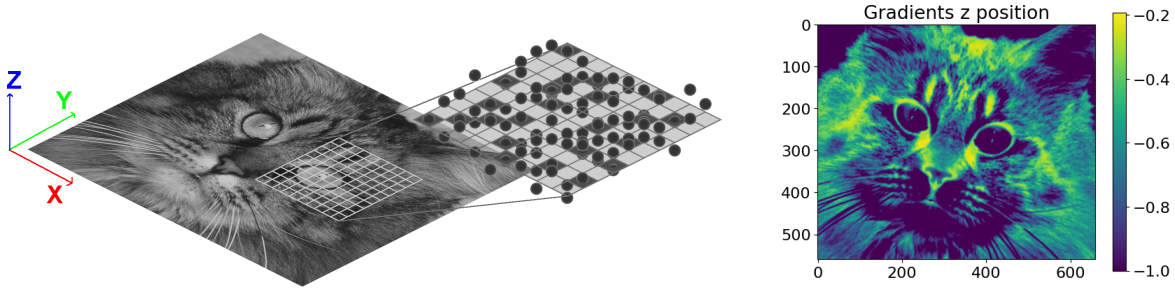



**Figure 2:** *A representation of how our method works. Dots move along the 3rd dimension and add a type to a pixel when they intersect with the plane. This then gets rendered as a pixel color. On the right we show the gradients of the dots position (along the z axis) at the beginning of the optimization process, when all the dots are outside of the volume.*

each particular dot, produces an assignment:

$$s^d_{x,y} = p\left(z^d_{x,y}\right), \tag{2}$$

with $p$ being some step function with positive values in the interval $[0,\tau]$ and 0 everywhere else, where $\tau$ is some positive value that defines the height of the slicing volume. The value of $\tau$ also represents the distance at which a dot is "most selected". This is perhaps counterintuitive but it makes for a simple formulation of the problem. The simplest selection strategy $p$ for a typical bi-level halftoning is a a thresholding function. For each pixel we build its assignment $\mathbf{s}_{x,y}$ as:

$$s^d_{x,y} = \begin{cases} 1 & \text{if } 0 \le z^d_{x,y} \le \tau \\ 0 & \text{otherwise} \end{cases}. \tag{3}$$

This produces assignment such as $\mathbf{s}_{4,2} = [0,1,0]$, that is, we only select the second dot. This can be modified for multi-level halftoning by having for example:

$$s^d_{x,y} = \begin{cases} 1 & \text{if } \frac{n-1}{n}\tau \le z^d_{x,y} \le \tau \\ \dots & \\ \frac{1}{n} & \text{if } 0 \le z^d_{x,y} \le \frac{1}{n}\tau \\ 0 & \text{otherwise} \end{cases}. \tag{4}$$

An example assignment is $\mathbf{s}_{2,9} = [\frac{2}{4}, \frac{3}{4}, \frac{0}{4}, \frac{4}{4}]$, that is, we partially selected the first two dots, excluded the third and fully selected the fourth. These partial selections have practical implications as many printers can depose different amount of inks. In the above example, a printer dot is formed by a maximum of 4 'droplets'.

In Figure 3 we can see two example assignments for a bi-level (conventional) and multi-level halftoning. An assignment always guarantees halftone images regardless of the selection strategy as each dot will fall in one of the cases, and the cases represent all the possible amounts of inks to be disposed.

From a given assignment we compute the final type of a pixel as:

$$\mathbf{t}_{x,y} = \text{mixing}\left(\left\{\mathbf{t}^d_{x,y} \mid s^d_{x,y} > 0,\ d = 1,\dots,D\right\}\right), \tag{5}$$

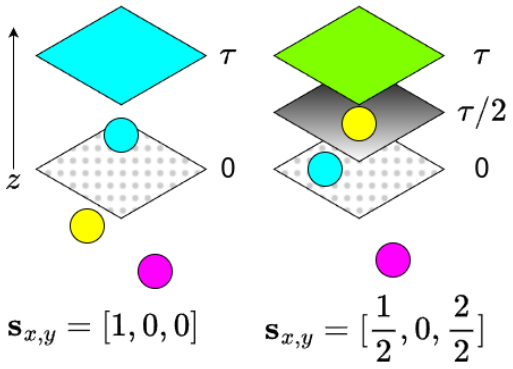



**Figure 3:** *Selection process for bi-level and tri-level halftoning showing dots moving across the z axis. In the first case only a cyan dot is selected, making the pixel appear cyan. In the second case the cyan dot is partially selected while the yellow one is fully selected, 'rendering' the pixel yellowish green.*

where the *mixing* function describes how different inks will mix together, that is, what is the overall type of the pixel when multiple of its dots are intersecting the plane at the same time? The simplest mixing function is just to add the types together. In the case where all the vector types are orthogonal to each other this is also always correct. Once we compute the type of a pixel its color is:

$$\mathbf{c}_{x,y} = \mathcal{R}(\mathbf{t}_{x,y}), \tag{6}$$

where $\mathcal{R}$ is the rendering function. This function takes as input a vector in ink space and produces a color in some color space. After this step we have formed the halftoned image $\mathbf{I_d}$ which is compared to $\mathbf{I_c}$ using $\mathcal{L}$.

A note must be made on the distinction between the ink space and the color space in the context of digital halftoning. We are not dealing with physical inks and therefore there can be an isomorphism between the two spaces making the separation artificial.

However, keeping these two spaces separate makes our framework more general, for example, if we apply our pipeline to problems where the properties used to describe inks are separate from their color.

### 4.3. Continuous Relaxation

The derivative of Eq. 6 w.r.t. a dot position is:

$$\frac{\delta \mathbf{c}_{x,y}}{\delta z^d_{x,y}} = \frac{\delta \mathbf{c}_{x,y}}{\delta \mathbf{t}_{x,y}} \frac{\delta \mathbf{t}_{x,y}}{\delta \mathbf{t}^d_{x,y}} \frac{\delta \mathbf{t}^d_{x,y}}{\delta z^d_{x,y}}. \tag{7}$$

Because of the discrete selection process as defined in Eq. 3 and Eq. 4, this derivative chain is broken. We introduce a relaxation to fix this non-continuity. Instead of a discrete selection we redefine the selection vector $\hat{\mathbf{s}}_{x,y}$ as:

$$\hat{s}^d_{x,y} = \exp(-(\tau - z^d_{x,y})^2), \tag{8}$$

changing the mixing equation as defined in Eq. 5 to:

$$\hat{\mathbf{t}}_{x,y} = \text{mixing}(\left\{ s^d_{x,y} \mathbf{t}^d_{x,y} \mid d = 1, \ldots, D \right\}), \tag{9}$$

therefore instead of a full selection for each type we have a continuous weighing of each dot type. This formulation is differentiable, and allows us to move the dots by using the gradients from the loss. Additionally, since the relaxation only affects the backward pass and not the forward, we are sure that each intermediate solution is also a solution to the original problem.

### 4.4. Initialization

To initialize our optimization, each $z^d_{x,y}$ must be set to some value. Different initializations yield distinct optimization trajectories, although, unlike other search based halftoning [PQW*08], almost all initializations are able to find a local minimum. The only requirement is that, due to the characteristics of the loss functions, our formulation requires a certain level of noise for proper convergence. In regions with uniform intensity, losses computed over a local window result in identical gradients for all dots, causing them to move in an identical manner and potentially destabilizing the optimization. To mitigate this issue, we inject noise into the initialization. Our current strategy perturbs the input image $\mathbf{I_c}$ with additive noise, and the resulting pixel values are assigned to the corresponding dots initial positions. Formally, the initialization is given by:

$$z^d_{x,y} = -\mathbf{I_c}(x,y) - \epsilon_{x,y}, \tag{10}$$

where $\epsilon_{x,y}$ is a noise term sampled from a suitable distribution such as a blue noise one. The optimizer can also start from an already halftone image, such as the output from an error diffusion algorithm. To do so we need to alter Eq. 10, such that we have:

$$z^d_{x,y} = \begin{cases} \epsilon_{x,y} & \text{if } \mathbf{I_h}(x,y) > 0 \\ -\epsilon_{x,y} & \text{otherwise} \end{cases}, \tag{11}$$

with $\mathbf{I_h}$ being the halftone image. A dot is selected if the corresponding pixel is also marked in the original image. This makes it so that the initial image is exactly the halftone image given as input. We still make sure that the dots do not have locally uniform values of $z$ to avoid the aforementioned issues.

### 4.5. Optimization Procedure

The optimization proceeds via standard gradient descent, where the dot positions are updated iteratively using the computed gradients. Additionally, a cosine annealing schedule is employed to adjust the learning rate throughout the optimization process. This schedule gradually reduces the learning rate following a cosine function, allowing for larger updates during the initial stages and finer adjustments as the optimization proceeds. The learning rate at iteration $i$ is computed as:

$$\eta(i) = \eta_{\min} + (\eta_0 - \eta_{\min}) \frac{1 + \cos\left(\frac{i}{N}\pi\right)}{2}, \tag{12}$$

where $\eta_0$ is the initial learning rate, $\eta_{\min}$ is the minimum learning rate, and $N$ is the total number of iterations. We refer to the original paper of [LH17] for further explanations.

### 4.6. Loss

We propose a novel loss function in order to preserve the tone, the structure, and the overall perception of the continuous-tone image. Our loss is inspired by the works of [PQW*08] and [XHLW21a], however, we introduce some key additions based on our own observations. Our proposed loss is defined as:

$$\mathcal{L}^i = \mathbf{w} \cdot [\text{TONE}, \text{STRUCT}, \text{FREQ}^i]. \tag{13}$$

The TONE component is then defined as:

$$\text{TONE}(\mathbf{I_d}, \mathbf{I_c}) = \text{MSE}(G(\mathbf{I_d}), \mathbf{I_c}), \tag{14}$$

where MSE is the mean squared error:

$$\text{MSE}(\mathbf{I}, \hat{\mathbf{I}}) = \frac{1}{WH} \sum_{x=1}^{W} \sum_{y=1}^{H} \left(\mathbf{I}_{x,y} - \hat{\mathbf{I}}_{x,y}\right)^2. \tag{15}$$

Similarly the STRUCT component is then defined as:

$$\text{STRUCT}(\mathbf{I_d}, \mathbf{I_c}) = \text{SSIM}(G(\mathbf{I_d}), \mathbf{I_c}). \tag{16}$$

with SSIM being the Structural Similarity:

$$\text{SSIM}(\mathbf{I}, \hat{\mathbf{I}}) = \frac{(2\mu_{\mathbf{I}}\mu_{\hat{\mathbf{I}}} + C_1)(2\sigma_{\mathbf{I}\hat{\mathbf{I}}} + C_2)}{(\mu^2_{\mathbf{I}} + \mu^2_{\hat{\mathbf{I}}} + C_1)(\sigma^2_{\mathbf{I}} + \sigma^2_{\hat{\mathbf{I}}} + C_2)}, \tag{17}$$

where:

- $\mu_{\mathbf{I}}$, $\mu_{\hat{\mathbf{I}}}$ are the means of the image patches $\mathbf{I}$ and $\hat{\mathbf{I}}$,
- $\sigma^2_{\mathbf{I}}$, $\sigma^2_{\hat{\mathbf{I}}}$ are the variances,
- $\sigma_{\mathbf{I}\hat{\mathbf{I}}}$ is the covariance between $\mathbf{I}$ and $\hat{\mathbf{I}}$,
- $C_1 = (K_1 L)^2$, $C_2 = (K_2 L)^2$ are small constants for numerical stability, with $L$ the dynamic range of pixel values and typical values $K_1 = 0.01$, $K_2 = 0.03$.

For a full explanation we refer to the original paper [WBSS04].

Finally, the $\text{FREQ}^{\text{i}}$ component is a novel loss that we introduce here, based on what we call the Hierarchical Discrete Cosine Transform Loss,

$$\text{FREQ}^i(\mathbf{I_d}, \mathbf{I_c}, i, i_{\max}) = \text{H-DCT}(G(\mathbf{I_d}), \mathbf{I_c}, i, i_{\max}), \tag{18}$$

where $i$ and $i_{\max}$ are two parameters that describe the current iteration of the loss and $i_{\max}$ is the maximum number of iterations the loss is going to be computed over. The loss dynamically compares

at each iteration of the optimization the Discrete Cosine Transform (DCT) of the original image with the DCT of the blurred version of the halftone image. We report the algorithm for the H-DCT here:

**Algorithm 1** H-DCT

**Require:** $\mathbf{I_d}$, *current_iteration*, *max_iterations*

$dct\_image \leftarrow \text{DCT\_2D}(G(\mathbf{I_d}))$
$percentage\_cutoff \leftarrow \frac{current\_iteration+1}{max\_iterations}$
$width \leftarrow image.width$
$height \leftarrow image.height$
$h\_cutoff \leftarrow \lfloor (1-\exp(-\phi \cdot percent\_cutoff)) \cdot width \rfloor$
$v\_cutoff \leftarrow \lfloor (1-\exp(-\phi \cdot percent\_cutoff)) \cdot height \rfloor$
$loss \leftarrow \text{MSE}(dct\_image[0:v\_cutoff, 0:h\_cutoff],$
$original\_image\_dct[0:v\_cutoff, 0:h\_cutoff])$
**return** *loss*

The idea behind our H-DCT loss is to first compare the low frequency components of the image and then as the optimization progresses try to recover the higher frequencies too. The parameter $\phi$ defines how quickly the loss moves from comparing the lowest frequencies only, to the whole components of the DCT. This is to help the optimization converge more smoothly as initially a lot of dots are moved into the volume creating low resolution images which then get adjusted progressively as the optimization continues.

Note that all loss components never directly operate on the halftone image, instead we first apply a Gaussian filter and then compute each loss on the "blurred" version of the image. The Gaussian filter is defined as $G_{\sigma,w}(\cdot)$ where $\sigma$ is the standard deviation and $w$ is the window size. In our evaluation we use a window size of 11 and a standard deviation of 1.5. We set our weights for the losses to the values of 1 for the TONE and STRUCT losses and 5 for the FREQ component. We use the same set of weights for all the images halftoned with our method shown in the paper.

As each loss has wildly different values we perform a normalization step by trying to estimate the maximum of each loss. To do so we compute the value of each loss given the original image $\mathbf{I_c}$ and an image composed purely of noise. Given the vector $\mathbf{m}$ of estimates the normalized loss is then simply:

$$\mathcal{L}_n^i = \frac{\mathbf{1}}{\mathbf{m}} \odot \mathcal{L}^i. \tag{19}$$

where $\odot$ is the Hadamard product.

### 4.7. Interactive Loss Tuning

We find the weights of the loss using a tool that we built to perform a continuous adjustment of each one. The tool shows at all time 3 images to the user. The original continuous tone image, the current result of the halftoning and the result of the halftoning from $x$ (optimization) iterations ago. The halftoning can either be one-shot or continuous. In the first case, the image will be halftoned with the given weights and the results shown to the user. In the continuous mode, a background thread continuously runs the optimization, updating the shown image each preset number of iterations. Changing the values of the weights of the loss in this mode will not result in an immediate change of the loss function but rather a smooth change. We use a simple interpolation scheme such that the loss function at each iteration is:

$$\mathbf{w_{next}} = \alpha \mathbf{w_{current}} + (1-\alpha)\mathbf{w_{target}}. \tag{20}$$

This assures that the position of the minimum of the loss function changes smoothly, helping the optimization achieve a stable change in the image. While the possibility to adjust the loss is not unique to our method, we do provide an almost real-time visualization of the effects of changing the loss function. The best competing methods, i.e., deep learning methods [XHLW21b, GS20] require retraining while methods based on stochastic optimization [PQW*08] are too slow to provide real-time adjustment.

## 5. Evaluation

In this section, we discuss the experimental results of our proposed search-based halftoning method. The evaluation provides an analysis of the efficiency of our formulation as well as the quality of the results. We also perform several ablations, including on the loss and different initialization.

### 5.1. The Performance of Our Formulation

First and foremost, we are interested in evaluating our proposed relaxed optimization in terms of the power of our formulation to reduce the loss value. For this purpose, we halftone an image using 21 different loss functions and report the objective value of the optimization in Table 2. For reference, we compare the objective value of our formulation with that of structure-aware halftoning (SAH) [PQW*08], a prominent search-based halftoning. The employed loss function, for both methods, is a different linear combination of TONE and STRUCT. We choose this set of loss functions as the base C++ implementation of SAH works only with this combination. As evident in Table 2, our formulation achieves a smaller objective value, i.e., a better loss, consistently. This result is not due to the optimization per se but also due to the additional freedom we allow the formulation in choice of dots. SAH is limited to swapping around pixel as to preserve the total "grayness" of the image, but we are not bound by such arbitrary constraint.

Next, we analyze the scalability of our method with respect to the size of the input image. This is a critical aspect since today's printing industry has extreme scale requirements, as both the hardware resolution and the printing size have undergone significant increases. This explains the continuing popularity of classic methods, such as error diffusion or smart thresholding. Therefore, scalability to larger resolutions is essential for any search-based method to be extensively utilizable. We measure our method's halftoning time and required memory as a function of the image size.

In Figure 4 left we can see the results for the memory usage of our method compared to Xia *et al.* deep halftoning method [XHLW21a] and Pang *et al.* structure-aware halftoning [PQW*08]. The growth rate of our memory usage is much slower and our top memory usage is 13.1 GB for a 8521 x 5969 image. This is a very substantial amount of memory but within the limits of modern GPU capabilities. SAH is extremely scalable memory-wise as it is an in-place algorithm that uses just a $\mathcal{O}(2|\mathbf{I}|)$ of memory.

In the same Figure 4 on the right we compare the elapsed time

**Table 2:** *Comparison of error values for different MSE/SSIM weighting configurations.* ↓

| Weight (MSE) | 0.00 | 0.05 | 0.10 | 0.15 | 0.20 | 0.25 | 0.30 | 0.35 | 0.40 | 0.45 | 0.50 | 0.55 | 0.60 | 0.65 | 0.70 | 0.75 | 0.80 | 0.85 | 0.90 | 0.95 | 1.00 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Weight (SSIM) | 1.00 | 0.95 | 0.90 | 0.85 | 0.80 | 0.75 | 0.70 | 0.65 | 0.60 | 0.55 | 0.50 | 0.45 | 0.40 | 0.35 | 0.30 | 0.25 | 0.20 | 0.15 | 0.10 | 0.05 | 0.00 |
| SAH | 0.341 | 0.330 | 0.319 | 0.308 | 0.297 | 0.287 | 0.276 | 0.265 | 0.254 | 0.243 | 0.232 | 0.222 | 0.211 | 0.200 | 0.189 | 0.178 | 0.167 | 0.157 | 0.146 | 0.135 | 0.138 |
| Ours | **0.138** | **0.164** | **0.169** | **0.172** | **0.173** | **0.172** | **0.170** | **0.172** | **0.173** | **0.170** | **0.171** | **0.168** | **0.170** | **0.163** | **0.160** | **0.156** | **0.151** | **0.145** | **0.140** | **0.134** | **0.126** |

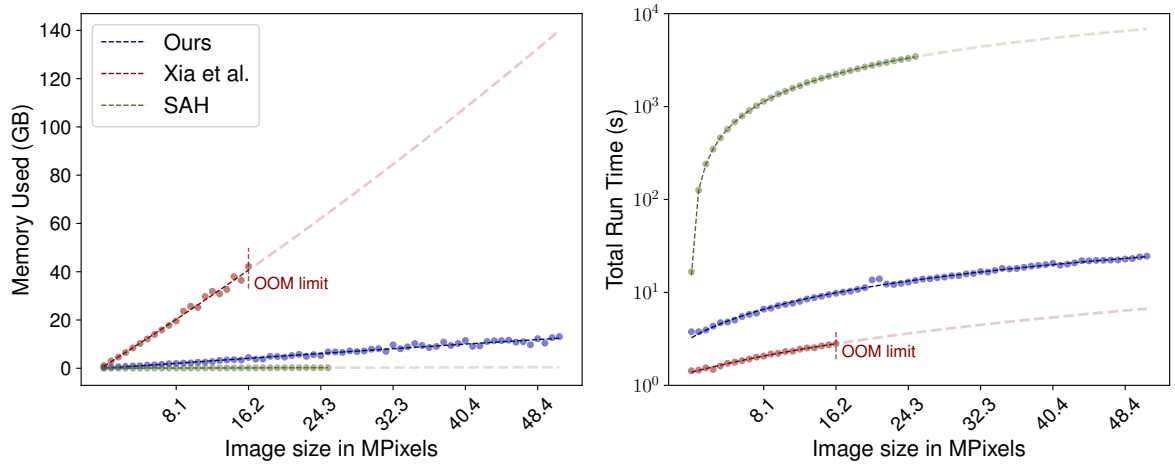



**Figure 4:** *Comparison of Pang, Xia and Ours method when halftoning the same image scaled at different resolutions from a base resolution of 266x381 up to 5969x8521 pixels. Xia's method runs out of memory on our NVIDIA Tesla H100NVL 94GB. Pang's method is prematurely stopped when the time to halftone the image surpasses the hour.*

between different algorithms. Our method performs 200 steps before stopping, this has been chosen by observing the loss curve and finding at which iteration it started to flatten out. Pang's method stops when the temperature reaches a certain threshold, which we set to 0.01 as in the original paper. Xia's method is a deep method and the timing is given by how long it takes to run the inference. We can see that method by Xia *et al*. [XHLW21a] (a deep learning method) and our method follow a similar scaling and comparable running time. SAH on the other hand has a completely different scaling and the flattening at the top is explained more by the choice of annealing strategy than by an actual reach of a good loss value.

### 5.2. Testing Different Halftoning Losses

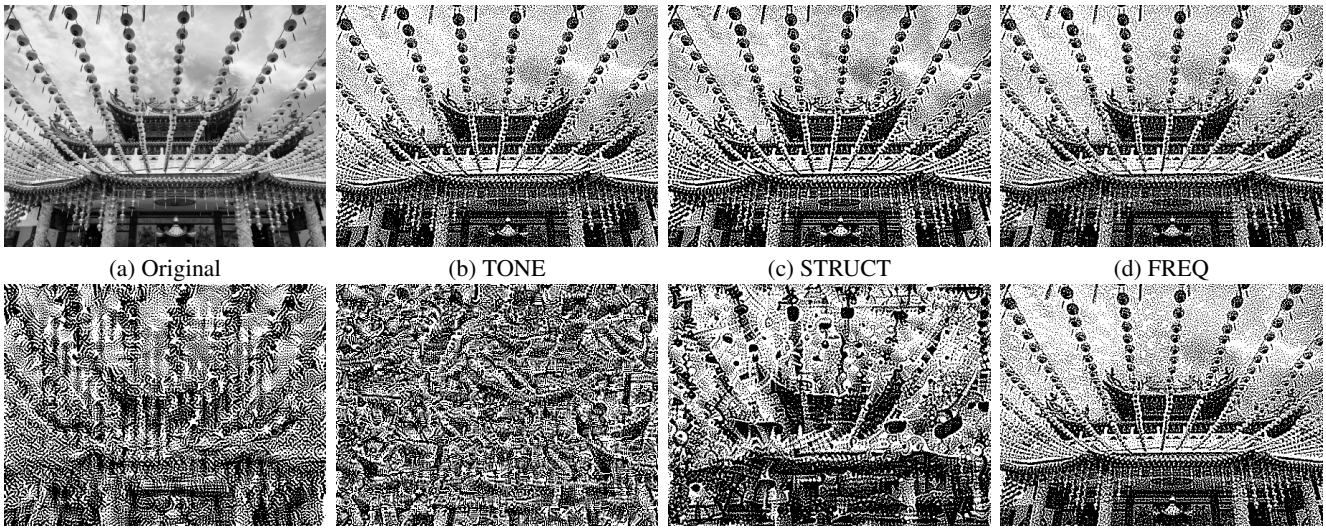
Figure 5 illustrates the optimization results when only a single loss component is active at a time. Despite each image achieving a minimum for its respective loss, none exhibit satisfactory visual quality. Only preserving the TONE leads to loss of fine details, the STRUCT loss alone produces oversharpened images and FREQ leads to pleasing uniform areas but otherwise a lack of fine details and tone preservation. This highlights the need for a composite loss function that integrates multiple loss terms.

An interesting observation is that, when used in isolation, simple heuristic-based losses (TONE, STRUCT, and FREQ) significantly outperform data-driven, feature-based similarity losses (LPIPS, ID-MRF, and DISTS). While LPIPS [ZIE*18] is capable of capturing the general texture of an image, such as pointy roofs, lanterns, and pillars, it fails to preserve finer details. This reflects the well-known bias in deep neural networks: a tendency to prioritize textural cues over object shape, especially when trained for classification tasks [GJM*20]. This bias is even more pronounced in ID-MRF [WTQ*18], which evaluates similarity based on feature patches without accounting for their spatial arrangement. As a result, strong textural elements (e.g., pointy roofs) appear frequently across the halftoned image, regardless of their original position. Note that DISTS [DMWS20] can recover finer details to some extent because it measures both textural similarity and structural similarity via global feature means and global feature correlations, respectively.

### 5.3. Multi-level, Multi-color Differentiable Search-based Halftoning

Our method can natively be used for multi-class scenarios [SGSS22], such as multi-level halftoning. Recall from Eq. 4, where we produce values for the assignment, that we can define as many cases as we want for the value that we assign to the selection $s^d_{x,y}$. In Figure 6 we show extension to different number of cases for a black and white halftoning. In each image we define **levels** $+1$ different cases, where the $+1$ corresponds to the case where the dot is not included.

Our method can also naturally perform color halftoning as shown in Figure 7. In this case we are utilizing RGB colored dots. Each pixel uses 3 dots and each dot is either fully Red, Green, or Blue. Multilevel halftoning can also be performed on color images.



(a) Original (b) TONE (c) STRUCT (d) FREQ

(e) LPIPS (f) ID-MRF (g) DISTS Ours

**Figure 5:** *Comparison of different halftoning results with only 1 component of the loss and our combined loss shown at the end. Every loss optimizes for a different visual component but none of them single-handedly are enough to obtain a good halftoning.*

Original **1**+1 **2**+1 **4**+1 **8**+1 **16**+1

**Figure 6:** *Comparison of different halftoning using a different number of discretization levels*

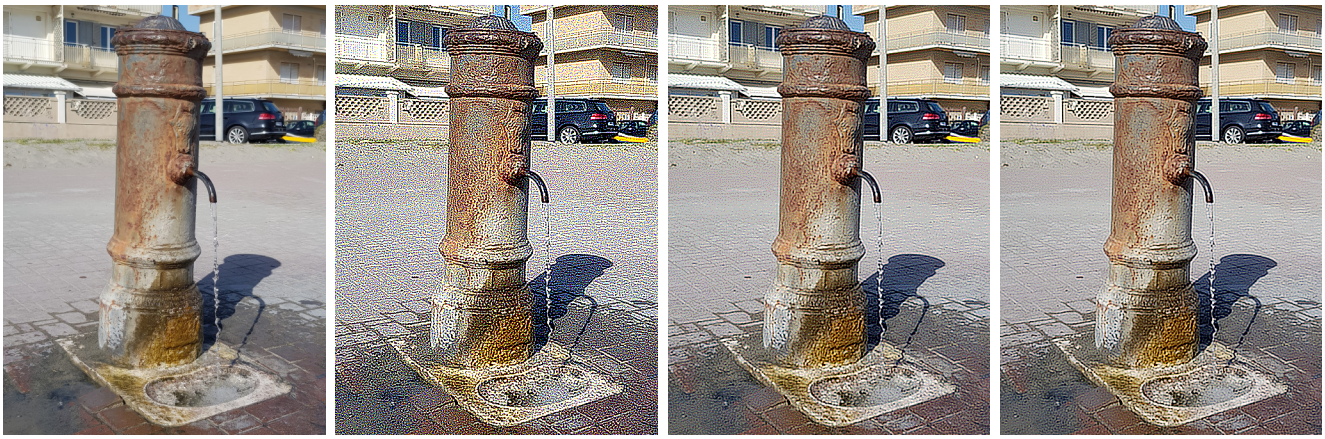
for the assignment. Already from 4 different levels, which is obtainable with modern printing hardware, we have a very smooth image.

Finally a peculiarity of our method is that it can perform color halftoning using an arbitrary choice of dots colors. Recall that each dot has a distinct type $\mathbf{t}^d_{x,y} \in \mathbb{R}^K$ that maps it into a color space. We are free to choose any values for the elements of this vector. As an assignment selects dots, not colors, the optimizer is forced to minimize the loss function using whatever dots it has at its disposal. If we give it dots of which color has nothing to do with the original image we obtain some peculiar results. In Figure 8 we see the results of such mappings. We first show a halftoned image where the colors of the dots are R,G, and B respectively, leading to a normal colored image. Then we alter the colors of the dots such that only yellow and green-blue dots are available and similarly for the last image where we supply only blue and red dots.

### 5.4. Robustness to Initialization

Some methods require careful initialization in order to work properly. Our method instead is very robust to different initialization (of dot potentials), although some differences can still be observed. In Figure 9 we show the results of such different initializations. We initialize the positions of the dots according to Eq. 10 for images (b) and (c), where the $\epsilon$ term is sampled from a white and blue noise distribution respectively. Image (d) is initialized according to Eq. 11 using Ostromokhouv's method as initial halftoned image. Finally, we show the result of failing to supply a good noise source in image (e), where we set all the dot position to a constant value.

A particular advantage of our method is that it does not need a



Original **1**+1 **4**+1 **8**+1

**Figure 7:** *Comparison of different color halftoning with multilevel. The addition of color increases the different values a pixel can assume making the 4 level halftoning already almost indistinguishable from higher multilevels.*

Original RGB CMYK Yellow-Blue

**Figure 8:** *Comparison of an RGB image halftoned by using dots of different types. When the dots are also directly mapped to RGB space the halftoning achieves a high color fidelity. Nonetheless when an unusual palette of dots is supplied the optimizer still finds a minimal solution for the given palette.*

careful setting of the number of initial dots. Recall that Jiang *et al.* [JZM24] develop a totally separate neural network to compute the number of dots given the input image. Figure 10 shows the results initializing two search based methods (ours and [PQW*08]) with a lower number of dots than necessary. We can see that other search-based methods [PQW*08] fail to recover the missing dots, resulting in halftone images that are brighter than the reference image. In contrast, our method can recover the missing dots and reproduce the reference image with the correct grayness level.

Finally in Figure 11 we show a comparison of different initial assignments of our method when compared to Pang's method. As we can see our method is much more robust to different halftoned initializations.

### 5.5. Visual comparison

In Figure 12, we provide a visual comparison of different halftoning methods, each using its own image metrics. Such a comparison is unfortunately inherently ill-posed as "image quality" is highly subjective. Nonetheless we can agree that halftoning methods should not produce artifacts in the image. Examples of such artifacts include low-frequency noise components or highly regular structures that are visually unpleasing and unfaithful to the original image. Both Xia *et al.* [XHLW21a] and our method are free of such artifact but we can notice them in Pang's method as it carries them over from its initialization which is based on error diffusion.

## 6. Conclusion

We developed a differentiable dot based halftoning method. By embedding the 2D halftoning problem into 3D space, we allow

(a) Original (b) White (c) Blue (d) Ostromokhouv (e) Uniform

**Figure 9:** *Comparison of different halftoning results with different initializations. We can observe that our method is working properly with very different initial guesses but predictably fails when no noise is provided.*

(a) Original (b) Brightened (c) Ostromokhouv (d) Pang (e) Ours

**Figure 10:** *We brighten the reference image (a) by reducing its intensity resulting in image (b). We generate image (c) using Ostromokhouv's method, which has around 50% less dots, and use it as an initialization. Pang's method (d) fails to recover the missing dots, resulting in a halftone image that is significantly brighter than the reference image. Our method (e) recovers the missing dots and faithfully reproduce the reference image.*

dots to form images through vertical movement and plane intersection. We introduce a novel relaxation of the problem, enabling the use of gradient-based techniques to address an originally integer-based optimization. Unlike previous randomization methods, our approach allows non-differentiable dot disappearance and reappearance to be handled in a differentiable manner. Each dot is processed in parallel, significantly speeding up the computation. Our method is robust to different initializations requiring no special input to minimize the given loss. Additionally, our flexible framework supports any differentiable image quality metric for optimization. We leverage this flexibility to explore new halftoning losses that combine image statistics and perceptual metrics. Open questions still remain. Finding a good loss remains an ongoing challenge. While we present a way to tune the components of a loss, it is still challenging to find a particular chosen combination that satisfies all criteria. Finally, it would be highly interesting to see our pipeline applied to a real world printing application. Particularly, we are excited to see our method to be integrated with state of the art 3D appearance printing workflows [LPB24, SRB*19]. This is interesting from two perspectives. First, the extension of our halftoning to 3D is a considerable challenge. Second, while the novel 3D appearance printing methods arrange the dots (voxels) with respect to the resulting light transport, they do not consider this arrangement from a halftoning point of view.

## Acknowledgments

Open Access funding enabled and organized by Projekt DEAL.

## References

[AA92] ANALOUI M., ALLEBACH J. P.: Model-based halftoning using direct binary search. In *Human vision, visual processing, and digital display III* (1992), vol. 1666, SPIE, pp. 96–108. 2

[AGKN21] ABEDINI F., GOORAN S., KITANOVSKI V., NYSTRÖM D.: Structure-aware halftoning using the iterative method controlling the dot placement. *Journal of Imaging Science and Technology 65*, 6 (2021). Funding: ApPEARS (Appearance Printing European Advanced Research School); European Unions Horizon 2020 programme under the Marie Skodowska-Curie [814158]. doi:10.2352/j.imagingsci.technol.2021.65.6.060404. 2

[Ami09] AMIDROR I.: *The Theory of the Moiré Phenomenon: Volume I: Periodic Layers*, vol. 38. Springer Science & Business Media, 2009. 2

[Bay76] BAYER B. E.: An optimum method for two-level rendition of continuous-tone pictures. In *Ineternl. Conf. on Comm.* (1976), vol. 50, pp. 69–77. 2

[CA22] CHOI B., ALLEBACH J. P.: Mimicking dbs halftoning via a deep learning approach. *Electronic Imaging 34* (2022), 1–7. 2

[CAO09] CHANG J., ALAIN B., OSTROMOUKHOV V.: Structure-aware error diffusion. In *ACM SIGGRAPH Asia 2009 Papers* (New York, NY, USA, 2009), SIGGRAPH Asia '09, Association for Computing Machinery. URL: https://doi.org/10.1145/1661412.1618508, doi:10.1145/1661412.1618508. 2

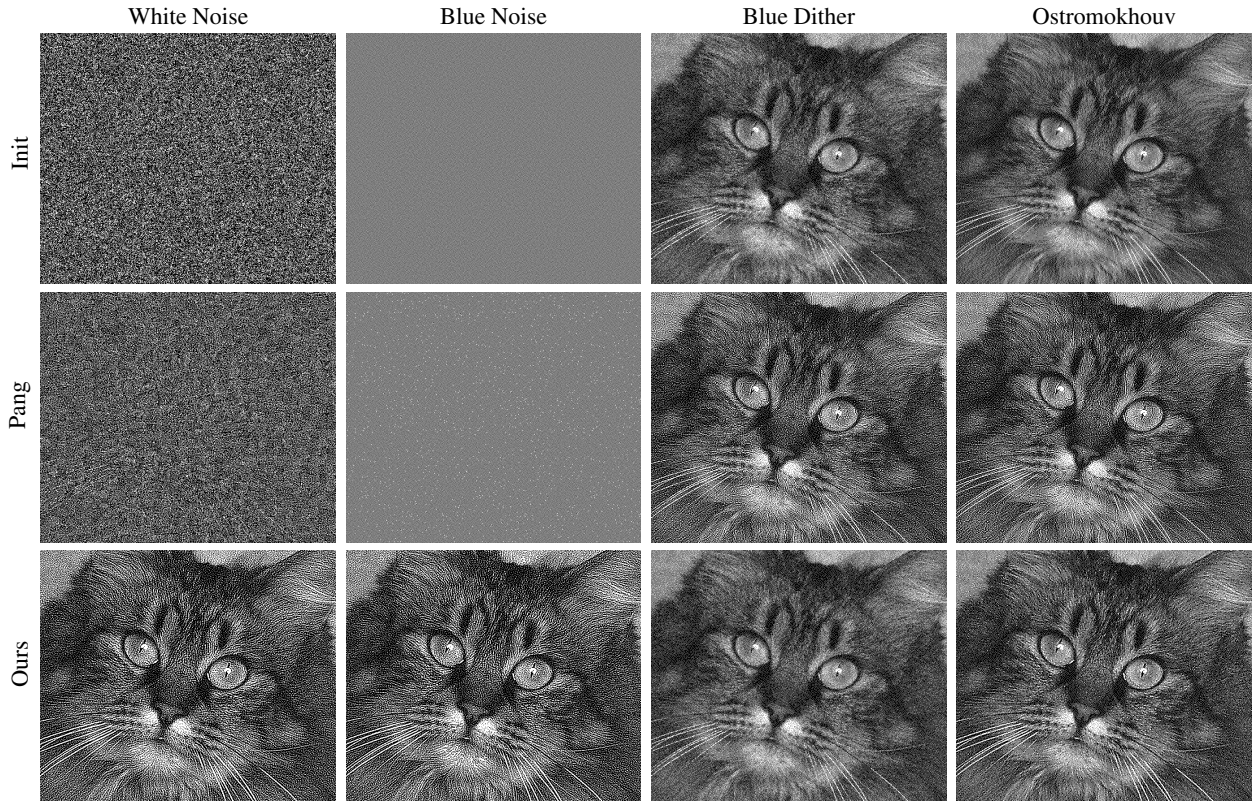



**Figure 11:** *Starting from different initialization our method yields different but similar results. Pang's method on the other hand as a difficult time optimizing from complete noise.*

[CTP12] CHATTERJEE A., TUDU B., PAUL K. C.: Towards optimized binary pattern generation for grayscale digital halftoning: A binary particle swarm optimization (bpso) approach. *Journal of Visual Communication and Image Representation 23*, 8 (2012), 1245–1259. 2

[DMWS20] DING K., MA K., WANG S., SIMONCELLI E. P.: Image quality assessment: Unifying structure and texture similarity. *CoRR abs/2004.07728* (2020). URL: https://arxiv.org/abs/2004.07728. 7

[Flo76] FLOYD R. W.: An adaptive algorithm for spatial gray-scale. In *Proc. Soc. Inf. Disp.* (1976), vol. 17, pp. 75–77. 2

[GJM*20] GEIRHOS R., JACOBSEN J.-H., MICHAELIS C., ZEMEL R., BRENDEL W., BETHGE M., WICHMANN F. A.: Shortcut learning in deep neural networks. *Nature Machine Intelligence 2*, 11 (2020), 665–673. 7

[Gra06] GRATTON M.: Backward diffusion methods for digital halftoning. 3

[GS20] GUO J.-M., SANKARASRINIVASAN S.: H-gan: Deep learning model for halftoning and its reconstruction. In *2020 IEEE International Conference on Consumer Electronics (ICCE)* (2020), IEEE, pp. 1–2. 2, 6

[JM22] JIANG H., MU Y.: Conditional diffusion process for inverse halftoning. In *Advances in Neural Information Processing Systems* (2022), Koyejo S., Mohamed S., Agarwal A., Belgrave D., Cho K., Oh A., (Eds.), vol. 35, Curran Associates, Inc., pp. 5498–5509. URL: https://proceedings.neurips.cc/paper_files/paper/2022/file/2492288f6878e6f99124b362604e58f5-Paper-Conference.pdf. 2

[JZM24] JIANG H., ZHOU B., MU Y.: Ink dot-oriented differentiable optimization for neural image halftoning. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2024), pp. 27518–27527. doi:10.1109/CVPR52733.2024.02599. 2, 9

[JZW*15] JIANG M., ZHOU Y., WANG R., SOUTHERN R., ZHANG J. J.: Blue noise sampling using an sph-based method. *ACM Transactions on Graphics (TOG) 34*, 6 (2015), 1–11. 3

[Kan99] KANG H. R.: Digital color halftoning. *SPIE/IEEE Series on Imaging Science and Engineering* (1999). 2

[KP18] KIM T.-H., PARK S. I.: Deep context-aware descreening and rescreening of halftone images. *ACM Transactions on Graphics (TOG) 37*, 4 (2018), 1–12. 2

[LH17] LOSHCHILOV I., HUTTER F.: SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations* (2017). URL: https://openreview.net/forum?id=Skq89Scxx. 5

[LPB24] LUCI E., PELLACINI F., BABAEI V.: Differentiable modeling of material spreading in inkjet printing for appearance prediction. In *SIGGRAPH Asia 2024 Conference Papers* (2024), pp. 1–10. 10

[MEMS14] MOHAMMADI P., EBRAHIMI-MOGHADAM A., SHIRANI S.: Subjective and objective quality assessment of image: A survey, 2014. URL: https://arxiv.org/abs/1406.7799, arXiv:1406.7799. 3

[MSL*25] MA C., SHI Z., LU Z., XIE S., CHAO F., SUI Y.: A survey on image quality assessment: Insights, analysis, and future outlook, 2025. URL: https://arxiv.org/abs/2502.08540, arXiv:2502.08540. 3

(a) Original (b) Pang (c) Xia (d) Ours

**Figure 12:** *Visual comparison of different halftoning methods. All three methods produce satisfactory results however low-frequency components can be observed in Pang's method as a result of the initialization with an error diffusion halftoned image. In particular vertical structures can be observed in the clouds and wood panels. Diagonal "worms" can also be seen in the sky. This is caused by the error diffusion scanline pattern that the algorithm uses to diffuse the remainder to the next pixel.*


[Näs84] NÄSÄNEN R.: Visibility of halftone dot textures. *IEEE transactions on systems, man, and cybernetics*, 6 (1984), 920–924. 2

[Ost01] OSTROMOUKHOV V.: A simple and efficient error-diffusion algorithm. *Proceedings of the ACM SIGGRAPH Conference on Computer Graphics* (07 2001). doi:10.1145/383259.383326. 2

[PQW*08] PANG W.-M., QU Y., WONG T.-T., COHEN-OR D., HENG P.-A.: Structure-aware halftoning. *ACM Transactions on Graphics (SIGGRAPH 2008 issue) 27*, 3 (2008), 89:1–89:8. 2, 3, 5, 6, 9

[SGBW10a] SCHMALTZ C., GWOSDEK P., BRUHN A., WEICKERT J.: Electrostatic halftoning. In *Computer Graphics Forum* (2010), vol. 29, Wiley Online Library, pp. 2313–2327. 3

[SGBW10b] SCHMALTZ C., GWOSDEK P., BRUHN A., WEICKERT J.: Electrostatic halftoning. *Computer Graphics Forum 29*, 8 (2010), 2313–2327. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2010.01716.x, arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/j.1467-8659.2010.01716.x, doi:https://doi.org/10.1111/j.1467-8659.2010.01716.x. 3

[SGSS22] SALAÜN C., GEORGIEV I., SEIDEL H.-P., SINGH G.: Scalable multi-class sampling via filtered sliced optimal transport. *ACM Trans. Graph. 41*, 6 (Nov. 2022). URL: https://doi.org/10.1145/3550454.3555484, doi:10.1145/3550454.3555484. 7

[SGW12] SCHMALTZ C., GWOSDEK P., WEICKERT J.: Multi-class anisotropic electrostatic halftoning. In *Computer Graphics Forum* (2012), vol. 31, Wiley Online Library, pp. 1924–1935. 3

[SRB*19] SUMIN D., RITTIG T., BABAEI V., NINDEL T., WILKIE A., DIDYK P., BICKEL B., KŘIVÁNEK J., MYSZKOWSKI K., WEYRICH T.: Geometry-aware scattering compensation for 3d printing. *ACM Transactions on Graphics 38*, 4 (2019). 10

[WBSS04] WANG Z., BOVIK A. C., SHEIKH H. R., SIMONCELLI E. P.: Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing 13*, 4 (2004), 600–612. 5

[WTQ*18] WANG Y., TAO X., QI X., SHEN X., JIA J.: Image inpainting via generative multi-column convolutional neural networks. In *Advances in Neural Information Processing Systems* (2018), pp. 331–340. 7

[WW17] WONG K.-M., WONG T.-T.: Blue noise sampling using an n-body simulation-based method. *The Visual Computer 33* (2017), 823–832. 3

[WWH13] WU H., WONG T.-T., HENG P.-A.: Parallel structure-aware halftoning. *Multimedia Tools Appl. 67*, 3 (dec 2013), 529–547. URL: https://doi.org/10.1007/s11042-012-1048-6, doi:10.1007/s11042-012-1048-6. 2

[XHLW21a] XIA M., HU W., LIU X., WONG T.-T.: Deep halftoning with reversible binary pattern. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 14000–14009. 2, 5, 6, 7, 9, 13

[XHLW21b] XIA M., HU W., LIU X., WONG T.-T.: Deep halftoning with reversible binary pattern. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)* (October 2021), pp. 14000–14009. 2, 6

[YLQS23] YANG J., LYU M., QI Z., SHI Y.: Deep learning based image quality assessment: A survey. *Procedia Computer Science 221* (2023), 1000–1005. Tenth International Conference on Information Technology and Quantitative Management (ITQM 2023). URL: https://www.sciencedirect.com/science/article/pii/S1877050923008384, doi:https://doi.org/10.1016/j.procs.2023.08.080. 3

[ZIE*18] ZHANG R., ISOLA P., EFROS A. A., SHECHTMAN E., WANG O.: The unreasonable effectiveness of deep features as a perceptual metric. *CoRR abs/1801.03924* (2018). URL: http://arxiv.org/abs/1801.03924, arXiv:1801.03924. 7

## Appendix

We include here additional experiments and ablations of our method.

### 6.1. Gradient Estimator

The relaxation we introduced requires us to define a nearness function when computing the selection vector. In Eq. 8 we show a possible choice for such a function. In Fig. 13 we show results of the optimization for different function choices. The simplest estimator is a linear one where the selection function is simply the opposite of the distance.

$$\hat{s}^d_{x,y} = c - \left|(\tau - z^d_{x,y})\right|. \tag{21}$$

We then show a quadratic estimator:

$$\hat{s}^d_{x,y} = c - (\tau - z^d_{x,y})^2, \tag{22}$$

where $c$ is some positive constant. Finally we show a sigmoid estimator, which is closer to our original formulation:

$$\hat{s}^d_{x,y} = \frac{1}{1 + \exp((\tau - z^d_{x,y})^2)}, \tag{23}$$

We also compare the produced figures with our estimator. A good estimator should have a smooth first derivative that flattens out as it approaches a certain distance value. Linear and quadratic estimators posses the first but not the second property, making them poor choices.

### 6.2. Loss Search

In Section 4.7, we describe a method to interactively look for the weights to assign to each loss component. While our method is relatively fast, it is not real-time. Therefore, the interactivity is achieved by starting each successive optimization from the result of the previous one with the rationale that the position of the minimum in the loss space will change smoothly as the user is changing the weights. As the space is non-convex, however, this does not guarantee that the result will be the same as starting the optimization process from the initialization image. In Figure 14, we perform such a test to show the difference between the image found after repeatedly changing the weights of the losses and the one found from a fresh start. The values chosen (at random) are 0.24 for TONE, 0.34 for STRUCT and 0.42 for FREQ. We believe the images are sufficiently close, making the interactive loss tuning a valid approach.

### 6.3. Neural Losses

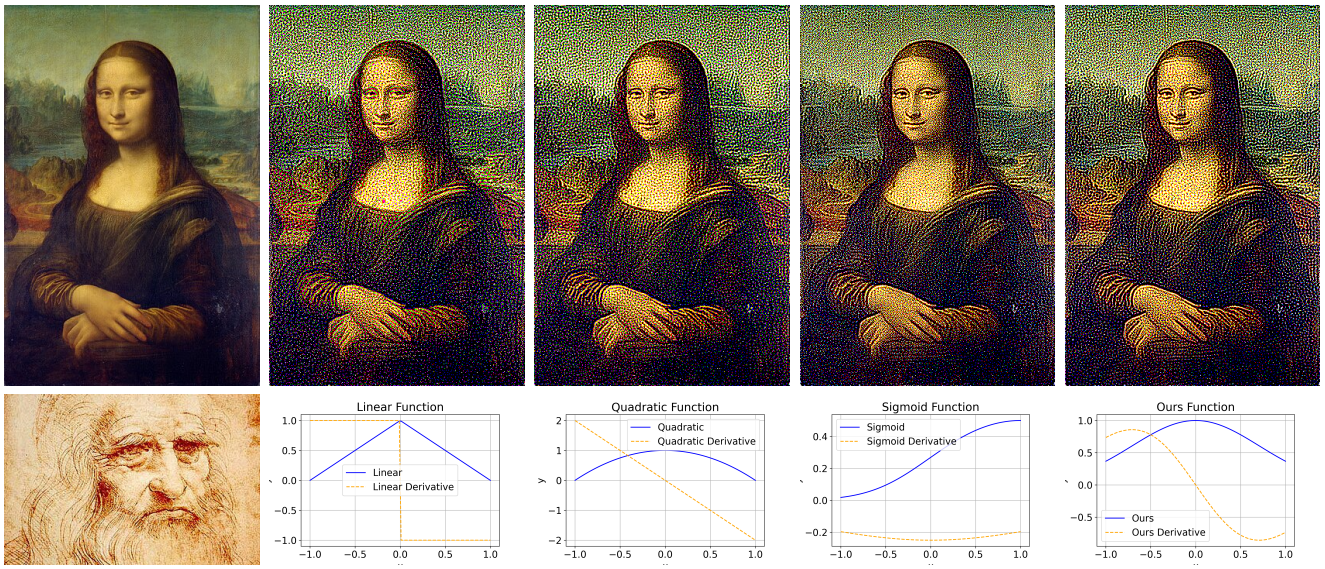
Neural perceptual losses have established themselves as powerful metrics and alternative to traditional visual losses. As we showed in Figure 5 however, they fare poorly when employed as single metrics for the halftoning problem. In Figure 15 we show the results of combining the TONE loss with a neural loss. Both losses are assigned a weight of 1. We can observer that combining the perceptual losses with a traditional per-pixel loss produces much better results. This marginal improvement however comes at an additional cost in performance as all the neural losses are relatively slow to compute. Furthermore they all require a significant amount of VRAM. Halftoning the picture shown which is just $417 \times 417$ require a whooping 66367MiB for the ID-MRF loss!

### 6.4. Periodogram

In Figure 16 we show a periodogram of a halftoned uniform gray (RGB = 0.7) picture. We can see that the periodogram shows a blue noise power spectrum. We do not directly inject such a distribution into the image, the "blueness" is purely the result of the optimization process.

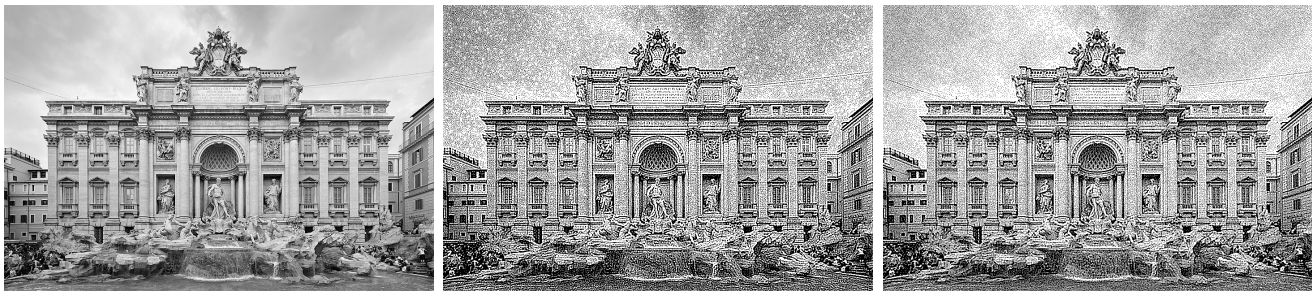
### 6.5. Performance on Consumer GPU

The experiments presented in section 5.1 utilized an NVIDIA Tesla H100NVL 94GB card since the method by Xia *et al*. [XHLW21a] requires significant amounts of VRAM. In Figure 17 we show the timing (and memory) of our method for the same test on a consumer NVIDIA GeForce RTX 3090. The memory requirements naturally stay consistent but the timing do increase by an order of magnitude. Interestingly, some upscaling resolutions seem to consistently (the test is averaged over 8 runs) run much faster.




(a) Original (b) Linear (c) Quadratic (d) Sigmoid (e) Ours

**Figure 13:** *Comparison of different halftoning results with different gradient estimators. All estimators produce reasonable results. Higher noise and artifacts can be observed in the linear and quadratic estimator. Slight differences can also be noticed in the hair and background between ours and the sigmoid estimator. This last one in particular seems to produce a different noise pattern in the background.*



(a) Original (b) Search (c) Single

**Figure 14:** *Results of halftoning an image using our interactive loss tuning tool (b) and halftoning the image from scratch using the same parameters (c). While Obvious differences can be observed especially in the sky region the images are similar enough that it justifies the utility of the loss tuner.*

(a) Original (b) LPIPS (c) ID-MRF (d) DISTS (e) Ours

**Figure 15:** *When combined with a classical TONE loss, neural losses perform better than when used in isolation. This however does not justify their overhead in terms of time and memory w.r.t. non-deep losses for the halftoning problem.*

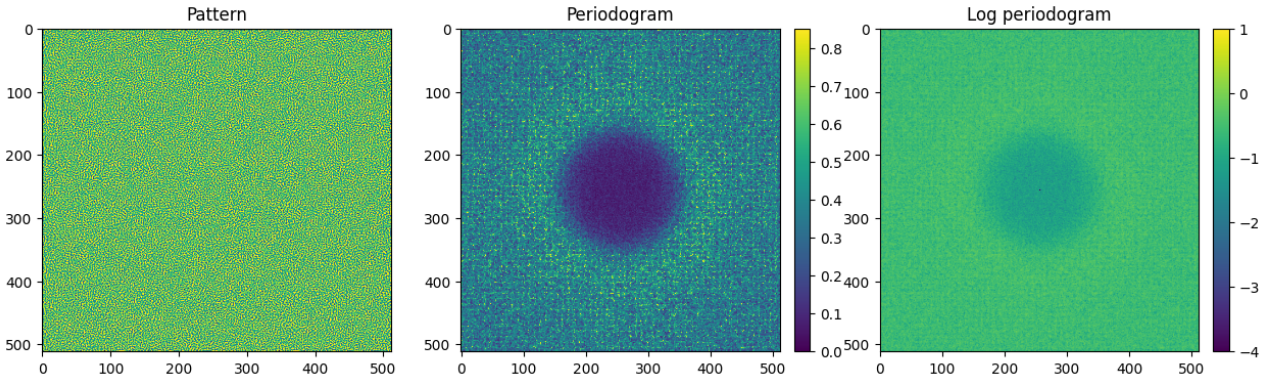



**Figure 16:** *Periodogram of a halftone uniform gray image. On the left we show the result of the halftoning. On the centre and right the periodogram and its log. The periodogram resembles that of 2D blue noise distribution.*

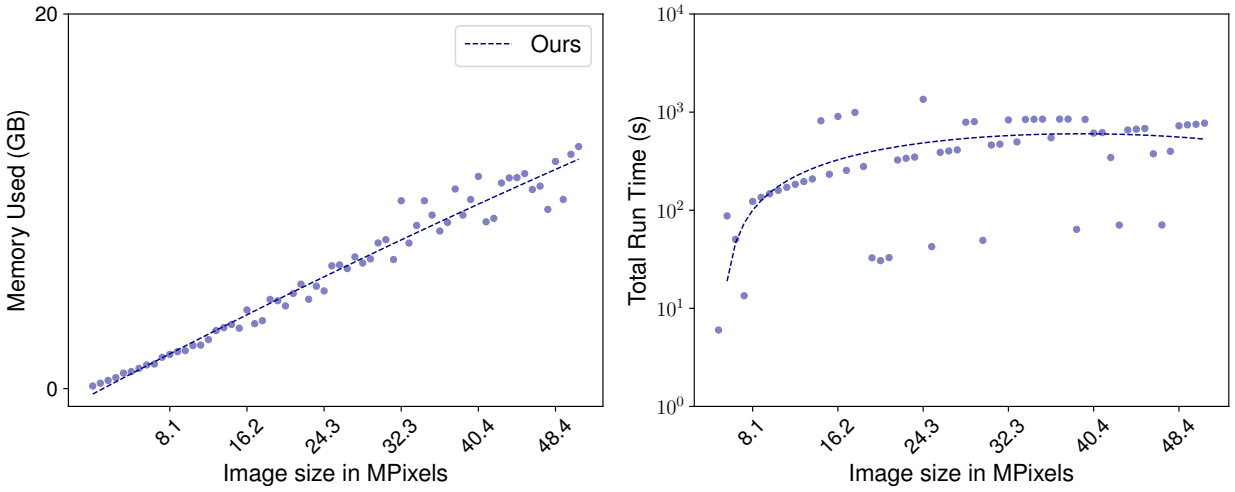



**Figure 17:** *Comparison of time and memory requirements for our method run on an NVIDIA GeForce RTX 3090. The curve resembles the one shown in Figure 4 but curiously we consistently observe much faster halftoning for some resolutions. The results are averaged over 8 runs.*