# DIGITAL IMAGING AND HALFTONING ALGORITHMS DESIGN: PARALLEL PROCESSING, PRINTED IMAGE ARTIFACTS MODELING, AND LEARNING-BASED IMAGE ANALYSIS

by

**Yafei Mao**

**A Dissertation**

*Submitted to the Faculty of Purdue University*

*In Partial Fulfillment of the Requirements for the degree of*

**Doctor of Philosophy**

School of Electrical and Computer Engineering

West Lafayette, Indiana

December 2021

**THE PURDUE UNIVERSITY GRADUATE SCHOOL**
**STATEMENT OF COMMITTEE APPROVAL**

**Prof. Jan P. Allebach, Chair**
School of Electrical and Computer Engineering

**Prof. Amy R. Reibman**
School of Electrical and Computer Engineering

**Prof. George T. Chiu**
School of Mechanical Engineering

**Prof. Mary L. Comer**
School of Electrical and Computer Engineering

**Approved by:**
Dr. Dimitrios Peroulis

To my parents Yongdong Mao and Minqi Han,

for their love, support, and encouragement.

# ACKNOWLEDGMENTS

I am very grateful to many people without whom this thesis could not have been fruitful. The most important among them is my advisor Professor Jan P. Allebach. Pursuing a Ph.D. is a long journey. Without Professor Allebach's faith and investment in me, his broad vision and wise ideas, his patient and kind way of cultivating both my technical and soft skills, this journey could have been bumpy. For having this opportunity, input, and guidance, I feel extremely fortunate.

I would like to express my sincere thanks to my advisory committee Professor Amy R. Reibman, Professor George T. Chiu, and Professor Mary L. Comer. Over the years, I have received help and instructions from my committee in various ways. From lectures to research projects collaborations, I am grateful for every piece of the precious advice that they gave me.

I also owe many thanks to my schoolmates and friends. My EISL lab mates participated in the psychophysical experiment in this work. Yifei Xue, Ruotong Ji, and Yu Hong helped with testing the graphical user interface of the experiment. Zhuangzi Li shared his fun insights on neural networks with me. Jing Guo provided useful technical writing tips. Tianyu Li recommended helpful math and computer vision courses for me to take.

Moreover, I am blessed with family. Aunt Minyi Han and uncle Ji Wang listened to me and cheered me up when I was down. Aunt Minxin Han and uncle Jiangfan Li inspired me with their fun stories. Grandma Jinghua Zhao encouraged me to strive for my goals. Grandpa Xingtang Han enlightened my interest in math and science when I was little. I want to express my heartfelt thankfulness to them.

Lastly, I give my greatest gratitude to my parents Yongdong Mao and Minqi Han. They are the ones who empowered me to fly up high. No matter ordinary or outstanding, I am just their little girl. Thank them for always being supportive and giving me all the best things in the world that they can give.

# TABLE OF CONTENTS

# LIST OF FIGURES

# ABBREVIATIONS

| | |
|---|---|
| TDFED | Tone-Dependent Error Diffusion |
| HVS | Human Visual System |
| SD | Standard Definition |
| HD | High Definition |
| UHD | Ultra High Definition |
| DBS | Direct Binary Search |
| LUT | Lookup Table |
| EQGS | Equivalent Grayscale |
| HCD | Hard Circular Dot |
| IDD | Ink Drop Displacement |
| MFP | Multi-Functional Printer |
| SFFS | Sequential Forward Floating Selection |
| DAGSVM | Directed Acyclic Graph Support Vector Machine |
| SVR | Support Vector Regression |

## ABSTRACT

This thesis explores four topics in designing imaging processing algorithms. The first one is enhancing computation efficiency and implementation flexibility. We propose a parallel processing architecture for the error diffusion algorithm, allowing it to process multiple rows of image pixels simultaneously. We also develop a new loss function in the training system to minimize the weighted mean squared error in the power spectra. With this new algorithm, better halftone reproductions and higher hardware efficiency can be achieved. The second topic is improving the image quality of printed images. Two novel printer models for the direct binary search (DBS) algorithm are proposed. One is called the dot profile model. It is used to compensate dot shape irregularity errors of inkjet printers. The other one is called the ink drop displacement model. It is intended primarily to compensate for the stochastic ink drop displacement error in the prints. For both models, we first characterize the statistical properties of the printed dots and then integrate this prior knowledge in the fidelity metric which directs the image reproduction of DBS. In so doing, the image quality is greatly enhanced. The third topic is supervised learning algorithms applied to document image analysis. In this application, we develop a set of features to use with a support vector machine model to discriminate document images based on the color information and contents. The last topic is learning-based makeup shade matching. We train machine learning models to predict the best matching foundation shade based on the selfie images of a customer. Besides, a novel color correction algorithm is designed to minimize color inconsistency.

# 1. INTRODUCTION

The goodness of an image processing algorithm can be affected by various factors, including output image quality, computation and memory complexity, and applicability to real-world problems. When designing such algorithms, taking multiple factors into account at the same time is a challenging task.

This work contributes to the design of effective and efficient halftoning and learning-based computer vision algorithms. Implementations following this work resolved problems posed by the current resource limitations, at minimal cost.

More specifically, each chapter of this thesis looks at a different image processing task. In Chapter 1, a novel serpentine based error diffusion algorithm that uses tone dependent error weights and thresholds is proposed to increase the efficiency and flexibility of hardware implementations. We also propose an expanded error weight location matrix to improve the halftone quality in the extreme tones. With this new algorithm, we achieve better halftones compared to the original tone dependent fast error diffusion, especially in the quarter tones.

In Chapter 2, a dot profile model to compensate dot shape irregularity errors of inkjet printers is proposed. Previous tabular approaches for parameterizing the printer model rely on the measurements of the gray level of various printed halftone patterns. However, lots of patterns need to be printed and scanned if the printer generates large drops of colorant. To solve this problem, we propose to simulate the appearance of the rendered patterns so that the model parameters can be computed analytically. The simulation uses the mean dot as the printer dot profile and saturated addition to resolve dot overlap. Besides, we incorporate a standard definition (SD) and a high definition (HD) equivalent gray-scale representation of the printed halftone image produced by the dot profile model into the direct binary search (DBS) algorithm. Experimental results show great improvement in the mid-tone and shadow regions over the printed image halftoned by the original DBS. The HD model further enhances details in the shadows.

In Chapter 3, a novel ink drop displacement (IDD) printer model for the direct binary search (DBS) is proposed. It is intended primarily for pagewide inkjet printers that exhibit dot displacement errors. The tabular approach in the literature predicts the gray value of a printed pixel based on the halftone pattern in some neighborhood of that pixel. However, memory retrieval time and the complexity of memory requirements hamper its feasibility in printers that have a very large number of nozzles and produce ink drops that affect a large neighborhood. To avoid this problem, our IDD model embodies dot displacements by moving each perceived ink drop in the image from its nominal location to its actual location, rather than manipulating the average gray values. This enables DBS to directly compute the appearance of the final print without retrieving values from a table. In so doing, the memory issue is eliminated and the computation efficiency is enhanced. Experimental results show significant improvement in the quality of the printed image over the original DBS. Besides, the image quality obtained by the proposed approach appears to be slightly better than that obtained by the tabular approach for a specific deterministic and idealized printer model.

In Chapter 4, to enable printers to better discriminate highlighted documents, we designed a set of features in CIE $Lch(a^*b^*)$ space to use along with the support vector machine. The features include two gamut-based features and six low-level color features. By first identifying the highlight pixels, and then computing the distance from the highlight pixels to the boundary of the printer gamut, the gamut-based features can be obtained. The low-level color features are built upon the color distribution information of the image blocks. The best feature subset of the existing and new features is constructed by sequential forward floating selection (SFFS) feature selection. Leave-one-out cross-validation is performed on a dataset with 400 document images to evaluate the effectiveness of the classification model. The cross-validation results indicate significant improvements over the baseline highlighted document classification model.

Finally, Chapter 5 presents an approach to predict the color of skin-with-foundation based on a no makeup selfie image and a foundation shade image. Our approach first calibrates the image with the help of a color checker target, and then trains a supervised-learning model to predict the skin color. In the calibration stage, we propose to use three different transformation matrices to map the device dependent $RGB$ response to the reference CIE $XYZ$ space. In so doing, color correction error can be minimized. We then compute the average value of the region of interest in the calibrated images, and feed them to the prediction model. Cross-validation results show that the proposed approach can accurately make the prediction.

Overall, this work develops halftoning and learning-based algorithms toward compactness and better performance, with practical benefits.

# 2. 4-ROW SERPENTINE TONE DEPENDENT FAST ERROR DIFFUSION

## 2.1 Introduction

Digital halftoning is a method of creating the illusion of continuous-tone output through the use of a series of dots arranged differently in size or in spacing. Halftoning allows one to simulate various shades of color with one colorant, so it is an widely used technique in rendering devices that are only capable of producing a limited number of tone levels, for example printers and some displays.

According to the level of computational complexity, halftoning algorithms can be classified into three general categories: screening, error diffusion, and search-based methods. Since error diffusion renders better detail than screening while maintaining lower cost than search-based methods, it is the most popular algorithm for marking engine technologies that can stably render isolated dots, such as inkjet. In this chapter, we will focus on error diffusion.

As originally proposed by Floyd and Steinberg [1], error diffusion is a neighborhood operation that moves through the input image in a raster order, quantizing each pixel in the scan line, and feeding the error ahead to the neighboring pixels that have not yet been binarized. Despite excellent detail rendition, error diffusion sometimes creates worm-like patterns and visible structures. To solve these problems, a number of derivations and modifications of error diffusion were developed in previous research, including use of alternative scan paths [2]–[4], threshold modulation [5]–[8], variable weights [4], [9]–[11], and tone dependent parameters [12], [13]. Reference [14] is of particular note, as it provides an excellent summary of recent methods based on directly training the weights to match a desired blue noise spectral characteristic, and proposes an improvement to this approach. It also incorporates the training of the thresholds to eliminate edge sharpening.

Aside from generating visually unpleasant textures, another disadvantage of error diffusion is lack of locality. This means that when either a conventional raster or serpentine scan path [4] is used, we are not allowed to process a pixel until all pixels in its preceding scan line have been quantized. As a consequence, hardware must store the information associated with the states of the pixels that are spatially far away, which is inefficient. With the Peano

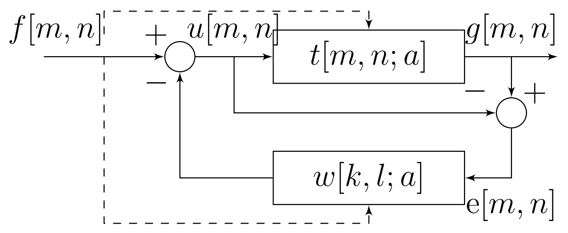


**Figure 2.1.** Tone dependent fast error diffusion system.

scan [2], the pioneer of parallel scan paths, however, the output quality of error diffusion is not satisfactory. In fact, customers in the printing market base their judgment on both print quality and implementation efficiency. Therefore, it is extremely beneficial to enable the hardware to decide the binary output locally without losing quality. In this regard, we design a novel 4-row serpentine scan path. The novelty of our approach lies in using a combination of conventional raster and serpentine scan patterns, which greatly enhances hardware efficiency. Other prior works that incorporated the concept of a serpentine raster with a novel error diffusion architecture include [15], [16].

In this chapter, we apply the 4-row serpentine scan path with the tone dependent fast error diffusion (TDFED) [17] algorithm. To reduce worm-like textures, a tone-dependent 4-weight location matrix that diffuses errors further back along the next line is designed. To further refine the halftone outputs, the weights and thresholds values for each gray level are optimized in an offline training process based on a visual cost function developed in [18]. However, the primary focus of this chapter is the development of a new error diffusion architecture that is suitable for efficient hardware implementation, rather than methods and cost functions for optimization of the weights and thresholds, which is the primary focus of [14]. In fact, the concepts introduced in this chapter could also be deployed with weights and thresholds optimized according to the methods introduced in [14]. The rest of the chapter is organized as follows: Section 2.2 demonstrates the 4-row serpentine TDFED algorithm. Section 2.3 discusses the experimental results. Section 2.4 draws the conclusions.

## 2.2 4-Row Serpentine Tone Dependent Error Defussion

We will start by providing an overview of TDFED in Sec. 2.2.1 and then present the details of the scan path. Section 2.2.3 will illustrate the expanded error location matrix. Lastly, Sec. 2.2.4 will discuss the training process.

### 2.2.1 Overview of Tone Dependent Fast Error Diffusion

Since we are presenting an error diffusion algorithm that is designed for monochrome printing devices, the pixel value is represented in units of absorptance $0 \leq a \leq 1$ , where 0 corresponds to white, and 1 corresponds to black.

Figure 2.1 illustrates the block diagram of the TDFED system. In this figure, $f[m, n] = a$ is the pixel absorptance of the continuous-tone image, $u[m, n]$ is the updated pixel value, and $g[m, n]$ is the binary output. Unlike Floyd Steinberg error diffusion, the thresholds $t[m, n; a]$ and error weights $w[k, l; a]$ of TDFED depend on the input absorptance, where $k$, $l$ are the relative position indices indicating the location of the neighboring pixels of $f[m, n]$. The neighboring pixels of $f[m, n]$ are defined by Floyd and Steinberg to be on its right, lower right, below, and lower left, assuming the image is scanned from left to right in a raster order. Since TDFED moves through the input image in a serpentine raster order, a mirror image of the weight location is adopted when scanning from right to left.

The binary output of the system is determined by thresholding the updated pixel value:

$$g[m, n] = \begin{cases} 1, & \text{if } u[m, n] \geq t[m, n; a], \\ 0, & \text{otherwise}. \end{cases} \tag{2.1}$$

The updated continuous-tone pixel value $u[m, n]$ is computed as:

$$u[m + k, n + l] \leftarrow u[m + k, n + l] - w[k, l; a] \cdot \mathrm{e}[m, n], \tag{2.2}$$

where $\mathrm{e}[m, n]$ is the quantization error. It is computed as:

$$\mathrm{e}[m, n] = g[m, n] - u[m, n], \tag{2.3}$$

and the weights $w[k,l;a]$ satisfy $\sum_{k,l} w[k,l;a] = 1$ to preserve the average local tone. To eliminate the checkboard patterns in the midtone, the threshold matrix $t[m,n;a]$ of TDFED is defined based on a halftone pattern $p[m,n;0.5]$ generated by DBS with period $128 \times 128$ for absorptance 0.5 as:

$$t[m,n;a] = \begin{cases} t_u(a), & \text{if } p[m,n;0.5] = 0, \\ t_l(a), & \text{otherwise.} \end{cases} \tag{2.4}$$

where $t_u(a)$ and $t_l(a)$ are tone dependent parameters that serve as upper and lower thresholds satisfying $t_l(a) \leq t_u(a)$. Substituting (2.4) into (2.1) yields:

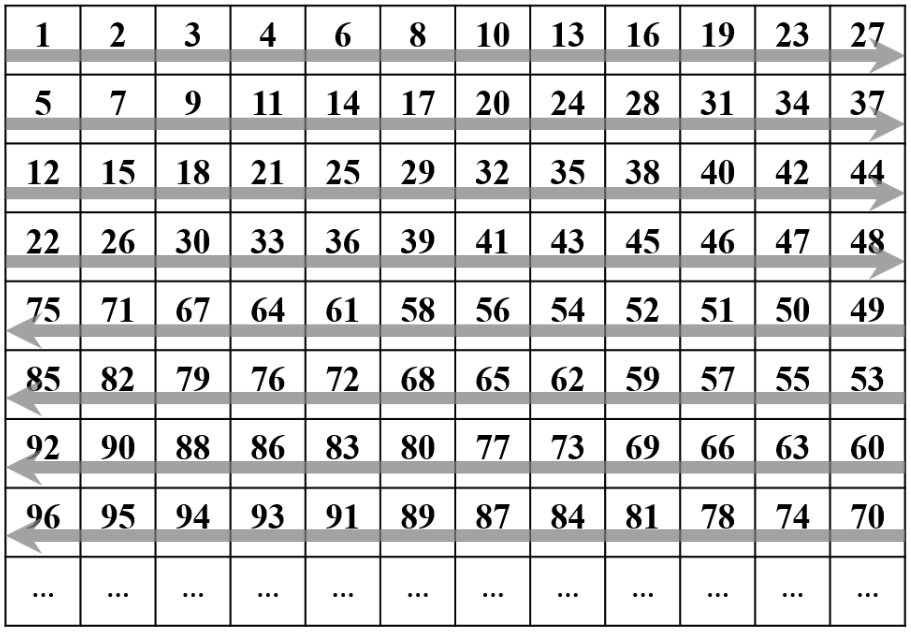
$$g[m,n] = \begin{cases} 1, & \text{if } u[m,n] \geq t_u(a), \\ 0, & \text{if } u[m,n] \leq t_l(a), \\ p[m,n;0.5], & \text{otherwise.} \end{cases} \tag{2.5}$$

In order to reduce the computational complexity, Li and Allebach chose the optimal filters for tone levels higher than 127/256 according to $t_u(a) = t_u(1-a)$ and $w[k,l;a] = w[k,l;1-a]$. The same strategy is used for all experiments in this chapter.

### 2.2.2 4-Row Serpentine Scan Path

It has been shown in [4] that implementing error diffusion in serpentine order effectively reduces the worm artifacts in the extreme gray levels. However, serpentine error diffusion is intrinsically a serial process, we must finish an entire scan line before moving on to the next one. This limits the parallelism and locality required by hardware implementations. Thus, it is necessary to mix the serpentine and raster scan together to solve the problem.

In 4-row serpentine TDFED, every 4 rows of an input image are grouped as a swath. As depicted by the arrows in Fig. 2.2, four scan lines in the same swath are processed in one direction and the consecutive four lines are processed in the opposite direction. By doing so, the serpentine property is preserved between adjacent swaths.



| | | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 2 | 3 | 4 | 6 | 8 | 10 | 13 | 16 | 19 | 23 | 27 |
| 5 | 7 | 9 | 11 | 14 | 17 | 20 | 24 | 28 | 31 | 34 | 37 |
| 12 | 15 | 18 | 21 | 25 | 29 | 32 | 35 | 38 | 40 | 42 | 44 |
| 22 | 26 | 30 | 33 | 36 | 39 | 41 | 43 | 45 | 46 | 47 | 48 |
| 75 | 71 | 67 | 64 | 61 | 58 | 56 | 54 | 52 | 51 | 50 | 49 |
| 85 | 82 | 79 | 76 | 72 | 68 | 65 | 62 | 59 | 57 | 55 | 53 |
| 92 | 90 | 88 | 86 | 83 | 80 | 77 | 73 | 69 | 66 | 63 | 60 |
| 96 | 95 | 94 | 93 | 91 | 89 | 87 | 84 | 81 | 78 | 74 | 70 |
| ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |

**Figure 2.2.** Scan path of 4-row serpentine scan with 4 pixel delay

To be more specific, inside each swath, error diffusion starts from the first pixel of the first row, and then it advances rightward along the first row in raster scan order until the next row is activated. The activation condition is that the binarization of $d$ pixels in the preceding row has been finished, where $d$ is defined as the delay between the processing of sequential lines in the 4-row swath. It can be adapted according to the need of the actual hardware architecture. Error diffusion travels back and forth across the activated scan lines. After the last pixel of the fourth row has been visited, the scan path then traverses leftward to process the next swath in the same manner described above. Figure 2.2 is an instance of the 4-row serpentine scan pattern in which $d$ equals 4 pixels. Each entry of the matrix represents a pixel of the continuous-tone input image, and the number indicates the order in which the pixel at that location is processed.

### 2.2.3 Error Weight Location Matrix

To reduce worm-like patterns with a conventional raster scan error diffusion, randomized weights [4] and a low frequency modulated threshold matrix [6] have been proposed. Unfortunately, these approaches introduce noise to the halftone image. Jarvis et al and Stucki [19] proposed a larger set of error weights with 24 terms, which requires heavy computation. Shiau and Fan [20] moved the 1/16 term in Floyd Steinberg weights from location (1,1) to (-2,1). Li and Allebach [17] developed a set of wider matrices that diffuse the errors further back. Our approach is based on [17].

As it can be seen from Fig. ?? (a), the 4-row serpentine scan pattern does not produce long diagonal structures but generates lots of short diagonal worms in the highlights and shadows. Therefore, we need to spread the quantization error over a wider region in the problematic gray levels to disperse the worms. In 4-row serpentine TDFED, diffusing the errors to a further location requires an increase in the delay. We explored delays of 2, 3, and 6 pixels, which allow an increasingly larger spatial spread in the tone-dependent weight location matrix. To ensure minimum computations, we chose to use 4 non-zero weights as originally described in Floyd and Steinberg error diffusion. The matrix set designed for 3-pixel delay is presented in Table 2.1. The weights allocation and input partition are

**Table 2.1.** Expanded error weight location matrices designed for 4-row serpentine TDFED with 4 pixel delay. The asterisk sign denotes the current pixel.

| | | | | |
|---|---|---|---|---|
| | | | * | $w_1$ |
| $w_4$ | | $w_3$ | $w_2$ | |

$\frac{1}{255} \leq a \leq \frac{63}{255}, \frac{192}{255} \leq a \leq \frac{254}{255}$

| | | |
|---|---|---|
| | * | $w_1$ |
| $w_4$ | $w_3$ | $w_2$ |

$\frac{64}{255} \leq a \leq \frac{191}{255}$

| | | |
|---|---|---|
| | * | |
| | $w_2$ | |

$a = 0,\ a = 1$

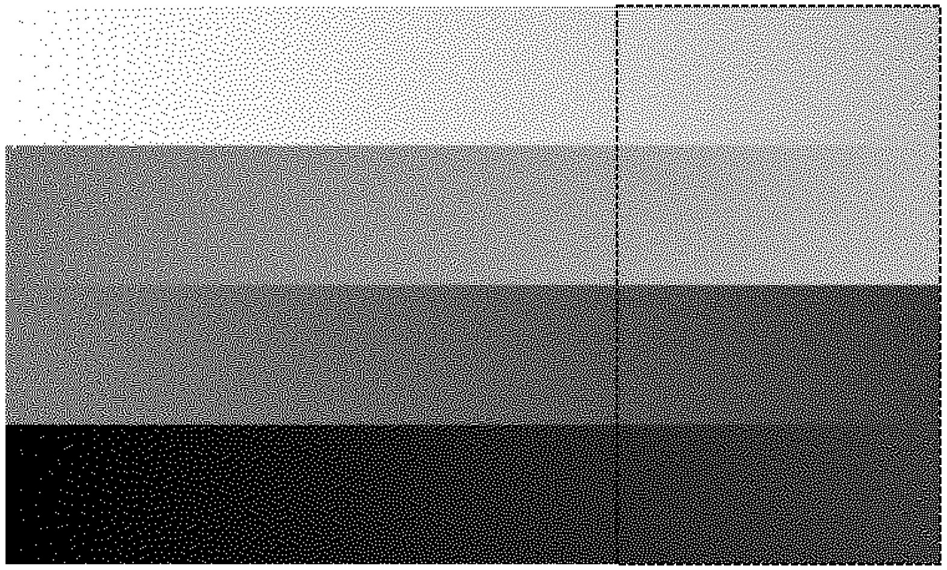
determined empirically. The values of the weights and thresholds are obtained using a search-based method, which we will discuss in the following section.

### 2.2.4 Training System for TDFED Parameters

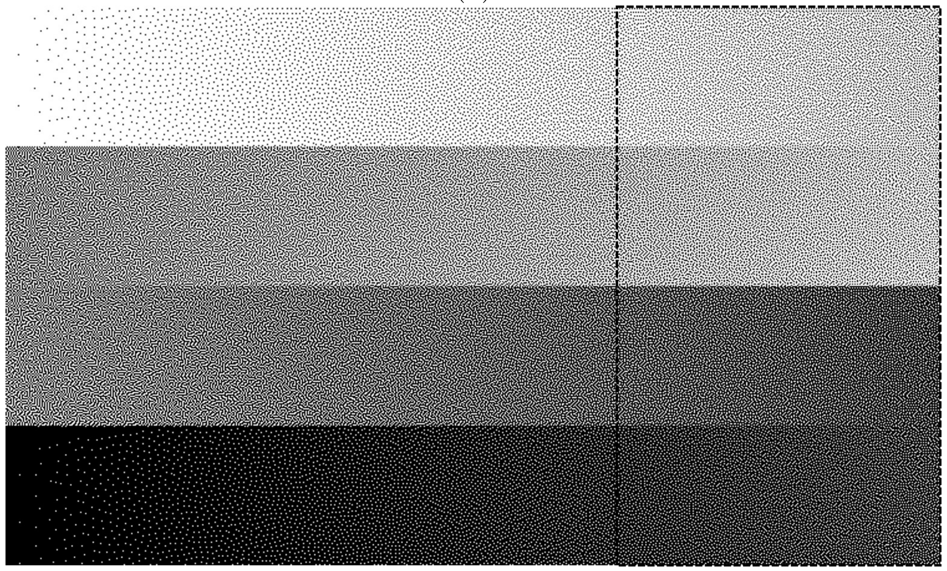
Although the expanded weight location matrix reduces worm artifacts, if the weights and thresholds are not properly designed, there will be correlated patterns and non-homogeneous textures in the output image. Thus, they must be optimized to achieve the best visual quality. We propose to use four non-zero error weights and two thresholds. There are only 4 degrees of freedom due to the constraints $\sum_{k,l} w[k,l;a] = 1$ and $t_u(a) + t_l(a) = 1$. We choose to optimize $w[0,1;a]$, $w[1,0;a]$, $w[1,1;a]$, and $t_u(a)$. Generally speaking, the TDFED training system searches for the optimal parameters by minimizing a cost function $\varepsilon$.

Li and Allebach [17] used two different cost functions depending on the tone level. For the extreme gray levels, the perceived mean squared error between the constant valued continuous-tone patch and a TDFED halftone is minimized based on Nasanen's HVS model [21]. For the midtones, the total squared error between the power spectra of the halftone patch generated by TDFED and by DBS is minimized, which is given by:

$$\varepsilon_{\mathrm{Li}} = \sum_u \sum_v (\bar{G}_{DBS}[u,v;a] - \bar{G}_{TDFED}[u,v;a])^2, \tag{2.6}$$



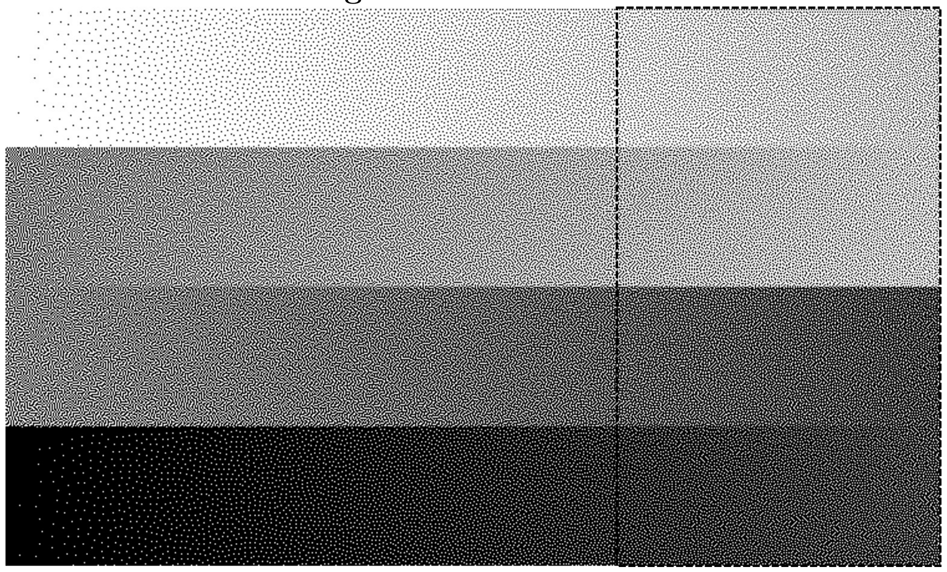
(a)

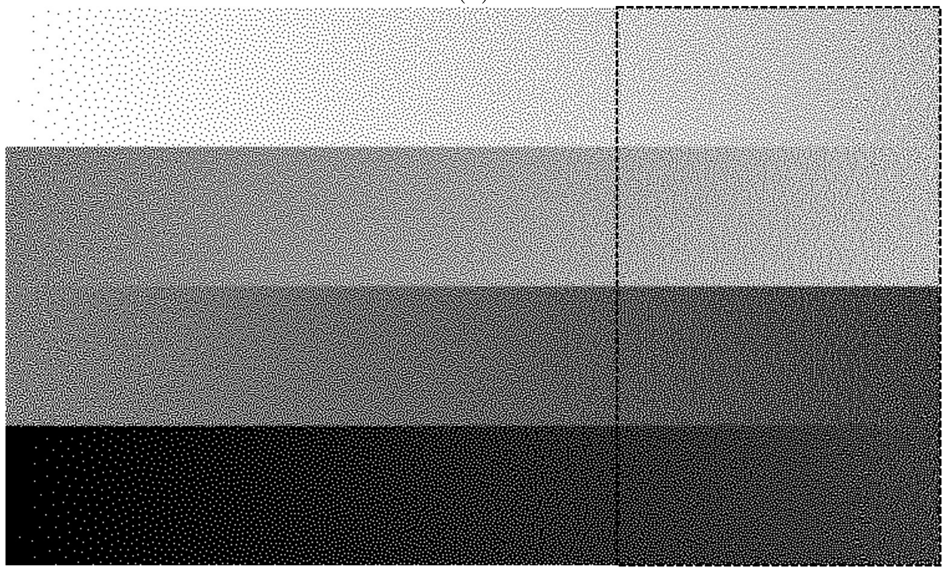


(b)

**Figure 2.3.** Halftones of a folded ramp image generated by: (a) 4-row serpentine TDFED with 2 pixel delay, (b) 4-row serpentine TDFED with 4 pixel delay, (c) 4-row serpentine TDFED with 7 pixel delay, (d) the original 1-row serpentine TDFED. The weights and thresholds are trained using the proposed scheme described in Section 2.2.4. It is recommended to zoom in on the figure so that individual pixels are clearly displayed and to view from a sufficient distance at which these individual pixels are not visually resolved. Note that (d) contains some vertical veining patterns in the region marked by the dotted box. However, all three 4-row halftones are very smooth and homogeneous in that area. This implies that the proposed 4-row algorithm outperforms the original TDFED.

**Figure 2.3.** continued.


(c)


(d)

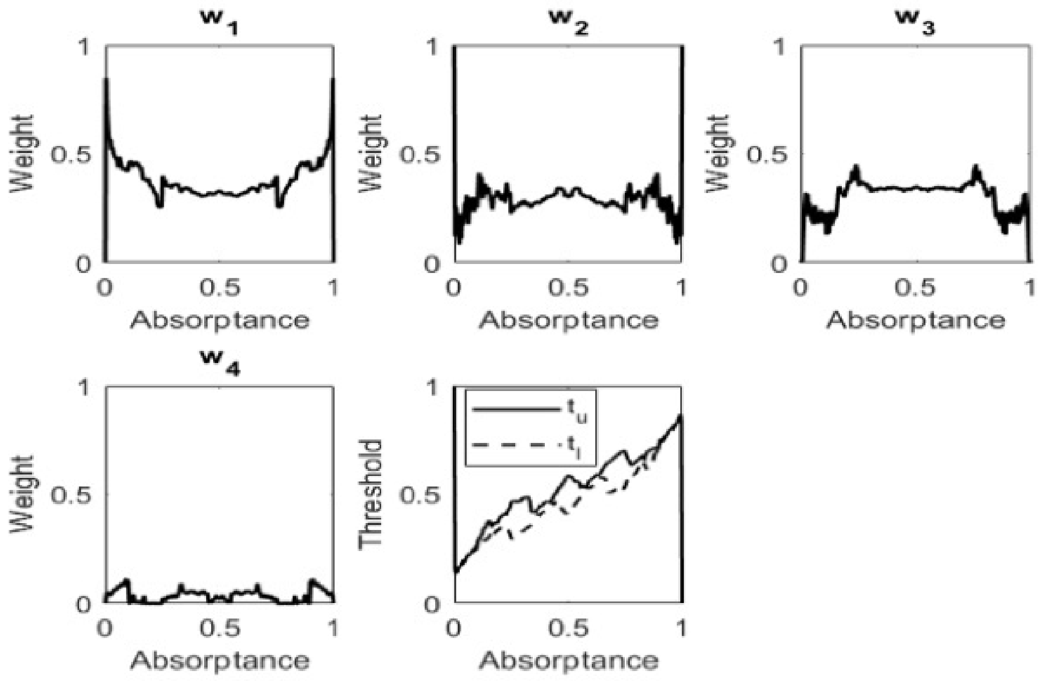


**Figure 2.4.** Optimal tone dependent weights and thresholds for 4-row serpentine scan with 4 pixel delay

where $\bar{G}_{TDFED}[u, v; a]$ and $\bar{G}_{DBS}[u, v; a]$ represent the average magnitude of the 2D Fourier amplitude spectra obtained from TDFED and DBS halftone patches, respectively.

Chang and Allebach [15] presented a cost function with a normalization as:

$$\varepsilon_{Chang} = \frac{\sum_u \sum_v (\bar{G}_{DBS}[u, v; a] - \bar{G}_{TDFED}[u, v; a])^2}{\bar{G}_{DBS}[u, v; a]^2}. \tag{2.7}$$

Han and Allebach [18] further modified the cost function by adding the power spectra of TDFED to the denominator:

$$\varepsilon_{Han} = \frac{\sum_u \sum_v (\bar{G}_{DBS}[u, v; a] - \bar{G}_{TDFED}[u, v; a])^2}{(\bar{G}_{DBS}[u, v; a] + \bar{G}_{TDFED}[u, v; a])^2}. \tag{2.8}$$

We concluded from our experiments that Han and Allebach's cost function yields the most reliable results for all levels. Thus, their cost function is adopted as the error metric.

As for the search strategy, we compared the performance of pattern search [22] with the downhill search [17], and established that downhill search is better suited to our application.

The optimized weights and thresholds of 4-row serpentine TDFED with 4 pixel delay are shown in Fig. 2.4.

## 2.3 Experimental Results

Figures ?? (a)-(c) show the result of 4-row serpentine TDFED with 2 pixel, 4 pixel, and 7 pixel delay, respectively. Figure ?? (d) is the original 1-row serpentine TDFED result. It can be observed that in the highlights and shadows, both 4-pixel and 7-pixel delay 4-row serpentine TDFED significantly reduce the short diagonal structures seen in (a) and are comparable to 1 row serpentine TDFED. Moreover, Fig. ?? (d) contains some vertical veining patterns in the region marked by the dotted square box. However, all three 4-row halftones are very smooth and homogeneous in that area.

## 2.4 Conclusion

With a modest delay value, 4-row serpentine TDFED can achieve essentially the same or better image quality than that provided by the original 1-row serpentine TDFED, except perhaps in the extreme gray levels. Besides, it will also boost efficiency and reduce memory cost in some hardware implementations.

# 3. DOT PROFILE MODEL-BASED DIRECT BINARY SEARCH

## 3.1 Introduction

Digital halftoning, an essential technique for printers, is the process of rendering a continuous-tone image with a limited number of tone levels. The direct binary search (DBS) algorithm as an iterative method, yields the best halftone reproductions [23]; so we are particularly interested in it.

Elementary halftoning algorithms assume that the ideal shape for the printed dots would be an $X \times X$ square, where $X$ is the dot spacing in inches; so $1/X$ is the printer resolution in dpi (dots per inch). However, real-world printers typically produce dots that are larger and more irregular than that. If the algorithm does not account for such printer distortions, the resulting print may contain annoying textures and distorted gray levels.

Several studies have focused on model-based techniques to mitigate printer effects. The main idea is to establish a model to predict the actual gray value of each pixel in the printed image. A popular model is the hard circular dot (HCD) model proposed in [24] and formalized in [25] and [26]. In this model, each printed pixel is assumed to be a circular spot with constant absorptance, and multiple dot overlap is resolved as a logical OR [27]–[32]. The HCD model can be used in DBS to enhance tonal reproduction and detail rendition [33] [34]. Pappas et al [35] proposed a tabular model based on the macroscopic measurements of a set of printed binary patterns in a neighborhood. Baqai et al [34] simplified the approach by using the microscopic measurement of the center pixel of all possible patterns in the neighborhood. A similar idea based on the tabular model was presented in [36].

Though these models have achieved successful results, there still are some limitations in their applicability. To be specific, the HCD model might be incompatible with printers that do not produce perfectly round and uniformly dark dots [26]. Moreover, tabular models require extensive measurements for devices that render large dots of colorant. This is because the number of possible dot configurations grows exponentially as the neighborhood size increases.

Most of the aforementioned researches were oriented to laser electrophotographic (EP) printers where the dots are very sensitive to the EP process, such that the printer cannot reliably print isolated pixels [34]. Accordingly, estimating the pixel absorptance has to rely on the interaction between adjacent pixels. In contrast, inkjet printers generally produce dots that are reasonably consistent and much more stable relative to those produced by EP printers [36]. Nevertheless, the ink dots may be irregularly shaped [37], much larger than the minimal size [38], and misaligned [36]. This suggests the use of an analytical model rather than a completely measurement-based model for inkjet printers.

To address these issues, we propose an analytical dot profile model in which arbitrarily shaped non-flat dots are studied. To parametrize the model, we first compute the dot statistics, such as the mean and standard deviation profiles, based on the data collected from individual dots. By using saturated addition to describe the effect of dot overlap, the model parameters can be computed analytically as a function of the mean dot profile and the values of the adjacent pixels. This simplifies the printer characterization process considerably because we only need to measure a limited number of isolated dots. We also assume that the ink dots are centered on their nominal locations with no misplacement.

Our goal is to incorporate the dot profile model in DBS to suppress the print artifacts caused by dot shape irregularity errors. Additionally, motivated by [39], we present both a standard definition (SD) and a high definition (HD) model. These models enable DBS to examine the visual fidelity of the rendered halftone at the printer resolution and a 3-times higher resolution, respectively. Experimental results demonstrate that by using these dot profile models, DBS can effectively improve the quality of the printed image. In particular, the HD model further refines the appearance of the shadow regions.

## 3.2 Preliminaries

### 3.2.1 Notation

An SD printer pixel $[m, n]$ is an $X \times X$ square centered at the point $(x_m, y_n)$ with $\{(x_m, y_n) : x_m = mX + \frac{X}{2}, y_n = nX + \frac{X}{2}\}$. An HD pixel $[u, v]$ is a $\frac{X}{3} \times \frac{X}{3}$ square centered at $(x_u, y_v)$ with $\{(x_u, y_v) : x_u = u\frac{X}{3} + \frac{X}{6}, y_v = v\frac{X}{3} + \frac{X}{6}\}$.

### 3.2.2 Printer Characterization

The first step towards printer modeling is to characterize the printer. We design a test page to directly capture the profile of each isolated dot. The idea is similar to [36]. Whereas the main concern of [36] is the dot displacement, our focus is the dot profile. Fig. 3.1 (a) shows a digital test patch. It contains a binary bitmap where only every 7-th column and 7-th row contains 1's. There are 15 such patches on the test page.

The test page is first printed at 1200 dpi by a prototype pagewide printhead manufactured by HP Inc.. Then, the printout is captured with a QEA PIAS-II camera at 7663.4 dpi. Each test patch fits entirely in the field of view of the camera, so 15 captures are needed for a printed page. Each printer pixel corresponds to $\frac{7663.4}{1200} \times \frac{7663.4}{1200} \approx 6.39 \times 6.39$ camera pixels. To facilitate the subsequent analysis, we convert this ratio to an integer, i.e. $6 \times 6$. We exploit the approach presented in [39] to transform the camera pixels into ultra high definition (UHD) pixels of size $\frac{1}{7200} \times \frac{1}{7200}$ in$^2$ via

$$c[\mathrm{i}, \mathrm{j}] = \sum_{[\mathrm{i},\mathrm{j}] \in \Omega_{\mathrm{i},\mathrm{j}}} \alpha_{\mathrm{i},\mathrm{j}}[\mathrm{i}, \mathrm{j}] c[\mathrm{i}, \mathrm{j}], \tag{3.1}$$

where $c[\mathrm{i}, \mathrm{j}]$ is the absorptance of the estimated UHD pixel, $\Omega_{\mathrm{i},\mathrm{j}}$ denotes a set of camera pixels $c[\mathrm{i}, \mathrm{j}]$ intersecting with the UHD pixel $c[\mathrm{i}, \mathrm{j}]$, and $\alpha_{\mathrm{i},\mathrm{j}}[\mathrm{i}, \mathrm{j}]$ is the intersection ratio. The validity of this transformation lies with the assumption that by definition, the UHD lattice is aligned with the printer lattice, and the camera lattice is not rotated or skewed with respect to the printer lattice. Horizontal or vertical displacement of the camera lattice with respect to the printer lattice can be estimated from the L-shaped patterns located on both sides of the bottom of the test patch shown in Fig. 3.1. Next, the segmentation mask is computed according to Otsu's method [40]. Shown in Figs. 3.1 (b) and (c) are the transformed image of the test patch and its corresponding segmentation mask. Finally, the sample mean and standard deviation profiles of all the 3300 dots on the printed test page are computed. The contour plots are presented in Fig. 3.2.

As can be seen from Fig. 3.2 (a), the mean dot is limited to a $5 \times 3$ printer-pixel region, being slightly elongated in the vertical direction. Besides, the shade is relatively dark in the

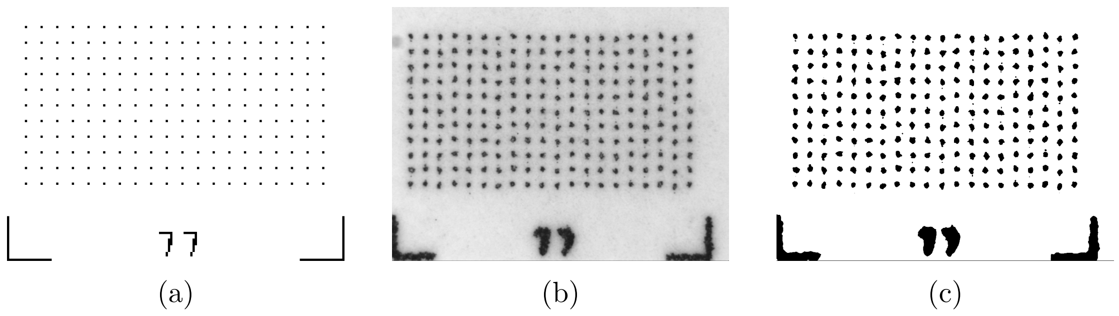


**Figure 3.1.** (a) A digital test patch. (b) The printed test patch captured at 7336.4 dpi and converted to 7200 dpi. (c) The binary segmentation mask of the test patch.

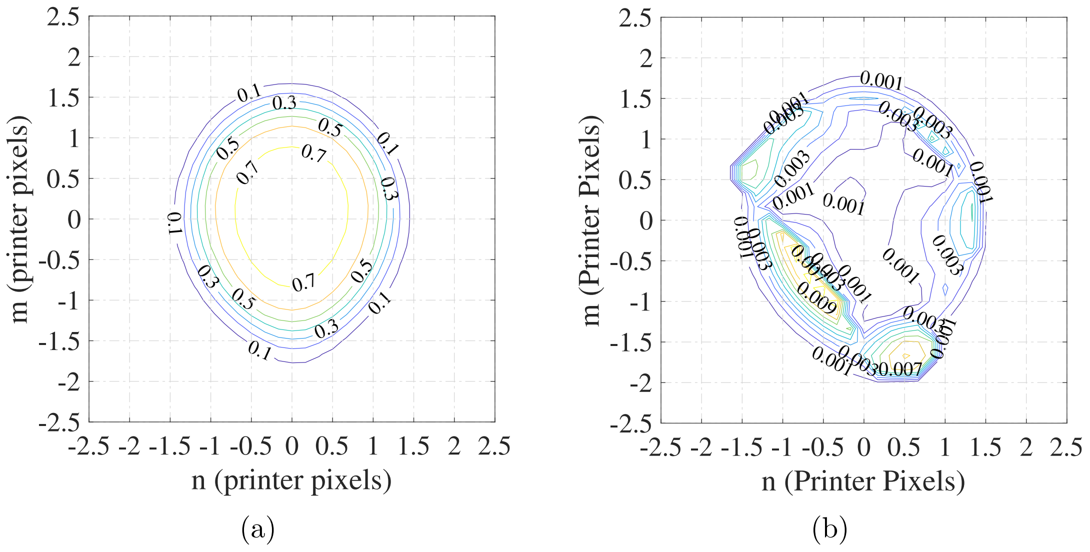


**Figure 3.2.** (a) The mean dot profile and (b) standard deviation profile in the units of absorptance.

center and becomes lighter towards the edge. Looking at Fig. 3.2 (b), we note that the standard deviation is the greatest around the shoulder of the dot, i.e. where the absorptance is changing most rapidly from 0.7 to 0.1 as we go from the center of the dot to the periphery. Nevertheless, the maximum standard deviation is just 0.009. Therefore, it is reasonable to assume that the dots produced by the print bar are constant in shape, and equal to the mean profile.

### 3.2.3 Overview of DBS

DBS is a search-based algorithm that uses a human visual system (HVS) model [41] $h(x, y)$ to find a halftone image $g(x, y)$ that minimizes the visual error between the perceived halftone image $\tilde{g}(x, y)$ and the perceived continuous-tone image $\tilde{f}(x, y)$. The error metric is

$$E = \int_{x,y} \left|\tilde{g}(x, y) - \tilde{f}(x, y)\right|^2 \mathrm{d}x\, \mathrm{d}y. \tag{3.2}$$

To achieve the local minimum, DBS iteratively toggles a pixel or swaps it with a neighboring pixel that has a different value.

Suppose there exists an ideal printer whose dot profile $p(x, y) = \mathrm{rect}(\frac{x}{X}, \frac{y}{X})$. Let $g[m, n]$ denote the digital halftone image. Then, the perceived halftone rendered by the printer is

$$\tilde{g}(x, y) = \sum_{m,n} g[m, n] \cdot \tilde{p}(x - mX, y - nX), \tag{3.3}$$

where $\tilde{p}(x, y) = h(x, y) * p(x, y) \approx h(x, y)$ [34] denotes the cascade of the dot profile with the HVS. The closed-form expression of $h(x, y)$ can be found in [42]. Similarly, $\tilde{f}(x, y)$ can be represented in terms of $f[m, n]$. Define $\mathrm{e}[m, n] = g[m, n] - f[m, n]$. Then, (4.4) can be written as

$$E = \sum_{m,n} \sum_{k,l} \mathrm{e}[m, n]\mathrm{e}[k, l] c_{\tilde{p}\tilde{p}}[m - k, n - l], \tag{3.4}$$

where $c_{\tilde{p}\tilde{p}}(x, y) = \int_{\xi,\eta} \tilde{p}(\xi, \eta) \cdot \tilde{p}(\xi + x, \eta + y)\, \mathrm{d}\xi\, \mathrm{d}\eta$ is the auto-correlation function of $\tilde{p}(x, y)$, and $c_{\tilde{p}\tilde{p}}[m, n] = c_{\tilde{p}\tilde{p}}(mX, nX)$ is the sampled version of it.

## 3.3 Dot Profile Model

Based on the dot statistics in Sec. 3.2.2, a dot profile model is developed to summarize the characteristics of the target printer. It is assumed that the printer is capable of ejecting consistent ink drops at designated locations. The size, shape, and the absorption uniformity of the drops are the same as those of the mean dot. The effect of dot overlap is modeled as addition with saturation to 1 (full black). Under this model, the rendered halftone image produced by the target printer can be simulated as a function of the mean dot profile and the digital halftone. We build a 7200 dpi simulated printer based upon this model with the UHD mean dot profile.

To embed the printer model in DBS, [34] and [36] create an equivalent gray-scale (EQGS) image. It is computed as

$$\bar{g}_{\mathrm{SD}}[m,n] = \frac{1}{X^2} \int_{\Omega^{\mathrm{SD}}[m,n]} g(x,y)\,\mathrm{d}x\,\mathrm{d}y, \tag{3.5}$$

where $\Omega^{\mathrm{SD}}[m,n]$ is a printer cell of size $\frac{1}{1200} \times \frac{1}{1200}$ in$^2$. We refer to $\bar{g}_{\mathrm{SD}}[m,n]$ as the SD EQGS image. Following the lead of these earlier works, we too use the EQGS image to measure the visual fidelity of the printed halftone to the original image.

Before incorporating the SD EQGS image in DBS, we use the simulated printer to examine its validity. Shown in Fig. 3.3 (a)-(c) are a bitmap, its simulated print, and the SD EQGS image. It can be observed that the SD EQGS image provides a reasonable approximation to dot overlap; but it is not good at depicting the edges and details. Therefore, it is necessary to refine the estimation. We develop an HD EQGS image by partitioning each printer pixel into $3 \times 3$ sub-pixels. The HD EQGS value is then computed within each $\frac{X}{3} \times \frac{X}{3}$ cell. This process raises the resolution of the EQGS image to 3600 dpi. The resulting HD EQGS image is shown in Fig. 3.3 (d).

Along the lines of the tabular approaches [34]–[36], an EQGS LUT will be precomputed. With our target printer, the EQGS value of a pixel can be determined in a $5 \times 3$ region, so there will be $2^{15}$ possible binary patterns. In this case, the approach in [34] is intractable, because one would need to manually capture lots of patterns. This necessitates the use of an analytical printer model. To minimize the measurements, we simulate the printed image of all these patterns using the simulated printer, and then compute the EQGS values digitally. In so doing, the LUT can easily be obtained. The algorithm for generating the LUT is summarized below. It is worth mentioning that each entry of the SD LUT is a single value, whereas that of the HD LUT is a 9-element array.

**Algorithm 1:** Computation of the LUT

```
Initialize the UHD image $\bar{g}_{\mathrm{UHD}} \leftarrow 0$;
Initialize the table LUT $\leftarrow 0$;
for $i \leftarrow 0$ to $2^{15} - 1$ do
    Convert i to a $5 \times 3$ binary bitmap $B$;
    for $b \in B$ do
        if $b$ == 1 then
            Add the UHD dot profile to $\bar{g}_{\mathrm{UHD}}$ on the corresponding location;
    if $\bar{g}_{UHD}[\,:,:\,] > 1$ then
        $\bar{g}_{\mathrm{UHD}}[\,:,:\,] \leftarrow 1$;
    LUT$[\mathrm{i},:\,]$ $\leftarrow$average value of the UHD pixels within the center pixel on the SD or
      HD lattice;
return LUT;
```

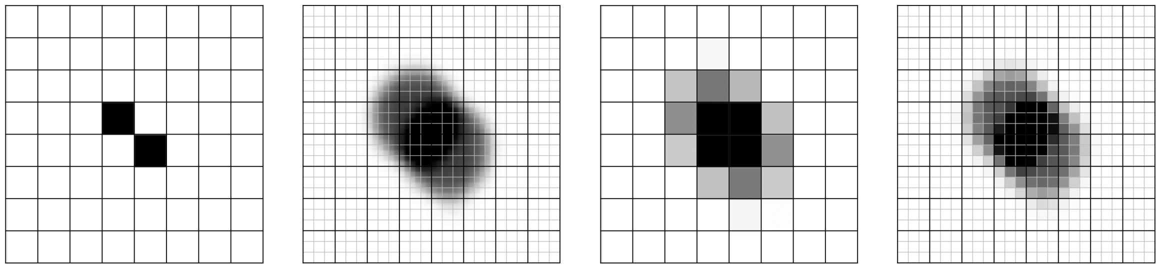

(a) (b) (c) (d)

**Figure 3.3.** Development of the SD and HD EQGS images. The black lines denote the printer lattice, and the gray lines denote the HD lattice. (a) Original digital halftone image sent to the simulated printer. (b) Rendered image at 7200 dpi. (c) SD EQGS image at 1200 dpi. (d) HD EQGS image at 3600 dpi.

## 3.4 Dot Profile Model-Based DBS

In this section, we show a detailed formulation of DBS with the HD dot profile model. The SD case can be easily derived from the HD case; so it is omitted here. Let $S$ denote the upsampling factor. i.e. $S = 1$ for SD and $S = 3$ for HD. The perceived HD EQGS image is given by

$$\bar{g}_{\mathrm{HD}}(x, y) = \sum_{u,v} \bar{g}_{\mathrm{HD}}[u, v] \cdot \tilde{p}_{\mathrm{HD}}\left(x - u\frac{X}{S}, y - v\frac{X}{S}\right). \tag{3.6}$$

Define the discretized and truncated version of the perceived dot profile $\tilde{p}_{\mathrm{HD}}[u, v] = \tilde{p}_{\mathrm{HD}}\left(x - u\frac{X}{S}, y - v\frac{X}{S}\right)$. Then,

$$E = \sum_{u,v}\sum_{s,t} \bar{\mathrm{e}}_{\mathrm{HD}}[u, v]\bar{\mathrm{e}}_{\mathrm{HD}}[s, t]c_{\tilde{p}_{\mathrm{HD}}\tilde{p}_{\mathrm{HD}}}[u - s, v - t], \tag{3.7}$$

where $\bar{\mathrm{e}}_{\mathrm{HD}}[u, v] = \bar{g}_{\mathrm{HD}}[u, v] - f_{\mathrm{HD}}[u, v]$.

Let us consider the effect of changing the state of halftone pixels on the cost value. Let $\mathcal{N}^{\mathrm{HD}}[u, v]$ denote the pixels in the $5 \times 3$ neighborhood of $[u, v]$. Toggling $[m_0, n_0]$ or swapping $[m_0, n_0]$ and $[m_1, n_1]$ in the halftone will affect the EQGS values of pixels in $\mathcal{N}^{\mathrm{HD}}_{\mathrm{accepted}} \triangleq \mathcal{N}^{\mathrm{HD}}[u_0, v_0] \cup \mathcal{N}^{\mathrm{HD}}[u_1, v_1]$. Given the fact that the input image $f_{\mathrm{HD}}[u, v]$ does not change, the new error image will be

$$\bar{\mathrm{e}}_{\mathrm{HD}}[u, v] = \begin{cases} \bar{\mathrm{e}}_{\mathrm{HD}}[u, v] + \Delta\bar{g}_{\mathrm{HD}}[u, v], & \text{for } [u, v] \in \mathcal{N}^{\mathrm{HD}}_{\mathrm{accepted}}, \\ \bar{\mathrm{e}}_{\mathrm{HD}}[u, v], & \text{otherwise.} \end{cases} \tag{3.8}$$

Thus, the new cost is

$$\begin{aligned}E &= \sum_{u,v}\sum_{s,t}\bar{\mathrm{e}}_{\mathrm{HD}}[u,v]\bar{\mathrm{e}}_{\mathrm{HD}}[s,t]c_{\tilde{p}_{\mathrm{HD}}\tilde{p}_{\mathrm{HD}}}[u-s,v-t] \\ &= E + \sum_{[u,v]\in\mathcal{N}^{\mathrm{HD}}_{\mathrm{accepted}}}\ \sum_{[s,t]\in\mathcal{N}^{\mathrm{HD}}_{\mathrm{accepted}}} \Delta\bar{g}_{\mathrm{HD}}[u,v]\Delta\bar{g}_{\mathrm{HD}}[s,t] \\ &\qquad \times c_{\tilde{p}_{\mathrm{HD}}\tilde{p}_{\mathrm{HD}}}[u-s,v-t] \\ &\quad + 2\sum_{[u,v]\in\mathcal{N}^{\mathrm{HD}}_{\mathrm{accepted}}}\ \sum_{s,t}\Delta\bar{g}_{\mathrm{HD}}[u,v]\bar{\mathrm{e}}_{\mathrm{HD}}[s,t] \\ &\qquad \times c_{\tilde{p}_{\mathrm{HD}}\tilde{p}_{\mathrm{HD}}}[u-s,v-t].\end{aligned} \tag{3.9}$$

We further define $c_{\tilde{p}_{\mathrm{HD}}\bar{\mathrm{e}}_{\mathrm{HD}}}[u,v] = \sum_{s,t}\bar{\mathrm{e}}_{\mathrm{HD}}[u,v]c_{\tilde{p}_{\mathrm{HD}}\tilde{p}_{\mathrm{HD}}}[u-s,v-t]$. Therefore, the change in the cost due to a trial toggle/swap can be written as

$$\begin{aligned}\Delta E = &\sum_{[u,v]\in\mathcal{N}^{\mathrm{HD}}_{\mathrm{accepted}}}\ \sum_{[s,t]\in\mathcal{N}^{\mathrm{HD}}_{\mathrm{accepted}}}(\Delta\bar{g}_{\mathrm{HD}}[u,v]\Delta\bar{g}_{\mathrm{HD}}[s,t]) \\ &\qquad \times c_{\tilde{p}_{\mathrm{HD}}\tilde{p}_{\mathrm{HD}}}[u-s,v-t] \\ &+ 2\sum_{[u,v]\in\mathcal{N}^{\mathrm{HD}}_{\mathrm{accepted}}}\Delta\bar{g}_{\mathrm{HD}}[u,v]c_{\tilde{p}_{\mathrm{HD}}\bar{\mathrm{e}}_{\mathrm{HD}}}[u,v].\end{aligned} \tag{3.10}$$

This error metric will guide the search of DBS. Note that $c_{\tilde{p}_{\mathrm{HD}}\tilde{p}_{\mathrm{HD}}}[u,v]$ is independent of the change, but if a change is accepted, $c_{\tilde{p}_{\mathrm{HD}}\bar{\mathrm{e}}_{\mathrm{HD}}}[u,v]$ needs to be updated by

$$\begin{aligned}&c_{\tilde{p}_{\mathrm{HD}}\bar{\mathrm{e}}_{\mathrm{HD}}}[u,v] \\ =&c_{\tilde{p}_{\mathrm{HD}}\bar{\mathrm{e}}_{\mathrm{HD}}}[u,v] + \sum_{[s,t]\in\mathcal{N}^{\mathrm{HD}}_{\mathrm{accepted}}}\Delta\bar{g}_{\mathrm{HD}}[s,t]c_{\tilde{p}_{\mathrm{HD}}\tilde{p}_{\mathrm{HD}}}[u-s,v-t].\end{aligned} \tag{3.11}$$

## 3.5 Experimental Results

In this section, the effect of incorporating the printer models in DBS is evaluated. Tone curves calculated according to [43] are shown in Fig. 3.4. As can be seen, the reproduction curve of DBS with no model, indicated by a red line, is severely distorted. It can be also seen that after embedding the printer models, the bias becomes negligibly small. This suggests

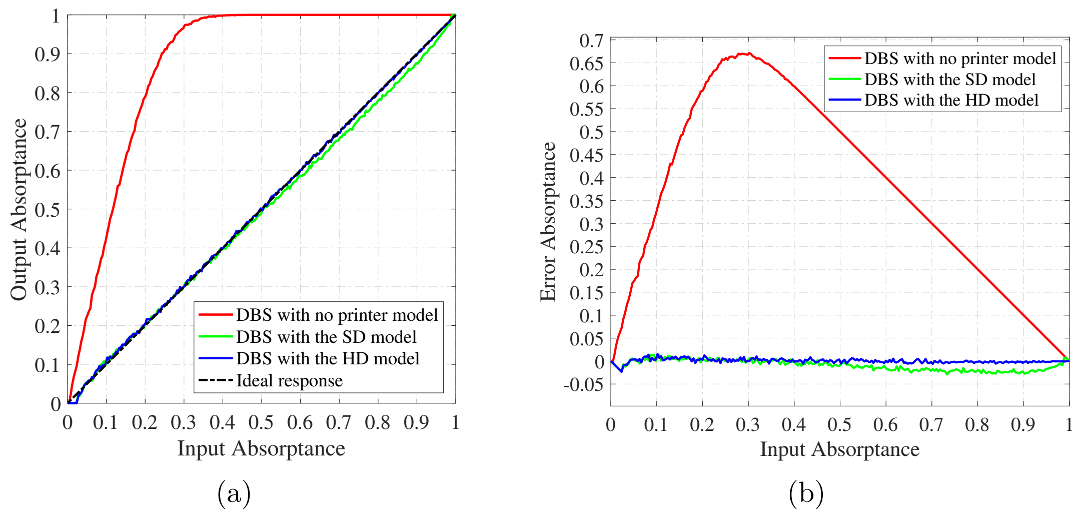


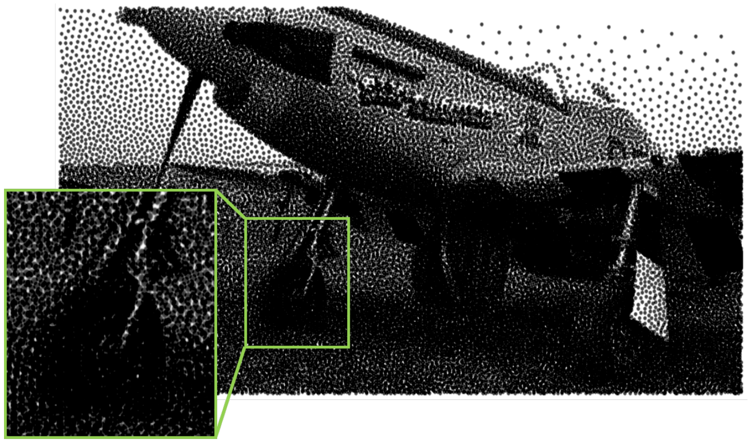
**Figure 3.4.** (a) Tone reproduction curves and (b) tone error curves of DBS with no printer model, DBS with the SD model, and DBS with the HD model. Note that the output absorptance is calculated based on the simulated output of the halftones of constant-tone patches, and the tonal error is obtained by subtracting the input absorptance from the output absorptance.

that both models can produce a correct average tone at most levels. It is noteworthy that the tone reproduction curves for both models contain a small flat region in the very highlights. This will result in a decrease in absorptance that can merely be rectified by applying the inverse mapping of the tone reproduction curve, namely tone correction.

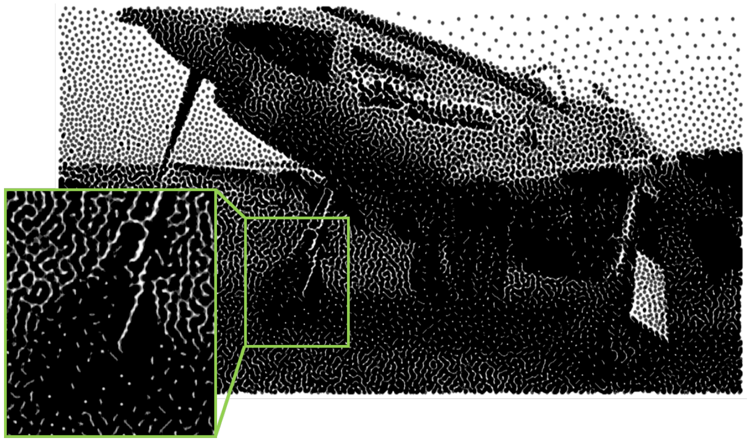
Fig. 3.5 shows the comparison of the simulated prints of a halftone generated by DBS with no model, with the SD model, and with the HD model. To present a fair comparison, tone correction was applied in all three cases before halftoning. These images were all printed at 7200 dpi using the simulated dot profile printer. It can be observed that the graininess in the mid-tone and shadow regions in (a) becomes less noticeable in both (b) and (c). This implies that the irregularly shaped dots were properly controlled by the models. Looking at the highlights, i.e. the sky area, we note that the improvement is less significant. We believe the reason is that there are fewer overlapping dots, and thus tone compensation should suffice to reduce the bias. Comparing (b) and (c), we note that the HD model performs better than the SD model in the shadow regions, i.e. the lower part of the aircraft and the lawn. The halftone structures are less visible and more homogeneous, making the overall appearance more visually pleasing. There is also more detail in the landing gear with the HD model.

## 3.6 Conclusion

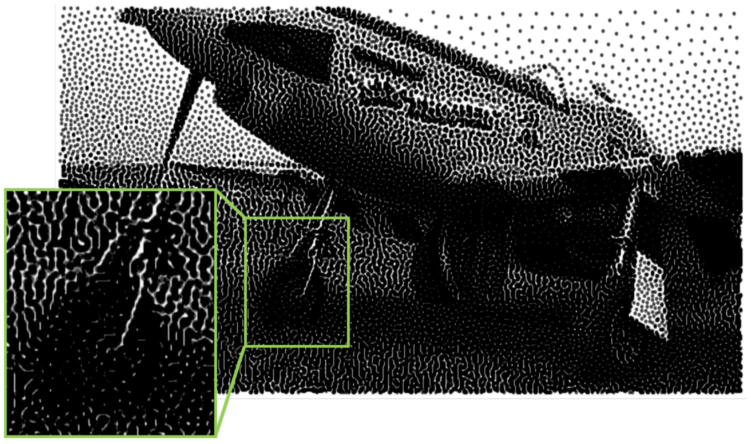
We characterized a printer by measuring and analyzing the dot profiles. Based on the dot statistics, we built a simulated printer to simulate the appearance of the rendered images produced by this printer. The model parameters were computed analytically, so the number of measurements was greatly reduced. We incorporated both the SD and HD printer models in DBS. With the two models, DBS effectively ameliorated the noisiness, and yields nearly linear tone reproduction without tone correction. The HD model outperformed the SD model in the shadows.



(a)



(b)



(c)

**Figure 3.5.** Simulated results printed at 7200 dpi. The halftone images are generated by DBS with (a) no printer model, (b) the SD model, and (c) the HD model. It is recommended to set the zoom level to 150% and to view these images at about 12 inches. Note from the call-outs that the HD model shows the most detail in the landing gear.

# 4. INK DROP DISPLACEMENT MODEL-BASED DIRECT BINARY SEARCH

## 4.1 Introduction

Digital halftoning is the process of transforming a continuous-tone image to a set of binary patterns so that the image can be rendered by bilevel devices such as printers and some displays. Relying on the fact that the human visual system (HVS) acts as a low pass filter, these binary patterns can be perceived as gray levels by the human eye.

According to the computational complexity, halftoning algorithms can be classified into three categories. They are point algorithms (screening or dithering), neighborhood algorithms (error diffusion), and iterative algorithms (least squares and direct binary search (DBS)). Screening involves a pixel-by-pixel comparison with a threshold matrix. Error diffusion consists of a feed-forward loop where the error between the binarization decision and the continuous-tone pixel value is fed to a set of pixels that have not yet binarized. The iterative methods search for the optimal binary image in multiple iterations, where the optimal halftone is the one that minimizes the visual error. We are particularly interested in the iterative methods as they offer the best reproduction with minimal quality degradation [23].

All the aforementioned algorithms can include customized printer models. The most elementary halftoning algorithms assume that printers can precisely place square pixels on predetermined locations. Unfortunately, this assumption does not hold for most real-world printers. Without an accurate model summarizing the non-ideal characteristics of the printer, there will be plenty of uncertainties in the halftone reproduction. These uncertainties will result in image artifacts and print-to-print image quality instability. To address these issues, researchers developed a number of printer models to correct for a variety of system errors in the printing process. Ultimately, these models will be embedded in the elementary algorithms in order to produce high-quality prints.

Printer models can be categorized into process characterization models or print characterization models according to the approach taken to parameterize the model. The process characterization models are derived from a physical understanding of the printing mechanism, and the print characterization models are obtained from measurements and analysis of the printed pages. The earliest work on the process characterization was presented by Ruckdeschel and Hauser [44]. They analyzed the Yule-Nielsen effect [45] by empirically modeling the nonlinear relationship between halftone absorptance and individual colorant absorptance. Loce et al. [46] examined the halftone banding induced by the vibratory motion of the photoconductor in an electrophotographic laser printer. Kacker and Allebach [47] developed an analytical characterization of the electrophotographic (EP) process.

Although fewer measurements are needed, analyzing the physical printing process is complex. Print characterization models are generally more popular. These include the widely-used hard circular dot (HCD) model [24], which assumes that the laser printer ejects circularly shaped black dots. Allebach [27] modified the HCD model to take into account of the effect of spot overlap. Stucki [28] extended error diffusion with an HCD model. Pappas et al. [25][26] popularized the HCD model to a parametric approach. Baqai and Allebach [34] embedded the HCD model into DBS.

The HCD model works well for many printers; but the circular dot assumptions are limiting for some printers. To meet the needs of these printers, Pappas et al. [35] proposed a tabular approach for modeling printers. They used the macroscopic measurements of a set of printed test patches to solve a constrained optimization problem to obtain the table parameters. Baqai and Allebach [34] simplified the approach by using a high resolution drum scanner to measure the absorptance of the center pixel for all possible binary patterns in the $3 \times 3$ neighborhood.

The basis of the above print characterization models is the relationship between the gray level of a pixel in the resulting print and the values of its surrounding halftone pixels. This relationship can be represented a formula, as in the HCD model or by a gray-level lookup table (LUT) compiled from the measurements in a small neighborhood, as in the tabular models. In fact, these models all implicitly rely on an essential tool, which is the equivalent gray-scale (EQGS) image. The EQGS image is developed to summarize the microstructure of the colorant within a printer pixel by the average absorptance over that pixel. It assumes that the human eye cannot resolve the halftone microstructure, and only perceives it as an average. Algorithms, such as [34] [36], compare the EQGS image with the input continuous-tone image to measure the reproduction fidelity of DBS.

Despite its success in creating high quality printed images, the EQGS tabular model has two fundamental limitations. One is the complexity of the memory requirements. The size of the EQGS LUT grows dramatically as the neighborhood expands. This is especially troublesome for some inkjet printers that feature an extra-wide printhead. In the inkjet world, nozzles play an important role in controlling the weight, speed, and trajectory of the ink drops. While every nozzle serves this purpose, not all nozzles can function in the same way. Taking into account the variations across the nozzles would greatly increase storage requirement. For example, with a $5 \times 5$ neighborhood and 1000 nozzles, the EQGS LUT will have $2^{25} \times 1000 = 3.4 \cdot 10^{10}$ entries. The other limitation is the computation time. Tabular models predict the EQGS value of each target pixel by converting the bit pattern in its neighborhood to an index, and then retrieving the corresponding entry from the LUT. Although effective, function calls and memory accesses can still increase the processing time relative to what would be consumed by straight formula computation. For iterative algorithms like DBS, the computation is already heavy. Adding a number of lookups in every iteration would further slow the program down. This makes the EQGS LUT model an infeasible solution to real-time applications of DBS, i.e. direct halftoning of images.

To overcome these limitations, we propose a novel printer model that creates the predicted print by directly modifying pixel locations in the halftone image, while preserving gray values. This model is independent of any neighborhood and the EQGS concept, hence it eliminates the need for an EQGS LUT. We refer to it as the ink drop displacement (IDD) model.

Specifically, we use a set of specially designed test-page geometries to measure dot displacements produced by individual nozzles. By moving each ink drop from the intended position to its actual new position by means of the HVS impulse response, the predicted print can be generated right away. The amount of the movement is determined by the nozzle through which the ink is ejected. Then, the continuous-tone original is subtracted from the resulting predicted image to construct a visual fidelity metric that guides the search of DBS. The change in this fidelity metric due to a trial change in the halftone image, can be easily computed with just a few basic arithmetic operations. With the proposed model, one can effectively improve the printed halftone quality with little increase in computation and memory cost.

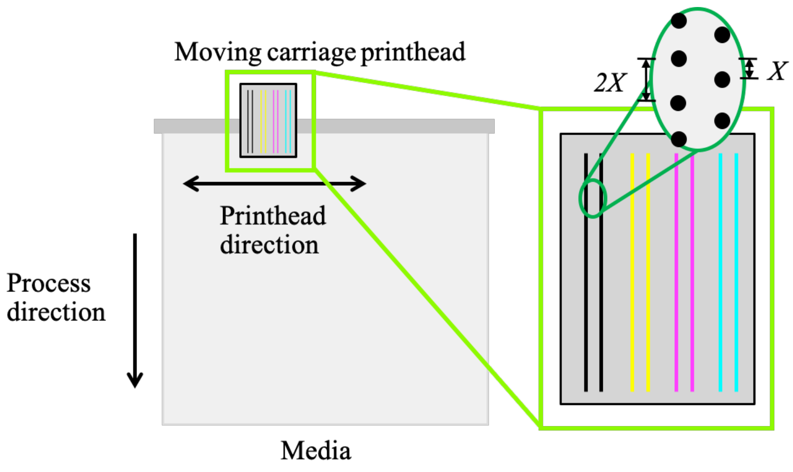
The remainder of this chapter is organized as follows. In Sec. 4.2, we describe the printing process of inkjet printers followed by test page design and printed page analysis. In Sec. 4.3, we briefly review the DBS algorithm and derive an error metric for IDD model-based DBS. We also compare the computation and memory costs for the proposed approach with that of the existing EQGS approach. Sec. 4.4 presents the experiment procedure and the results. In Sec 4.5 we design psychophysical tests to quantify the performance of the two printer models. Finally, conclusions are presented in Sec. 4.6.

## 4.2 Printer Modeling

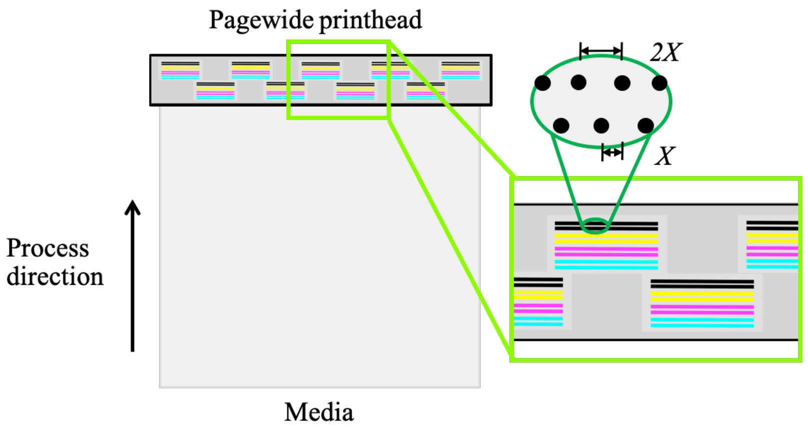
Even though most of the printer modeling work in the literature has been focussed on EP laser printers, inkjet technologies can generate artifacts that limit print quality, as well. We will give an introduction to the printing process of inkjet printers, and then talk about the steps for modeling these printers.

### 4.2.1 Background

Fig. 4.1 (a) shows the architecture of a typical moving carriage inkjet printer. Inkjet devices work by launching tiny ink drops onto the media surface. This can be accomplished by heating the ink, causing ejection of the drop (thermal inkjet), or by applying pressure to force a drop to be ejected by the nozzle (piezo-electric inkjet). For a moving carriage printer, each color ink is typically printed by two staggered columns of nozzles, as shown in Fig. 4.1 (a). Within either column, the nozzles are separated by twice the printer resolution $2X$. The two columns are shifted relative to each other by the printer resolution $X$. This arrangement increases the distance between adjacent nozzle holes, and thereby strengthens the nozzle plate. By using both columns and coordinating the movement of the printhead in the horizontal direction with the timing of the firing of the nozzles, we can achieve a resolution of $X$ in both the vertical (process) direction and the horizontal (scan) direction.



(a)



(b)

**Figure 4.1.** Illustration of two different inkjet printer architectures: (a) conventional moving carriage inkjet printer and (b) pagewide printhead inkjet printer. $X$ denotes the printer resolution.

Moving carriage printers can support a number of different *print modes*. First, we have the option to only print when the printhead is moving from left-to-right, and not during the retrace movement from right-to-left (uni-directional printing), or we can print when the cartridge is moving in both directions (bi-directional printing). Next, we have the option, between each pass of the cartridge over the media, to advance the media by the height of the cartridge (1-pass printing), or by the height of the cartridge divided by an integer $K$ ($K$-pass printing). Here it is assumed that the columns of nozzles span the height of the printhead. The advantage of $K$-pass printing is that we can print any given printer-addressable pixel during any one of $K$ different passes, using one of $K$ different nozzles. This enables us to minimize the effect of any missing nozzles on the resulting print quality. However, it increases the time to print a page by a factor of $K$. Printer drivers typically offer the option of 1- to 6-pass printing, where the higher number of passes are chosen when better image quality is desired. The *print mask*, which is a 2-D array of integers that is periodically tiled over the image to be printed, determines the pass during which any given printer-addressable pixel is printed. Design of the print mask is a challenging problem in its own right [48]–[51].

With pagewide printhead inkjet printers, the situation is quite different, as is shown in Fig. 4.1 (b). Here the printhead is replaced by a print bar that consists of multiple overlapping dies arranged in the scan direction along the length of the print bar [52]. The print bar is referred to as a *pagewide printhead* [52]. Each die consists of four pairs of rows of nozzles corresponding to the four different colorants CMYK, as shown in Fig. 4.1 (b). Where the dies do not overlap, a single nozzle is used to print an entire column of dots of a given color. Where the dies overlap, there is a choice of two different nozzles to print each dot of a given color within a given column of printer-addressable pixels. This flexibility is used to maintain consistency in the scan direction across the boundary between adjacent dies [52]. With a pagewide printhead inkjet printer, the only motion during printing is that of the media under the stationary printhead. So there is no possibility of multipass printing. And no print mask can be used. Thus, each column of pixels in the print will have been printed by the same nozzle or pair of nozzles where the printhead dies overlap. With respect to the contribution in this chapter, one must then rely on model-based halftoning to ameliorate the effects of non-ideal printer behavior.

Due to the complexity of the printing process, there are many uncertainties that can negatively impact the print quality. For inkjet printers, the source of print defects can be typically traced to drop shape irregularity and drop misplacement. Unstable ejection speed and inconsistent ink volumes lead to drop shape errors. Mispositioned nozzles and uncontrolled paper movement translate to drop placement errors. Other factors that influence print quality include air turbulence, ink and substrate properties, and printhead-to-paper spacing. Ultimately, it is the nozzles that transfer the ink from the printhead to paper. Therefore, to

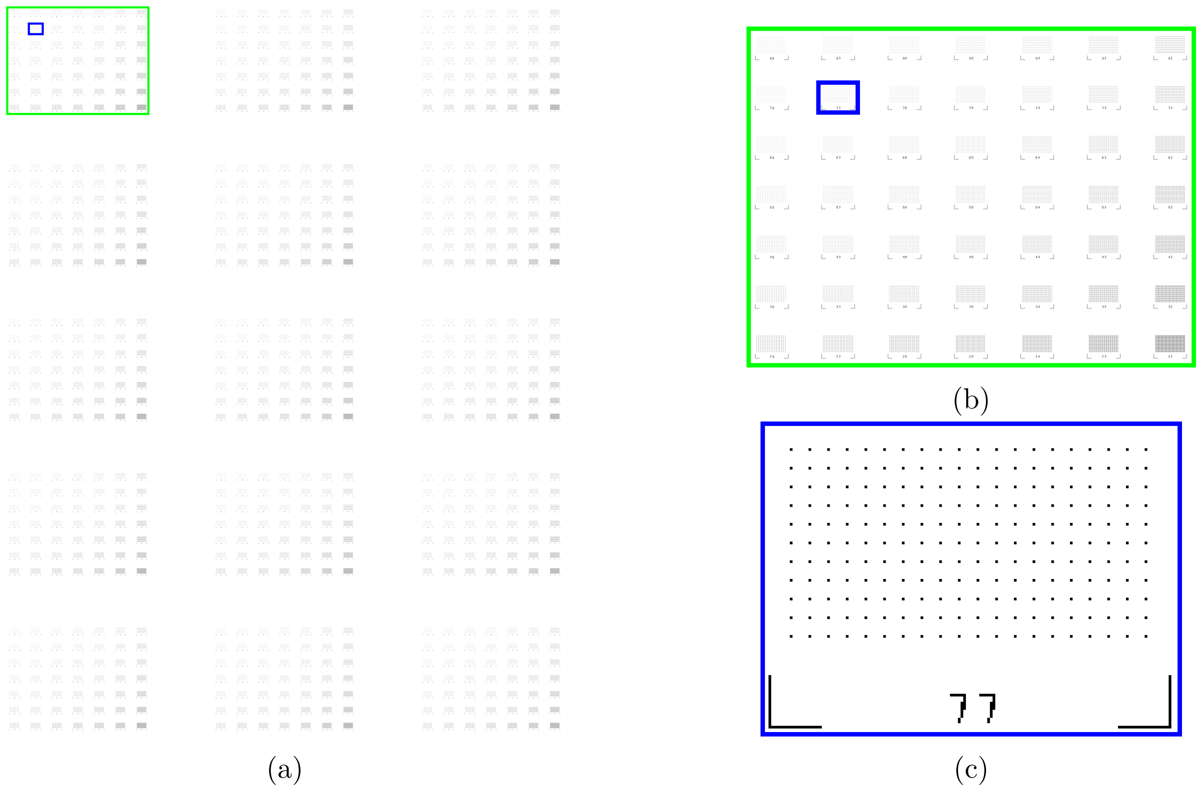


**Figure 4.2.** (a) The test page, consisting of a $5 \times 3$ array of identical blocks; (b) A sample block, consisting of a $7 \times 7$ array of non-identical cells; (c) A sample cell, consisting of a $10 \times 17$ array of individual printer dots.

accurately control the final printout, nozzles that fail to perform within specification need to be detected and carefully modeled.

In this chapter, we are primarily interested in modeling pagewide inkjet devices, as there is literally no existing model that is oriented to such printers. We use a prototype pagewide printhead manufactured by HP Inc. with 1200 dpi resolution as our test vehicle. Nevertheless, the discussion and methods are intended to be adaptable to any inkjet printer that suffers from dot displacement errors.

### 4.2.2 Print Characterization

A common practice for characterizing a printer is to use image analysis tools to analyze a printed diagnostic test page [34] [36]. The test page that we designed for this purpose is shown in Fig. 4.2 (a). We define the following terms. Dot rows are arranged in the

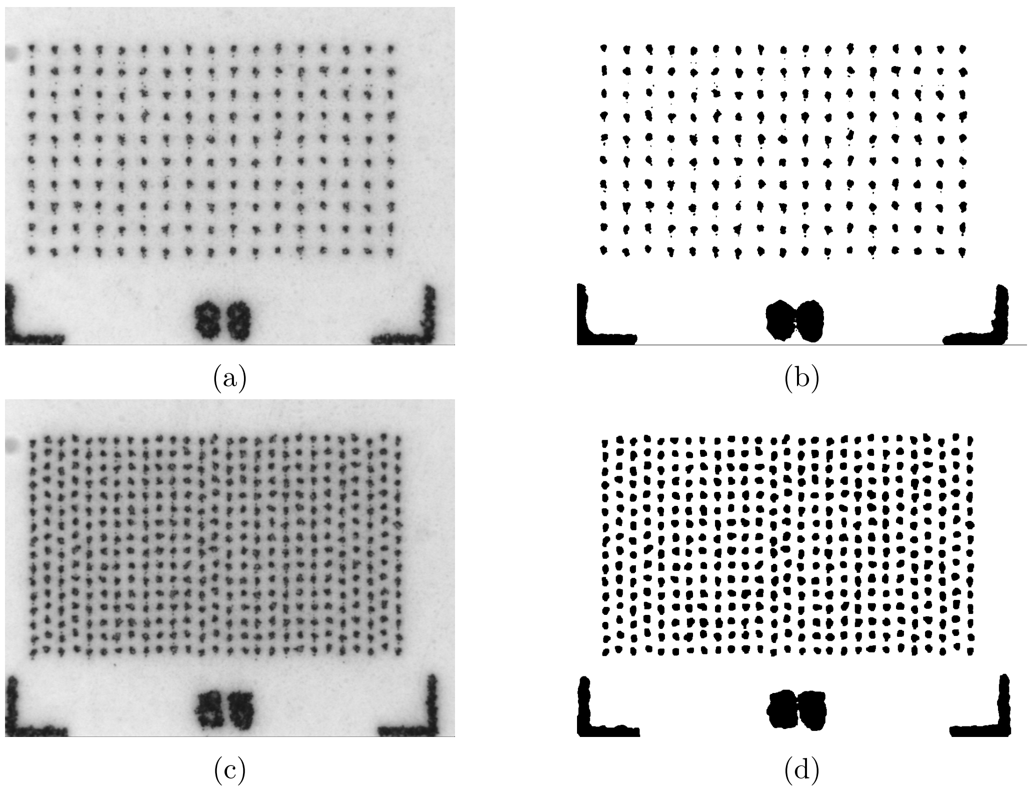


**Figure 4.3.** (a) Captured image and (b) segmentation mask of Cell 88. (c) Captured image and (d) segmentation mask of Cell 55.

vertical direction. Dot columns are arranged across the page, and are associated with nozzle positions on the pagewide printhead. We refer to the geometrical structures in Figs. 4.2 (b) and (c) as a block and a cell, respectively.

The blocks are placed at different locations to provide information about different nozzles. Each block consists of a $7 \times 7$ array of cells. A cell is composed of three main parts. The bottom part is a label $AB$ indicating the cell type, where $A \in [2, 8]$ represents the row spacing between dots and $B \in [2, 8]$ represents the column spacing between dots. The center part of cell $AB$ is a bitmap where only every $A$-th column and $B$-th row contains 1's (black pixels). The lower left and right corners have two L-shaped brackets that are used to help align the camera when capturing an image of the printed cell. All types of cells are included in each block.

Now that we have a test page, the next step is to print and analyze it. The test page is first printed with the target pagewide printhead using black ink. Each printed cell is captured as a separate $768 \times 1024$ pixel image at 7663.4 dpi resolution with a QEA PIAS-II camera. Each captured image is then binarized using Otsu's method [53] to obtain segmentation masks for dot clusters. Shown in Fig. 4.3 are two captured cell images and their corresponding segmentation masks. We note from our experiment that the printhead cannot robustly reproduce isolated pixels when the dot spacing is less than 5 pixels. This phenomena is a consequence of dot gain. For our target pagewide printhead, the relation between nozzle spacing and dot size is intentionally chosen to allow two nozzles that are immediately adjacent on the left and right sides of a missing nozzle to hide that missing nozzle [52]. With that being the case, the gap between adjacent dots is too small for the dots to be effectively separated by Otsu's method. Thus, we will discard these cells. The remaining 80 cells on the test page form 250 dot rows and 261 dot columns.

We perform gray balancing, as described in [54], to obtain the luminance (CIE Y) of the QEA camera pixels. The gray value is then computed by normalizing the luminance to the range of 0 to 1. Finally, this value is subtracted from 1 to convert to units of absorptance. The row and column coordinates of the centroid of the i-th dot are calculated according to

$$\begin{aligned} C_{row,\mathrm{i}} &= \frac{\sum_{[m,n]\in\Omega_\mathrm{i}} m \cdot q[m,n]}{\sum_{[m,n]\in\Omega_\mathrm{i}} q[m,n]} \\ C_{col,\mathrm{i}} &= \frac{\sum_{[m,n]\in\Omega_\mathrm{i}} n \cdot q[m,n]}{\sum_{[m,n]\in\Omega_\mathrm{i}} q[m,n]}, \end{aligned} \tag{4.1}$$

where $\Omega_\mathrm{i}$ denotes the binary mask of the i-th dot and $q[m,n]$ represents the absoprtance of the camera pixels. Note that the satellites, i.e. tiny spots that are not attached to the main drop, are ignored in our analysis.

After locating the centroids, we measure dot statistics such as profiles and displacements. The dot profiles are calculated according to the spatial distribution of ink within the segmentation mask. The displacements are computed as the perpendicular distance from the centroid to a reference line. This reference line is the vertical/horizontal least squares fit line to all dots in each column/row within a cell. Figure 4.4 shows an example illustrating the estimation of the centroids and their misalignment.

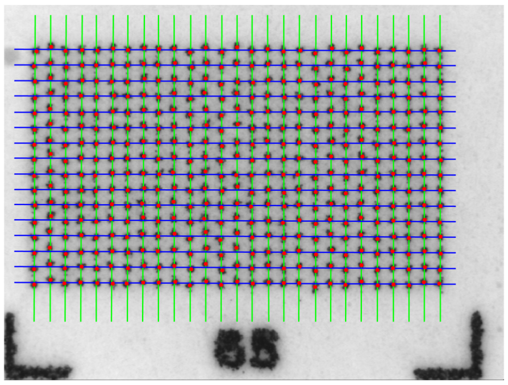


**Figure 4.4.** Centroids and reference lines for an example cell. The red dots denote the actual locations of the ink drop centroids. The green lines denote the vertical reference lines. The blue lines denote the horizontal reference lines.

Figure 4.5 depicts the measurement data from all 65250 dots. As can be seen from Fig. 4.5 (a), the mean dot has a size of roughly $3 \times 2.5$ pixels. We can also see from Fig. 4.5 (b) that the standard deviation is relatively low throughout the entire profile. This implies that the dots produced by the target pagewide printhead have good consistency in shape and absorptance with the mean dot profile. Thus, for simplicity, we will use the mean dot as the printer dot profile in later simulations. From the histograms shown in Figs. 4.5 (c) and (d), it can be seen that both the horizontal and vertical dot displacements are quite random, especially the vertical displacements. It is also clear that there is considerably more vertical than horizontal displacement. These characteristics can be ascertained by visual inspection of Fig. 4.4. Based on these observations and in the interest of simplicity, we choose to incorporate only vertical displacements in our printer model, as discussed later.

To gain more insight on the behavior of the vertical dot displacements, we calculate average statistics within each column, and then examine the distribution of these statistics across all columns, as shown in Figs. 4.6 (a) and (b). We see that the mean vertical displacement for each column varies a lot across all the columns with a mean of 0.00 pixels and a standard deviation of 0.44 pixels. The standard deviation of the vertical displacements is also random with a mean of 0.20 pixels and a standard deviation of 0.02 pixels.

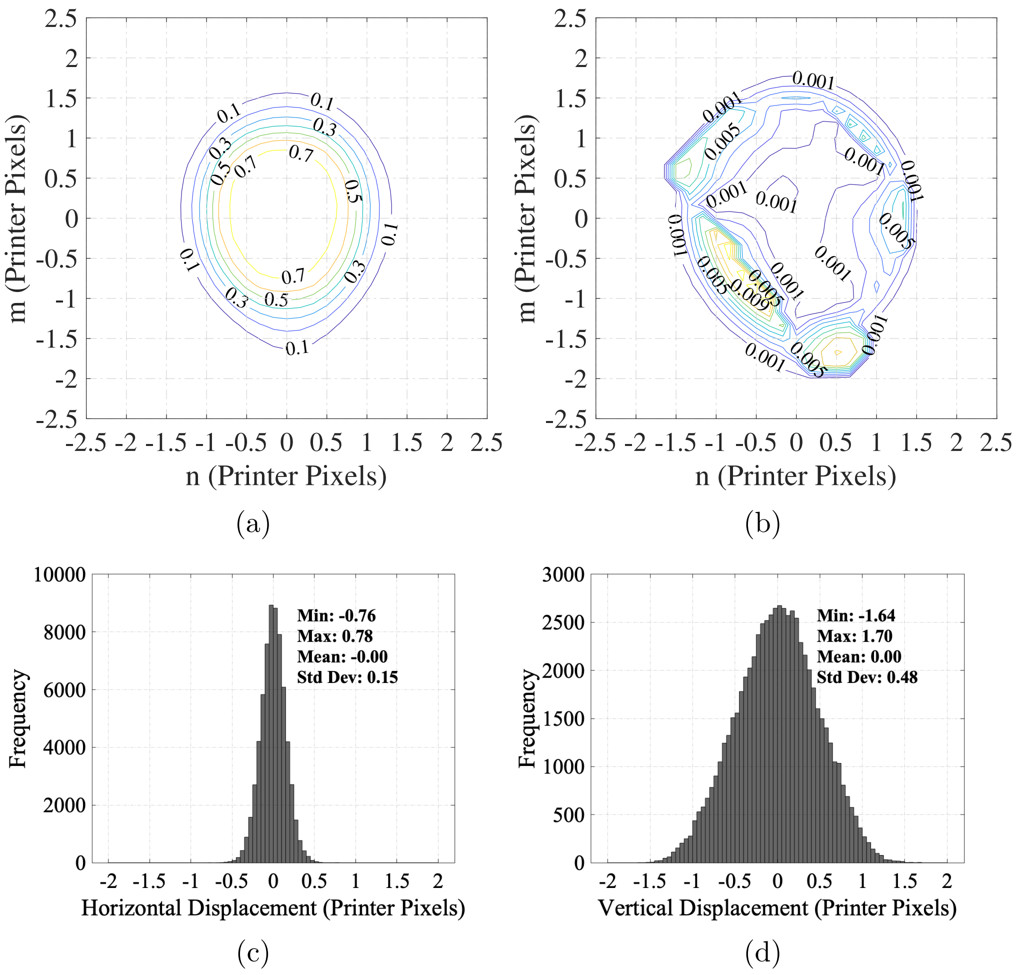


**Figure 4.5.** Measurement data for the pagewide printhead. Dot profiles: (a) Mean dot profile and (b) standard deviation of dot profiles. Histograms of the dot displacements: (c) Horizontal displacement. (d) Vertical displacement.

For each column, i.e. nozzle, it is our goal is to model the vertical displacement of each dot in that column as being drawn from a normal distribution with mean and standard deviation that is in turn drawn from two normal distributions based on Figs. 4.6 (a) and (b). That is, for all the dots in that column, the mean is drawn from a normal distribution with mean 0.00 and standard deviation 0.44 pixels (Fig. 6 (a)), and the standard deviation is drawn from a normal distribution with mean 0.20 and standard deviation 0.02 pixels. To assess the validity of the assumptions of normality, we apply the Anderson-Darling (A-D) test [55] to both the distribution of the statistics of the mean and variance across all nozzles, and to the distribution of the vertical displacements within each column (nozzle). It has been

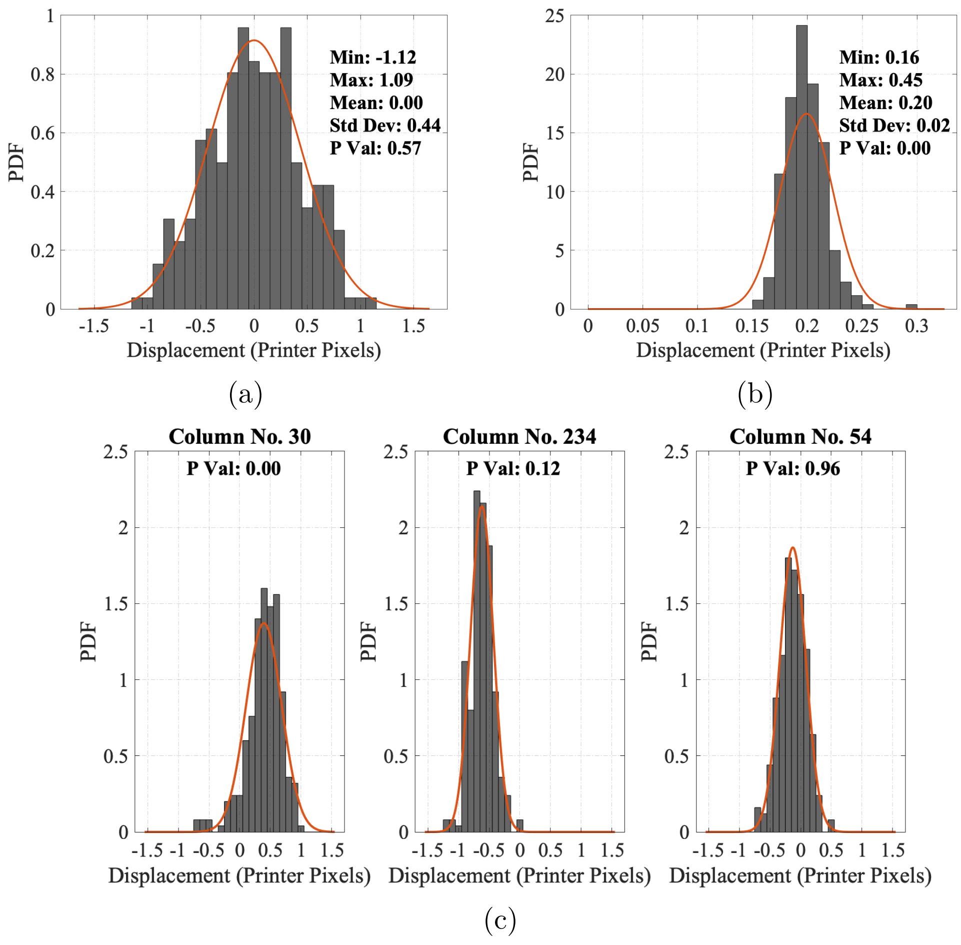


**Figure 4.6.** Distribution across all columns of (a) mean and (b) standard deviation of the vertical displacements, computed as the mean and standard deviation of the individual dot displacements within each column. (c) Distributions for vertical dot displacements in three different nozzles (dot columns). The red line is the probability density function of the normal distribution, i.e. the null hypothesis. The evidence against the null hypothesis decreases as we go from left to right, i.e. the $P$ value increases.

reported in the literature that the A-D test is among the best for assessing normality of an unknown distribution [56]. To compute these tests, we use the *adtest* function in MATLAB.

Here, the null hypothesis is that the underlying distribution is normal. We choose a $P$ value of 0.01 for rejection of the null hypothesis, corresponding to a 99% confidence level. According to this criterion, the assumption of normality for the distribution of the mean displacement for 261 columns (Fig. 4.6 (a)) cannot be rejected. However, the assumption of normality for the distribution of the standard deviation of the displacements for 261 columns (Fig. 4.6 (b)) is rejected. For this population, the null hypothesis (distribution is normal) was rejected for 16% of the nozzles, and was not rejected for 84% of the nozzles.

Figures 4.6 (c), (d), and (e) show the distributions and best fit Gaussian distributions for the lowest $P$ value (null hypothesis rejected), the median $P$ value (null hypothesis not rejected), and the highest $P$ value across the ensemble of 261 nozzles (null hypothesis not rejected). Note that even for the worst case $P = 0.00$, where the null hypothesis is rejected, visually, the displacement distribution does not appear to deviate too much from the best fitting normal distribution. Despite the fact that the A-D test does not completely justify the assumption of a normal distribution for the dot displacement statistics, we proceed with this assumption in the interest of simplicity.

Noting the stochastic nature of the vertical dot displacements, we develop a probabilistic displacement model to capture the statistical properties of the displacements. It will lay the groundwork for an accurate printer model. To be specific, the vertical displacement for the $l$-th column or nozzle at row $k$ is generated by drawing samples from a normal distribution that has fixed mean $\mu_l$ and standard deviation $\sigma_l$ for a given nozzle $l$,

$$d_{k,l} \quad \sim \quad \mathcal{N}(\mu_l, \sigma_l^2). \tag{4.2}$$

Here, the mean and the standard deviation of the nominal displacement for the $l$-th column or nozzle have been generated by drawing samples from two normal distributions that are the same over all the columns or nozzles on the entire page.

$$\mu_l \quad \sim \quad \mathcal{N}(\mu_\mu, \sigma_\mu^2), \quad \sigma_l \quad \sim \quad \mathcal{N}(\mu_\sigma, \sigma_\sigma^2). \tag{4.3}$$

Based on the statistics shown in Figs. 4.6 (a) and (b), we choose $\mu_\mu = 0.00, \sigma_\mu = 0.44, \mu_\sigma = 0.20,$ and $\sigma_\sigma = 0.02$.

To summarize, based on this probabilistic displacement model, we define a stochastic printer model such that

- There only exist vertical dot displacements.

- For each nozzle or column at a given row, the size of the vertical dot displacement is a continuous random variable.
- Vertical dot displacements are identically distributed within any given column, and are mutually independent from row-to-row and from column-to-column.

## 4.3 Printer Model-Based DBS

So far we have shown the steps to catch and parameterize failures in a pagewide inkjet printhead, our next goal is to embed the model in DBS to automatically correct for these failures. Before diving into the details of model-based DBS, an overview of regular DBS will be provided.

### 4.3.1 Overview of DBS

DBS is an iterative algorithm that seeks an optimal halftone $g(x, y)$ by minimizing the total squared error between the perceived halftone $\tilde{g}(x, y)$ and the perceived continuous-tone image $\tilde{f}(x, y)$, based on a visual model and a printer model. In the context of regular DBS, the low-pass characteristic of the HVS is modeled as a two-dimensional, linear, shift-invariant filtering operation. Let $f(x, y)$ denote a continuous-tone image. Then, the perceptually filtered outputs from the HVS filter can be computed as $\tilde{g}(x, y) = g(x, y) * h(x, y)$ and $\tilde{f}(x, y) = f(x, y) * h(x, y)$. Here $*$ indicates convolution and $h(x, y)$ represents the point spread function (PSF) of the HVS filter. Hence, the squared error can be expressed as

$$E = \int_{x,y} \left|\tilde{g}(x, y) - \tilde{f}(x, y)\right|^2 \mathrm{d}x\, \mathrm{d}y. \tag{4.4}$$

Kim and Allebach [57] found Näsänen's HVS model [41] to be good when incorporated in DBS. Its frequency response is given by:

$$\bar{H}(\bar{u}, \bar{v}) = b_0 \Gamma^{b_1} \exp\left(-\frac{\sqrt{\bar{u}^2 + \bar{v}^2}}{b_2 \ln(\Gamma) + b_3}\right), \tag{4.5}$$

where $b_0 = 131.6, b_1 = 0.3188, b_2 = 0.525, b_3 = 3.91$, $\Gamma$ is the average luminance of the light in cd/m$^2$, and $(\bar{u}, \bar{v})$ are the frequency coordinates in cycles/degree subtended at the retina. The impulse response of the HVS, namely the PSF, is given by

$$h(x, y) = b_0 \Gamma^{b_1} \frac{2\pi k}{\Big(k^2 + 4\pi^2(x^2 + y^2)\Big)^{3/2}}, \tag{4.6}$$

where $k = \frac{(\pi V)/180}{b_2 \ln(\Gamma) + b_3}$ [42]. Here $V$ is the viewing distance in inches.

As mentioned in the introduction, an ideal printer model is capable of producing perfect square $X \times X$ dots with each dot aligned to a given position on the printer-addressable lattice. We envision the array of printer-addressable points as a lattice with spacing of $\frac{1}{R}$ inches, where $\frac{1}{R} = X$. For notational convenience, we will switch from $X$ to $\frac{1}{R}$ throughout the rest of this chapter. Let $g[m, n]$ denote a digital halftone image, i.e., the sampled version of $g(x, y)$ rendered by a printer. Then, $g(x, y)$ can be written as a superposition over weighted PSFs, such that

$$g(x, y) = \sum_{m,n} g[m, n] p\Big(x - \frac{m}{R}, y - \frac{n}{R}\Big), \tag{4.7}$$

where $p(x, y) = \mathrm{rect}(xR, yR)$ denotes the PSF of the printing device. The perceived version of $g(x, y)$ is then

$$\tilde{g}(x, y) = \sum_{m,n} g[m, n] \tilde{p}\Big(x - \frac{m}{R}, y - \frac{n}{R}\Big), \tag{4.8}$$

where $\tilde{p}(x, y) = h(x, y) * p(x, y)$ represents the superposition of the printer and visual models. With an ideal high resolution printer, the PSF $p(x, y)$ is so narrow compared to the extent of the HVS PSF that it is valid to make the approximation

$$\tilde{p}(x, y) \approx h(x, y). \tag{4.9}$$

Additionally, $\tilde{f}(x, y)$ and its sampled image $f[m, n]$ are assumed to have the same relationship as that between $\tilde{g}(x, y)$ and $\tilde{g}[m, n]$,which is described in (4.8). Thus, we define the perceived difference as

$$\tilde{\mathrm{e}}(x, y) = \tilde{g}(x, y) - \tilde{f}(x, y). \tag{4.10}$$

The visual fidelity metric in (4.4) can now be written as the energy in the perceived error image in terms of its samples $\mathrm{e}[m,n] = g[m,n] - f[m,n]$, such that:

$$\begin{aligned}\mathcal{E} =& \langle \tilde{\mathrm{e}}(x,y), \tilde{\mathrm{e}}(x,y) \rangle \\ =& \int_{x,y} \left| \sum_{m,n} \mathrm{e}[m,n] \tilde{p}\Big(x - \frac{m}{R}, y - \frac{n}{R}\Big) \right|^2 dxdy \\ =& \sum_{m,n} \sum_{k,l} \mathrm{e}[m,n]\mathrm{e}[k,l] c_{\tilde{p}\tilde{p}}[m-k, n-l],\end{aligned} \tag{4.11}$$

where $c_{\tilde{p}\tilde{p}}[m,n] = c_{\tilde{p}\tilde{p}}(\frac{m}{R}, \frac{n}{R})$, and $c_{\tilde{p}\tilde{p}}(x,y) = \int_\xi \int_\eta \tilde{p}(\xi,\eta)\tilde{p}(\xi + x, \eta + y)\ \mathrm{d}\xi \mathrm{d}\eta$ is the autocorrelation function of the perceived PSF $\tilde{p}(x,y)$.

Starting from an initial estimate of the halftone image, DBS visits the halftone image pixel by pixel from left to right and from top to bottom. For the visit to each pixel, DBS will consider to either switch the binary state of the pixel from 0 to 1 or from 1 to 0 (toggle); or interchange the state of the current pixel with that of any one of its immediate neighbors that differs in state from that of the current pixel (swap). Among all candidate swaps and the toggle, DBS accepts the change, if any, that reduces the cost the most. Each pass through the image is called an iteration. In the end, the algorithm converges to a local minimum of the cost when no changes are kept in an iteration.

### 4.3.2 IDD Model-Based DBS

All printer models aim to purposely distort the halftone image to imitate the printer effects. Furthermore, the printer model should take a form that is easy to be applied in the architecture of the target halftoning algorithm. For example, EQGS models may be implemented within DBS by replacing the $g[m,n]$ term in (4.7) by an EQGS image [34] [36]. Now we will look at a detailed formulation of our IDD model-based DBS.

The ink drops produced by our target printhead, having variations in both shape and location, require an analysis that looks at both aspects. Looking first at the drop shape, we find that the size of the mean dot is much larger than the size of an ideal printer-addressable pixel ($\frac{1}{R} \times \frac{1}{R}$), as shown in Fig. 4.5 (a). So the locally rendered absorptance will be a lot greater than the local ratio of the number of black to the number of white pixels in the halftone image. To correct for this disparity in gray level, a gray-level transformation, namely tone correction, will be applied to the image before halftoning. We will talk in detail later about the steps to estimate such a transformation.

With vertical dot displacements, the visually filtered halftone rendered by the printer becomes

$$\tilde{g}_r(x,y) = \sum_{m,n} g[m,n]\tilde{p}\Big(x - \frac{m + d_{m,n}}{R}, y - \frac{n}{R}\Big), \tag{4.12}$$

where $d_{m,n} = d_{x,y} \cdot R$ pixels. By integrating the displacements into the HVS model, the positional errors can be compensated without any additional assumptions about the absorptance of the pixels, i.e. we do not need to use an EQGS model. And we again are assuming that the dot profile has support that is much less than that of the HVS; so (4.9) still holds. However, it should be noted that the validity of (4.7), (4.8), and (4.12) rests on an assumption of additivity of the printing process. This is not strictly correct. In fact, our simulated printers, to be discussed later, are not based on this assumption. But, we use the assumption here to assure tractability of the analysis of the IDD-DBS halftoning algorithm. The perceived error image under this model is

$$\tilde{\mathrm{e}}_r(x,y) = \tilde{g}_r(x,y) - \tilde{f}(x,y). \tag{4.13}$$

Thus, the cost function is now

$$\begin{aligned}\mathcal{E} &= \Big\langle \tilde{\mathrm{e}}_r(x,y), \tilde{\mathrm{e}}_r(x,y) \Big\rangle \\ &= \Big\langle \tilde{g}_r(x,y), \tilde{g}_r(x,y) \Big\rangle - 2\Big\langle \tilde{g}_r(x,y), \tilde{f}(x,y) \Big\rangle \\ &\quad + \Big\langle \tilde{f}(x,y), \tilde{f}(x,y) \Big\rangle.\end{aligned} \tag{4.14}$$

This cost function will direct the search for changes to the halftone in order to improve the fidelity and quality of the printed image. However, the cost value in (4.14) should not be calculated from scratch for each trial change, because the computation is too extensive to be permitted. In this regard, we present a method that will facilitate the computations. This method is built upon the framework of the efficient DBS presented in [23].

Let us consider the influence of changing the binary state of a halftone pixel on the cost value. Either a toggle at pixel $[m_0, n_0]$ or a swap between pixel $[m_0, n_0]$ and $[m_1, n_1]$ in the digital halftone $g[m, n]$ can be represented as

$$\begin{aligned} g[m,n] = g[m,n] + a_0\delta(m-m_0, n-n_0) \\ + a_1\delta(m-m_1, n-n_1), \end{aligned} \tag{4.15}$$

where $a_0 = \pm 1$, depending on the polarity of the toggle, and $a_1 = 0$ for a toggle and $a_1 = -a_0$ for a swap. Suppose the change is a trial toggle at pixel $[m_0, n_0]$. Substituting $g[m, n]$ into (4.12), we obtain

$$\tilde{g}_r(x,y) = \tilde{g}_r(x,y) + a_0\tilde{p}\Big(x - \frac{m_0 + d_{m_0,n_0}}{R}, y - \frac{n_0}{R}\Big). \tag{4.16}$$

Then, the new cost value after a trial toggle is

$$\begin{aligned} &\mathcal{E}' = \\ &\Big\langle \tilde{g}_r(x,y) + a_0\tilde{p}\Big(x - \frac{m_0 + d_{m_0,n_0}}{R}, y - \frac{n_0}{R}\Big) - \tilde{f}(x,y), \\ &\quad \tilde{g}_r(x,y) + a_0\tilde{p}\Big(x - \frac{m_0 + d_{m_0,n_0}}{R}, y - \frac{n_0}{R}\Big) - \tilde{f}(x,y)\Big\rangle \\ &= \mathcal{E} + 2a_0\Big\langle (\tilde{g}_r(x,y) - \tilde{f}(x,y), \\ &\qquad \tilde{p}\Big(x - \frac{m_0 + d_{m_0,n_0}}{R}, y - \frac{n_0}{R}\Big)\Big\rangle + a_0^2 c_{\tilde{p}\tilde{p}}(0,0). \end{aligned} \tag{4.17}$$

Thus, the influence of toggling the pixel can be computed as the change in the cost value, such that

$$\begin{aligned} \Delta\mathcal{E} &= \mathcal{E}' - \mathcal{E} \\ &= 2a_0 \sum_{m,n} g[m,n] \\ &\qquad \times c_{\tilde{p}\tilde{p}}\Big(\frac{(m-m_0) + (d_{m,n} - d_{m_0,n_0})}{R}, \frac{n-n_0}{R}\Big) \\ &\quad - 2a_0 \sum_{m,n} f[m,n] c_{\tilde{p}\tilde{p}}\Big(\frac{m - m_0 - d_{m_0,n_0}}{R}, \frac{n-n_0}{R}\Big) \\ &\quad + a_0^2 c_{\tilde{p}\tilde{p}}(0,0). \end{aligned} \tag{4.18}$$

As we have seen, due to all the variations in the printing process, dot displacements errors in a real world printer are usually random. Thus, we extend the derivation to the stochastic case by taking the expectation of the cost function with respect to the dot displacement. The new cost function will be

$$\phi = E\{\mathcal{E}\}. \tag{4.19}$$

With this new error metric, DBS minimizes the statistical average of the visually weighted error over the ensemble of all possible dot displacement sets for a given halftone image.

By treating the dot displacement in j-th column as a continuous random variable $D_\mathrm{j}$ with probability density function $\rho_\mathrm{j}$, the change in the cost in (4.18) can be expressed as

$$\begin{aligned}
&\Delta\phi \\
&= 2a_0 \sum_{m,n} g[m,n] \\
&\qquad\qquad \times E\Big\{c_{\tilde{p}\tilde{p}}\Big(\frac{(m-m_0)+(D_n-D_{n_0})}{R}, \frac{n-n_0}{R}\Big)\Big\} \\
&\quad - 2a_0 \sum_{m,n} f[m,n] E\Big\{c_{\tilde{p}\tilde{p}}\Big(\frac{m-m_0-D_{n_0}}{R}, \frac{n-n_0}{R}\Big)\Big\} \\
&\quad + a_0^2 c_{\tilde{p}\tilde{p}}(0,0)
\end{aligned} \tag{4.20}$$

Let $\mathrm{i} = m - m_0$. We define

$$\begin{aligned}
&\bar{\bar{c}}_{\tilde{p}\tilde{p}}[\mathrm{i}; n, n_0] \\
&\equiv E\Big\{c_{\tilde{p}\tilde{p}}\Big(\frac{\mathrm{i}+(D_n-D_{n_0})}{R}, \frac{n-n_0}{R}\Big)\Big\} \\
&= \\
&\begin{cases} \int_{\alpha,\beta} c_{\tilde{p}\tilde{p}}\Big(\frac{\mathrm{i}+(\alpha-\beta)}{R}, \frac{n-n_0}{R}\Big)\rho_{n,n_0}(\alpha,\beta)\,\mathrm{d}\alpha\,\mathrm{d}\beta, & \text{if } n \neq n_0 \\ c_{\tilde{p}\tilde{p}}\Big(\frac{\mathrm{i}}{R}, \frac{0}{R}\Big), & \text{if } n = n_0 \end{cases}
\end{aligned} \tag{4.21}$$

and

$$\begin{aligned}
&\bar{c}_{\tilde{p}\tilde{p}}[\mathrm{i}; n, n_0] \\
&\equiv E\Big\{c_{\tilde{p}\tilde{p}}\Big(\frac{\mathrm{i}-D_{n_0}}{R}, \frac{n-n_0}{R}\Big)\Big\} \\
&= \int_\alpha c_{\tilde{p}\tilde{p}}\Big(\frac{\mathrm{i}-\alpha}{R}, \frac{n-n_0}{R}\Big)\rho_{n_0}(\alpha)\,\mathrm{d}\alpha,
\end{aligned} \tag{4.22}$$

where $\rho_{n,n_0}$ is the joint probability density function of $D_n$ and $D_{n_0}$, and $\rho_{n_0}$ is the univariate probability density function of $D_{n_0}$.

Assuming that $D_n$ and $D_{n_0}$ are independent, their joint density function can be written as

$$\rho_{n,n_0}(\alpha, \beta) = \rho_n(\alpha)\rho_{n_0}(\beta), \qquad \text{for all } n \neq n_0. \tag{4.23}$$

To further simplify the equation, we define

$$\begin{aligned}
\bar{\bar{z}}[k, l] &= \sum_{m,n} g[m, n]\bar{\bar{c}}_{\tilde{p}\tilde{p}}(m-k; n, l), \\
\bar{s}[k, l] &= \sum_{m,n} f[m, n]\bar{c}_{\tilde{p}\tilde{p}}(m-k; n, l).
\end{aligned} \tag{4.24}$$

Finally, when a trial toggle is accepted, the change in the error metric is

$$\Delta\phi = 2a_0(\bar{\bar{z}}[m_0, n_0] - \bar{s}[m_0, n_0]) + a_0^2 c_{\tilde{p}\tilde{p}}(0, 0). \tag{4.25}$$

For a trial swap, $\Delta\phi$ can be similarly derived. It is

$$\begin{aligned}
\Delta\phi = {} & 2a_0(\bar{\bar{z}}[m_0, n_0] - \bar{s}[m_0, n_0]) \\
& + 2a_1(\bar{\bar{z}}[m_1, n_1] - \bar{s}[m_1, n_1]) \\
& + (a_0^2 + a_1^2)c_{\tilde{p}\tilde{p}}(0, 0) + 2a_0 a_1 \bar{\bar{c}}_{\tilde{p}\tilde{p}}(m_0 - m_1; n_0, n_1).
\end{aligned} \tag{4.26}$$

Both $\bar{\bar{z}}[k,l]$ and $\bar{s}[k,l]$ can be precomputed and stored in a LUT. Whereas $\bar{s}[k,l]$ is fixed and independent of the halftone, $\bar{\bar{z}}[k,l]$ should be updated every time when a trial change is accepted according to

$$\bar{\bar{z}}[k,l] = \begin{cases} \bar{\bar{z}}[k,l] \\ \quad + a_0 \bar{\bar{c}}_{\tilde{p}\tilde{p}}(m-m_0;n,n_0) \end{cases}, \quad \text{if accept a toggle,} \\ \begin{cases} \bar{\bar{z}}[k,l] \\ \quad + a_0 \bar{\bar{c}}_{\tilde{p}\tilde{p}}(m-m_0;n,n_0), \\ \quad + a_1 \bar{\bar{c}}_{\tilde{p}\tilde{p}}(m-m_1;n,n_1) \end{cases} \quad \text{if accept a swap.} \tag{4.27}$$

We now have a complete scheme for performing DBS with the new model. Next, we will assess the computation and memory costs of our algorithm and make a comparison to the EQGS tabular model.

### 4.3.3 Computation and Memory Complexity

Low computation and memory complexity are desirable characteristics to all halftoning algorithms. They are especially crucial to iterative methods, such as DBS, where efficiency is necessary for the algorithm to be feasible.

We will use the following notations and assumptions for the analysis in this section.

- The region of support for $\tilde{p}$ is $Q \times Q$. A typical value of $Q$ is 47 pixels, which covers 99% of the HVS filter for a printer with resolution $R = 1200$ dpi and a scale factor $R \cdot V = 3500$.
- The size of image is $M \times N$.
- The size of the neighborhood in the EQGS image that is affected by toggling a pixel is $H \times W$.
- Entries in LUT tables are double precision floating point.

For the IDD model, it is required to store $z$, $s$, $\bar{\bar{c}}_{\tilde{p}\tilde{p}}$, and $\bar{c}_{\tilde{p}\tilde{p}}$, where $z$ and $s$ both have the size of $M \times N$, and $\bar{\bar{c}}_{\tilde{p}\tilde{p}}$ and $\bar{c}_{\tilde{p}\tilde{p}}$ both have the size of $(2Q-1) \times (2Q-1) \times N$. Hence, the IDD model-based DBS consumes

$$S_{\text{IDD}}(Q, M, N) = 2\left[MN + (2Q-1)^2 N\right] \tag{4.28}$$

memory units.

For the EQGS tabular model, $c_{\tilde{p}\tilde{p}}$ and $c_{\tilde{p}\tilde{e}}$ should be stored. The former will be of size $(2Q-1) \times (2Q-1)$, whereas the later is of the same size as the input image. Apart from $c_{\tilde{p}\tilde{p}}$ and $c_{\tilde{p}\tilde{e}}$, an EQGS LUT is needed. Each entry of the LUT consists of the expected value, variance of the EQGS value of the subject pixel, and the pairwise covariance of the EQGS values of the neighboring pixels [36], so the LUT has size $2^{H \times W} \times \left[2 + \binom{H \times W}{2}\right] \times N$. In total, the EQGS model consumes

$$\begin{aligned} S_{\text{EQGS}}(Q, M, N, H, W) &= (2Q-1)^2 + MN \\ &+ 2^{HW}\left[2 + \binom{HW}{2}\right] N \end{aligned} \tag{4.29}$$

memory units.

Let us now calculate the memory cost of the two models when used with the target printhead. As can be seen in Fig. 4.5 (d), each dot may be displaced from roughly -1.6 pixels to 1.7 pixels in the vertical direction. For the EQGS model, changing the state of a single pixel in the digital halftone could affect a $5 \times 1$ window in the EQGS image, i.e., $H = 5$, and $W = 1$. Suppose the image width and height of the input image are both 512 pixels. Then, the memory usage of the EQGS model is $S_{\text{EQGS}}(47, 512, 512, 5, 1) = 0.0037$ GB, and that of the IDD model is $S_{\text{IDD}}(47, 512, 512) = 0.075$ GB. For this particular pagewide printhead, there would not be any advantage in terms of memory efficiency to using the IDD model over the EQGS model, but there might be with another printer that has more substantial dot displacement errors in both the horizontal and vertical directions.

Figure 4.7 compares the amount of memory required by the IDD model and the EQGS model when the neighborhood size ranges from 3 to 20 pixels. Units are measured in GB. It is clear that the memory cost of the IDD remains steady at 0.075, while that of the EQGS model sees a remarkable climb from almost 0 to 825. The 9-pixel neighborhood marks the point at which the memory cost of the EQGS model overtakes that of the IDD model. Thereafter, as the neighborhood broadens to 20 pixels, the EQGS model consumes 2.7 to $1.1 \cdot 10^4$ times more memory than what is consumed by the IDD model.

In brief, from the memory cost point of view, when there are just minor dot displacements (around 1 pixel in both directions), the EQGS model might be a better choice than the IDD model. Otherwise, the IDD model would be the only practical solution.

Next, we turn to the details of computational complexity. Both the IDD and EQGS approaches are built upon the same search framework as that of the efficient DBS. In this framework, the error is computed locally within the support of the HVS filter and only the terms that are related to the trial changes need to be updated [23]. Since evaluating the error metric for trial changes takes the majority of the processing time in each iteration of DBS [23], we will focus on analyzing the complexity of it. In the EQGS approach, toggling a pixel can affect the gray value of all pixels within the $H \times W$ neighborhood around the subject pixel, and swapping will affect two overlapping neighborhoods around the swapped pixels. Thus, it costs $O(H \times W)$ arithmetic operations to compute the EQGS terms in the error metric. By contrast, in the IDD approach, only the subject pixel or the swapped pixels need to be considered, independent of $H$ or $W$. With the use of the IDD model, a dramatic saving in the cost from $O(H \times W)$ to $O(1)$ arithmetic operations can be attained.

Regarding the processing time of the complete halftoning program, we executed DBS with no printer model, DBS with the IDD model, and DBS with the EQGS tabular model on a 2 GHz Dual-Core Intel Core i5 processor. The printer model is the simulated never-centered model to be described in Sec. 4.4, which has $H = 1$ and $W = 3$. For an image of size $512 \times 512$ pixels, efficient DBS took 2.5 - 3 minutes depending on the particular image, the EQGS model-based DBS took 35 - 45 minutes. On the other hand, the IDD model-based DBS took about 3 - 4 minutes, achieving a speedup factor of approximately $11\times$ relative to the EQGS model-based DBS. If DBS is involved indirectly, for instance, in screen design or in tone dependent error diffusion [58], then the computation time is not a

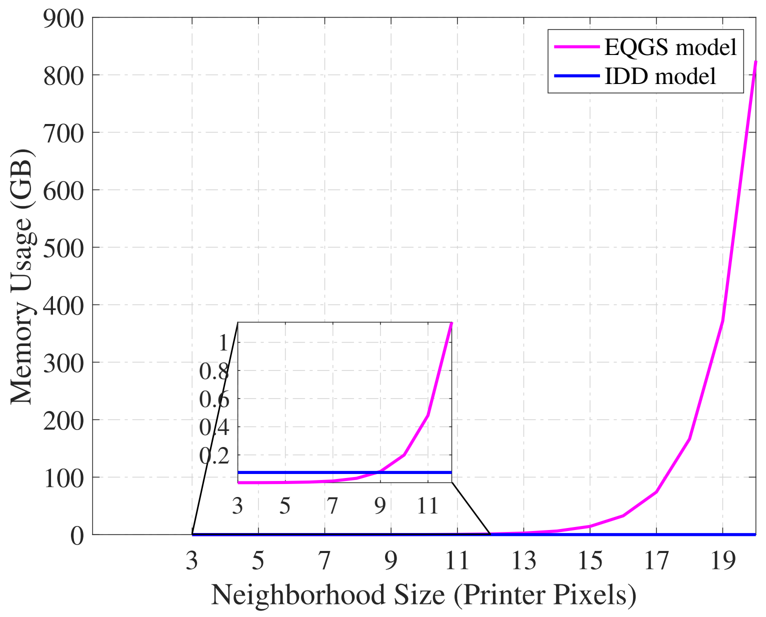


**Figure 4.7.** Memory consumption of the IDD model and the EQGS model. The neighborhood size is equal to $H \times W$ pixels. Note that neighborhood size $= 9$ (i.e. $3 \times 3$) is the cross-over point.

big issue. However, when it is used in direct halftoning applications, the IDD approach will significantly outperform the EQGS approach.

In summary, the IDD model has low space and computation costs due to the fact that it does not rely on an EQGS model. Thus, it is advantageous for use in real-time applications with printers that have significant dot displacement errors.

## 4.4 Experimental Results

To effectively show how well DBS exploits the model, we built two simulated printers. The first one is a stochastic printer that closely matches the target printhead. We use it to examine the effects of integrating the IDD model in DBS. The second one is a deterministic printer. It is created to compare the print quality produced by DBS with no printer model, DBS with the IDD model, and DBS with the EQGS model. The cost function of the deterministic IDD model is still based on (4.18). But the expectation operator is no longer needed, due to the fact that there is no randomness in the deterministic model.

### 4.4.1 Simulated Printers

#### Pagewide Printhead

This synthetic printer is built upon the statistics that we measured from the target prototype pagewide printhead. We assume that the printer has the following characteristics

- Every time the printer processes an image, it will draw a sample function that determines the vertical displacement at each pixel location based on the probabilistic displacement model described in Sec. 4.2. Note that the specific sample function is unknown to the halftoning algorithm.
- The ink dots produced by the printer are constant in shape and equal to the mean dot profile.

#### Never-Centered Printer [36]

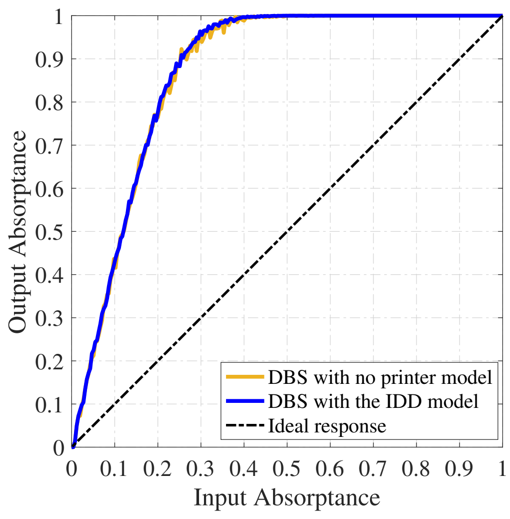
The dots delivered by this printer are misplaced by a fixed amount horizontally. We assume that the printer has the following characteristics

- Dots printed in even-numbered rows are always displaced to the right with a bias of 0.5 pixels, whereas those printed in odd-numbered rows are always displaced to the left by 0.5 pixels.
- The ink dots have an ideal square shape of size $\frac{1}{R} \times \frac{1}{R}$.

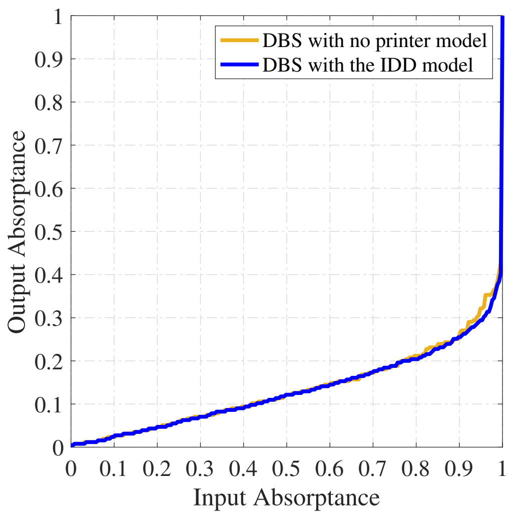
Both printer models are nonlinear in that they incorporate saturation. Absorptance values resulting from dot overlap that are greater than 1 are clipped to 1. This is an approximation to how a real printer would behave.

### 4.4.2 Tone Correction

Our experimental results indicate that tone correction [43] is an effective method to compensate for tone distortions with printers that exhibit significant amounts of dot overlap. Even with an ideal printer, the halftoning algorithm itself may not yield perfect tone reproduction, as is the case with DBS [43]. Tone correction can also compensate for this issue. In



(a)



(b)

**Figure 4.8.** Tone curves of DBS with no printer model and DBS with the IDD model when used with the simulated pagewide printhead. (a) Uncompensated tone reproduction and (b) tone correction curves.

this method, we first create a step wedge image that consists of 256 constant continuous-tone patches spanning all 8-bit gray levels. The step wedge image is then halftoned with the given halftoning algorithm, and printed by the given simulated printer. The mean absorptance of each printed patch is then computed. The mean output absorptances computed from the halftone patches printed by the simulated printer are plotted against the input absorptance to constitute a tone reproduction curve. Then, the inverse mapping is taken to form a tone correction curve. Finally, tone correction can be applied to the continuous-tone image prior to its being halftoned.

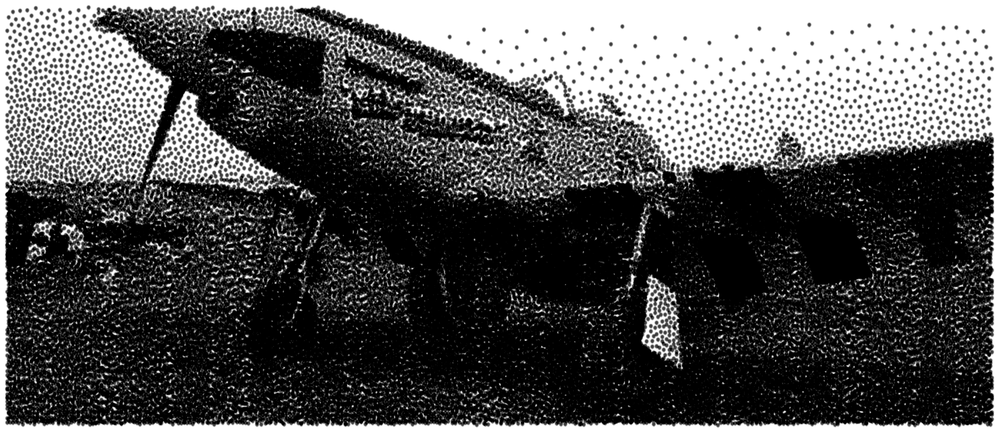
The tone reproduction curves and tone correction curves of the IDD model-based DBS algorithm and DBS with no printer model when used with the simulated pagewide printhead are shown in Fig. 4.8 (a). As is observed, both algorithms tend to darken the absorptance. The ideal curve is an identity function through the origin, as indicated by the dashed line shown in Fig. 4.8 (a). After tone correction, the RMS tone reproduction error in units of absorptance is 0.013 for DBS with no printer model and 0.0094 for DBS with the IDD model. So for this printer model, tone correction is quite effective in both cases. Note that the never-centered printer uses an ideal dot profile. So the only tone reproduction error is due to DBS itself [43], which is relatively small. Thus, we do not perform tone correction with it.

### 4.4.3 Simulated Results

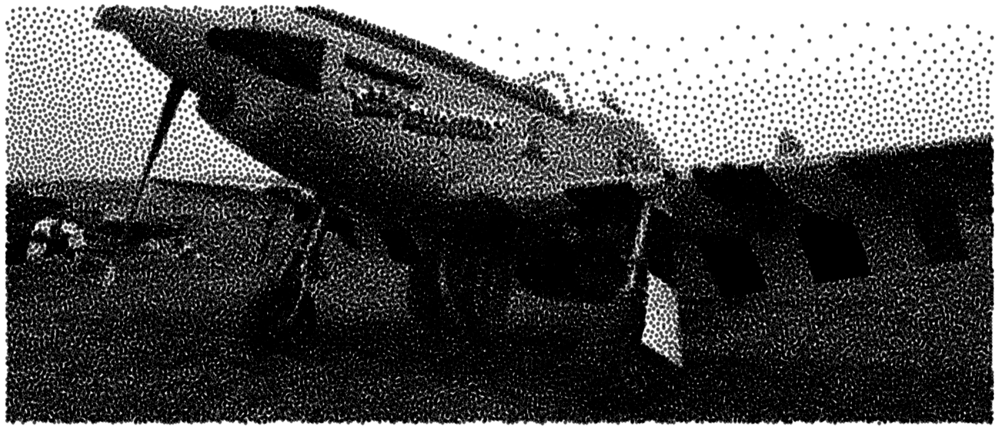
For the simulations of the prototype pagewide printhead, the sample images were first tone corrected and then halftoned at 1200 dpi. The scale factor $R \cdot V$ for Näsänen's HVS model was set to be 3500, since it gives the best visual quality [43], [59]. Next, the halftoned images were rendered with the simulated printer at 12000 dpi, which was realized by replacing each halftone pixel with a displaced $10 \times 10$ pixel cluster. The absorptance profile of each pixel cluster was computed based on the mean dot profile, assuming additive combination of the individual dot profiles with a saturation limit of 1.0, as discussed previously. The amount of displacement was obtained from the probabilistic displacement model. As for printing with the never-centered printer, the procedure was similar, except that the tone correction step was skipped. Besides, each printer pixel in the halftone image corresponds to a $2 \times 2$ block of simulated pixels [36].

Figures 4.9 (a) and (b) show the pagewide printhead simulations of the "mustang" image halftoned by the original DBS and IDD model-based DBS, respectively. These simulated images should be held at a distance of about 20 inches when viewed, assuming that the displayed page is 8.5 in wide. From visual inspection of Fig. 4.9 (a), one sees objectionable white streaks, the result of vertical dot displacements, throughout the mid-tone and shadow areas. Besides, the texture of the image is patchy and noisy. In contrast, no streaks can be seen in Fig. 4.9 (b). The dots are also placed more uniformly, making the overall appearance more visually pleasing. These improvements suggest that the IDD model successfully incorporated dot displacement information for the pagewide printhead in DBS so that high quality prints can be achieved.

Shown in Fig. 4.10 are the simulated prints of the "mustang" image halftoned by DBS with no printer model, DBS with the EQGS model, and DBS with the IDD model for the never-centered printer, all rendered by the never-centered printer. They should also be viewed from a distance of around 20 inches, assuming that the displayed image of the



(a)



(b)

**Figure 4.9.** Simulated pagewide printhead results. Halftone images generated by (a) DBS without printer model and (b) DBS with the IDD model at 1200 dpi for the pagewide printhead, and both rendered with the simulated page printhead at 12000 dpi, with 10× upscaling. The resolution of the continuous-tone input image is 300 × 700 pixels. The resolution of the simulated prints is 3000 × 7000 pixels.

page has width 8.5 in. From visual inspection, it is clear that the image in Fig. 4.10 (a) suffers from veining artifacts near the mid-tones. On the other hand, both the IDD and EQGS model show a remarkable improvement over the artifacts found in the printed image halftoned by the original DBS algorithm. They produce very homogenous dot distributions. In addition, the "SIX SHOOTER" phrase and the image of the Yosemite Sam are rendered more clearly by using the printer models. Our overall conclusion is that the IDD approach is comparable to the EQGS approach in terms of resisting dot displacement distortions.

## 4.5 Psychophysical Image Quality Assessment

The ultimate goal of model-based halftoning is to deliver a satisfying end-user experience. Therefore, a psychophysical image quality assessment experiment is designed as a quantitative metric to judge and compare the performance of the printer models when embedded in DBS.

This experiment consists of two parts. In Part 1, a double-stimulus study was conducted to compare the quality of images that were halftoned using DBS with no printer model and DBS with the IDD model for the pagewide printhead, when rendered by the simulated printer with a pagewide printhead. In Part 2, a tri-stimulus study was designed to compare the quality of images that were haftoned using DBS with no printer model, DBS with the EQGS model, and DBS with the IDD model for the never-centered printer, when rendered by the simulated never-centered printer. Eighteen unique images from Set5 [60] and Set14 [61] (a duplicate is removed), ranging from size $256 \times 256$ to $576 \times 720$ pixels, were used as the test images. The image contents include natural scenes, portraits, animals, and a book cover. These color images were first converted to grayscale using the function *rgb2gray* from the Python toolkit Scikit-image. They were then processed using the above techniques, and the resulting simulated prints were displayed on a computer screen through a graphical user interface (GUI) written in Python.

A total of 26 subjects with normal or corrected to normal vision participated in the experiment. Fifteen of them were experts in the field of halftoning. The other eleven were graduate students with image processing background, but no prior experience with halftoning research. The subjects were asked to rate each simulated print according to its smoothness, texture, and detail rendition at a viewing distance of 20 inches. Nevertheless, they were allowed to make small adjustments to their position if they felt it was necessary. As for the rating scale, we adopted the commonly-used mean opinion score (MOS). This score is set on a five-point scale where 1 means bad and 5 means excellent. To avoid possible ambiguities, we provided detailed descriptions of each of the five scales.

The simulated prints were presented to the subjects in a random order. Half of the images were repeated 1-2 times for the purpose of consistency checking. In total, each subject rated around 50 instances (2 versions of 18 images plus repetitions) in Part 1 and around 80 instances (3 versions of 18 images plus repetitions) in Part 2.

Before analyzing the raw scores, we detected outliers. The j-th subject was considered to be an outlier if the scores that he/she assigned satisfied all of the following criteria:

- $S_{\max}(\mathrm{i},\mathrm{j}) - S_{\min}(\mathrm{i},\mathrm{j}) > 1$, where $S_{\max/\min}(\mathrm{i},\mathrm{j})$ represents the maximum/minimum score assigned to the i-th image by the j-th subject.

- $\sigma\{\bar{S}(\mathrm{i},\mathrm{j})\} > 0.5$, where $\bar{S}(\mathrm{i},\mathrm{j})$ is the mean score over multiple observations of the i-th image by the j-th subject.

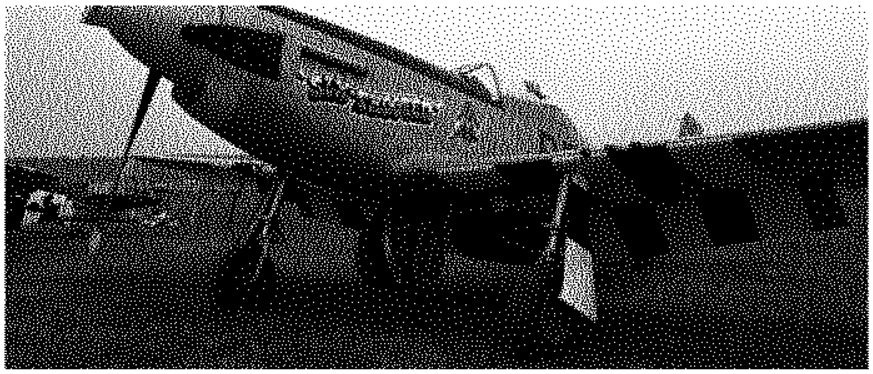
- $\frac{1}{L}\sum_{\mathrm{i}=1}^{L}|\bar{S}(\mathrm{i},\mathrm{j}) - \hat{S}(\mathrm{i})| > 0.4$, where $\hat{S}(\mathrm{i})$ is the mean score of the i-th image over all subjects, and $L = 18$.

Nine subjects in Part 1 were identified as outliers, and eight subjects in Part 2 were identified as outliers. The experimental results with the outliers taken out are shown in Fig. 4.11. It is clear from Figs. 4.11 (a) and (b) that all subjects preferred IDD model-based DBS rather than the original DBS with no printer model when the image is printed by the simulated pagewide printhead. Figure 4.11 (c) shows that the IDD model was rated best (4.0) with EQGS model slightly behind (3.9), while the algorithm with no printer model received the lowest score (2.6). It is revealed from the plots that the incorporation of information about the printing device can significantly improve the performance of DBS, but the difference between the two printer models is not very large. This matches our observations from Fig. 4.10. As is observed from Fig. 4.11 (d), 15 out of 18 subjects thought that the IDD model was no worse than the EQGS model, adding credence to our claim that the IDD model-based DBS produces high quality simulated prints.

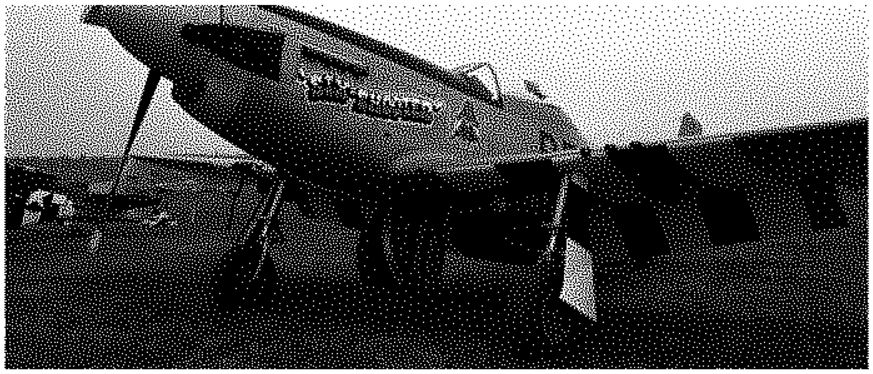
## 4.6 Conclusion

We have developed an IDD printer model within the framework of DBS to deal with dot displacement errors. The dot displacements produced by individual nozzles on a pagewide printhead were characterized and integrated in the printer model. We used tone correction to rectify the gray level distortion caused by the large ink drops. This makes it possible to predict the locally averaged printed halftone absorptance accurately without using an EQGS model. Based on a stochastic model for the dot displacements, we derived a simple formula for IDD model-based DBS to evaluate the visual cost of applying candidate toggles and swaps in a computationally efficient manner. Not only does the proposed model allow DBS to handle dot-positional errors in a large neighborhood, but it also substantially reduces the execution time relative to that required for the EQGS model. An important aspect of our IDD model is that it does not depend on a detailed knowledge of the sample function of the dot displacement errors, but rather only the statistics of those errors. This is particularly important for prints made using a pagewide inkjet printhead, since the alignment between the printed page and the nozzles will be unknown prior to the generation of a halftone version of that printed page.

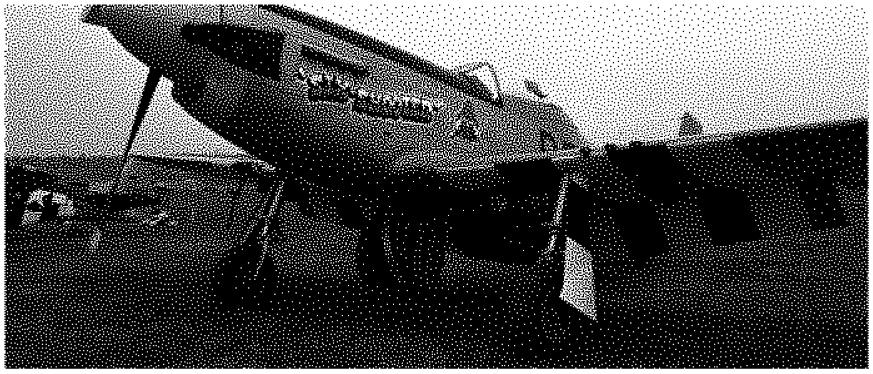
Experimental results show that the IDD model greatly improves the appearance of the simulated print over the original DBS algorithm. In order to quantitatively assess and compare the quality of printed images that were halftoned using DBS with no model, DBS with the IDD model, and DBS with the EQGS model, MOS tests were conducted. The MOS results demonstrated that IDD model-based DBS yields the highest quality and the original DBS with no printer model yields the lowest quality among the three algorithms. Furthermore, the IDD model was ranked slightly more favorably than the EQGS model by the human viewers.



(a)

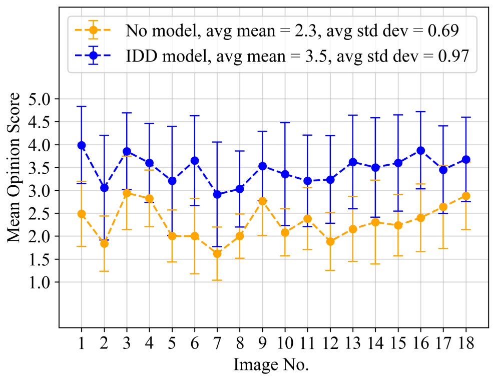


(b)

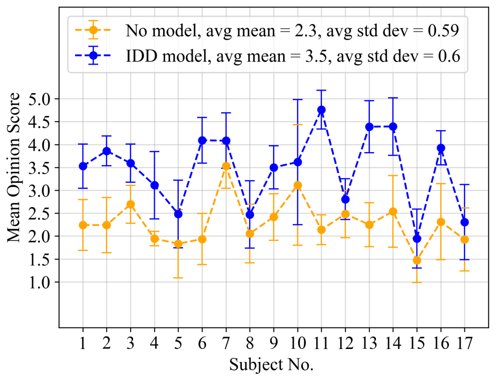


(c)

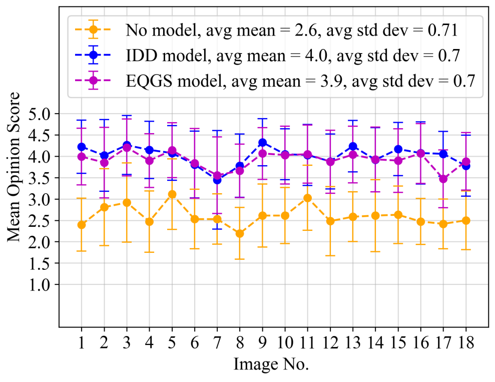
**Figure 4.10.** Simulated never-centered printer results. Halftone images generated by (a) DBS without printer model, (b) DBS with the EQGS model for the never-centered printer, and (c) DBS with the IDD model for the never-centered printer at 1200 dpi, and all rendered with the simulated never-centered printer at 2400 dpi, with $2\times$ upscaling. The resolution of the continuous-tone input image is $300 \times 700$ pixels. The resolution of the simulated prints is $600 \times 1400$ pixels.



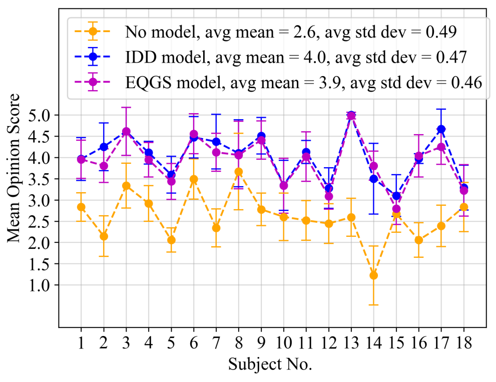
(a)



(b)



(c)



(d)

**Figure 4.11.** Psychophysical experiment results. Part 1: Simulated printer with pagewide printhead. (a) Mean score across images. (b) Mean score across subjects. Part 2: Simulated never-centered printer. (c) Mean score across images. (d) Mean score across subjects. The whiskers indicate the standard deviation associated with each data point.

# 5. HIGHLIGHTED DOCUMENT IMAGE CLASSIFICATION

## 5.1 Introduction

Multifunction printers (MFPs) are popular in home and small office. This is because they offer multiple functions, such as print, copy, and scan, for the price and size of a single device. Apart from the cost and efficiency, the most important factor that the customers care about is image quality. In this regard, different configurations of the scan/copy pipeline are embedded in the device to optimize the image quality of a particular kind of image type, such as text documents, highlighted pages, or photos. For example, the configuration designed for the text mode may increase the contrast and sharpen the edges to get clear text; and the configuration designed for the photo mode may impose a smoothing effect to reduce the noise. A common method to change image quality settings is through manual selection [62]. Users can choose the most appropriate mode from a list of predefined modes according to the content of the document, or adjust each attribute from the submenu. Such a method often requires trained users, which may not always be desirable. Therefore, it is necessary to integrate an automatic document image classification model into the printer firmware.

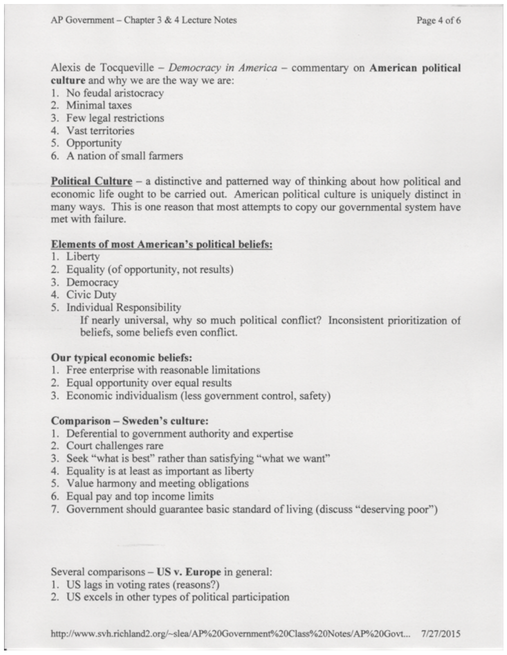
There are many research papers on document image classification [63]–[67]. However, they are not all suited for use in entry-level printers due to the memory and computational complexity restrictions of the printer firmware. To avoid such problems, Lu et al [68] developed a low-complexity algorithm using SVMs and several features to classify text, photo, and mixed documents. Xu et al [69] extended the approach by adding more features and two additional classes highlight and faded document. The features in [69] are

- *Luminance and chroma flatness scores* describing the spread of the histograms.
- *Color variability score* indicating color consistency by measuring the height of the histogram bins.
- *Text edge count* based on counting the number of pixels that differ by 100 in gray value on a scale of 0 to 255 from their adjacent pixels.
- *Chroma around text* indicating the distribution of chroma around text edges.

- *Color block ratio* based on counting the number of $32 \times 32$ non-overlapping blocks with at least 10% chromatic pixels.
- *White block count* based on counting the number of $32 \times 32$ non-overlapping white blocks.

However, [69] is not accurate enough; many highlighted documents are misclassified into the text category and vice versa. Note that highlighter marks on the highlighted documents are made using a highlighter pen after the document has been printed, as shown in Fig. 5.1(c). Misclassifying these highlighted documents is especially problematic for printers. Most highlighter pens have bright and fluorescent colors [70], so they reflect more light than conventional colors. This reflection will result in unreliable color reproduction by the printer. For example, the scanned or copied highlighters may appear lighter or darker than expected or even change colors, i.e. yellow becomes green. The highlighting may even disappear completely when the scanned document is printed. For these reasons, the need still exists for an improved method for differentiating highlighted documents from other types of documents.

In this paper, we propose two novel gamut-based features and six low-level color features to capture the color specifics of the highlighted regions in the image. These new features are concatenated to the seven features in [69]. The sequential forward floating selection (SFFS) feature selection algorithm [71] is applied to find the best feature subset for our application. The optimum set of features is then used to train a directed acyclic graph support vector machine (DAGSVM) [72] to classify the documents. We work with a specific model of MFP and a specific image processing pipeline equipped with four classes of documents. They are text, photo, highlight, and mixed. Some example images are in Fig. 5.1. Nevertheless, our work can be easily applied to any MFP by re-measuring the printer gamut. Our cross-validation results show that the new feature subset significantly improved the precision and recall for all document types.



AP Government – Chapter 3 & 4 Lecture Notes Page 4 of 6

Alexis de Tocqueville – *Democracy in America* – commentary on **American political culture** and why we are the way we are:
1. No feudal aristocracy
2. Minimal taxes
3. Few legal restrictions
4. Vast territories
5. Opportunity
6. A nation of small farmers

**Political Culture** – a distinctive and patterned way of thinking about how political and economic life ought to be carried out. American political culture is uniquely distinct in many ways. This is one reason that most attempts to copy our governmental system have met with failure.

**Elements of most American's political beliefs:**
1. Liberty
2. Equality (of opportunity, not results)
3. Democracy
4. Civic Duty
5. Individual Responsibility
   If nearly universal, why so much political conflict? Inconsistent prioritization of beliefs, some beliefs even conflict.

**Our typical economic beliefs:**
1. Free enterprise with reasonable limitations
2. Equal opportunity over equal results
3. Economic individualism (less government control, safety)

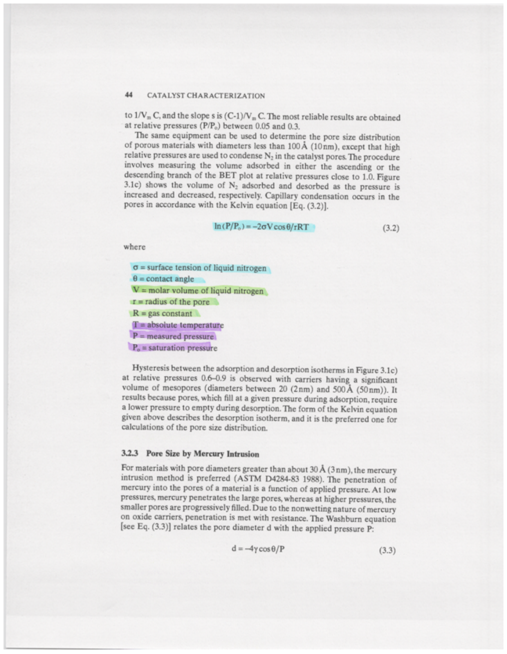
**Comparison – Sweden's culture:**
1. Deferential to government authority and expertise
2. Court challenges rare
3. Seek "what is best" rather than satisfying "what we want"
4. Equality is at least as important as liberty
5. Value harmony and meeting obligations
6. Equal pay and top income limits
7. Government should guarantee basic standard of living (discuss "deserving poor")

Several comparisons – **US v. Europe** in general:
1. US lags in voting rates (reasons?)
2. US excels in other types of political participation

http://www.svh.richland2.org/~slea/AP%20Government%20Class%20Notes/AP%20Govt... 7/27/2015

(a)

(b)



44 CATALYST CHARACTERIZATION

to $1/V_m C$, and the slope s is $(C-1)/V_m C$. The most reliable results are obtained at relative pressures ($P/P_o$) between 0.05 and 0.3.

The same equipment can be used to determine the pore size distribution of porous materials with diameters less than 100 Å (10 nm), except that high relative pressures are used to condense $N_2$ in the catalyst pores. The procedure involves measuring the volume adsorbed in either the ascending or the descending branch of the BET plot at relative pressures close to 1.0. Figure 3.1c) shows the volume of $N_2$ adsorbed and desorbed as the pressure is increased and decreased, respectively. Capillary condensation occurs in the pores in accordance with the Kelvin equation [Eq. (3.2)].

$$\ln(P/P_o) = -2\sigma V \cos\theta / rRT \qquad (3.2)$$

where

$\sigma$ = surface tension of liquid nitrogen
$\theta$ = contact angle
$V$ = molar volume of liquid nitrogen
$r$ = radius of the pore
$R$ = gas constant
$T$ = absolute temperature
$P$ = measured pressure
$P_o$ = saturation pressure

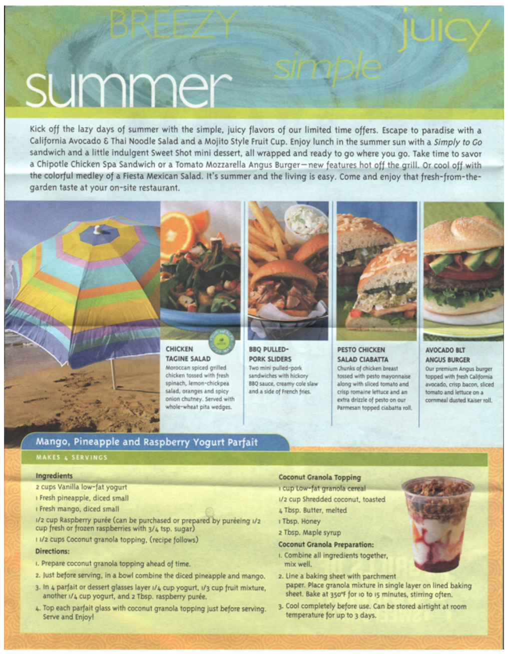
Hysteresis between the adsorption and desorption isotherms in Figure 3.1c) at relative pressures 0.6–0.9 is observed with carriers having a significant volume of mesopores (diameters between 20 (2nm) and 500 Å (50 nm)). It results because pores, which fill at a given pressure during adsorption, require a lower pressure to empty during desorption. The form of the Kelvin equation given above describes the desorption isotherm, and it is the preferred one for calculations of the pore size distribution.

**3.2.3 Pore Size by Mercury Intrusion**

For materials with pore diameters greater than about 30 Å (3 nm), the mercury intrusion method is preferred (ASTM D4284-83 1988). The penetration of mercury into the pores of a material is a function of applied pressure. At low pressures, mercury penetrates the large pores, whereas at higher pressures, the smaller pores are progressively filled. Due to the nonwetting nature of mercury on oxide carriers, penetration is met with resistance. The Washburn equation [see Eq. (3.3)] relates the pore diameter d with the applied pressure P:

$$d = -4\gamma \cos\theta / P \qquad (3.3)$$

(c)



BREEZY
juicy
simple
summer

Kick off the lazy days of summer with the simple, juicy flavors of our limited time offers. Escape to paradise with a California Avocado & Thai Noodle Salad and a Mojito Style Fruit Cup. Enjoy lunch in the summer sun with a *Simply to Go* sandwich and a little indulgent Sweet Shot mini dessert, all wrapped and ready to go where you go. Take time to savor a Chipotle Chicken Spa Sandwich or a Tomato Mozzarella Angus Burger—new features hot off the grill. Or cool off with the colorful medley of a Fiesta Mexican Salad. It's summer and the living is easy. Come and enjoy that fresh-from-the-garden taste at your on-site restaurant.

**CHICKEN TAGINE SALAD**
Moroccan spiced grilled chicken tossed with fresh spinach, lemon-chickpea salad, oranges and spicy onion chutney. Served with whole-wheat pita wedges.

**BBQ PULLED-PORK SLIDERS**
Two mini pulled-pork sandwiches with hickory BBQ sauce, creamy cole slaw and a side of French fries.

**PESTO CHICKEN SALAD CIABATTA**
Chunks of chicken breast tossed with pesto mayonnaise along with sliced tomato and crisp romaine lettuce and an extra drizzle of pesto on our Parmesan topped ciabatta roll.

**AVOCADO BLT ANGUS BURGER**
Our premium Angus burger topped with fresh California avocado, crisp bacon, sliced tomato and lettuce on a cornmeal dusted Kaiser roll.

**Mango, Pineapple and Raspberry Yogurt Parfait**

MAKES 4 SERVINGS

**Ingredients**
2 cups Vanilla low-fat yogurt
1 Fresh pineapple, diced small
1 Fresh mango, diced small
1/2 cup Raspberry purée (can be purchased or prepared by puréeing 1/2 cup fresh or frozen raspberries with 3/4 tsp. sugar)
1 1/2 cups Coconut granola topping, (recipe follows)

**Directions:**
1. Prepare coconut granola topping ahead of time.
2. Just before serving, in a bowl combine the diced pineapple and mango.
3. In 4 parfait or dessert glasses layer 1/4 cup yogurt, 1/3 cup fruit mixture, another 1/4 cup yogurt, and 2 Tbsp. raspberry purée.
4. Top each parfait glass with coconut granola topping just before serving. Serve and Enjoy!

**Coconut Granola Topping**
1 cup Low-fat granola cereal
1/2 cup Shredded coconut, toasted
4 Tbsp. Butter, melted
1 Tbsp. Honey
2 Tbsp. Maple syrup

**Coconut Granola Preparation:**
1. Combine all ingredients together, mix well.
2. Line a baking sheet with parchment paper. Place granola mixture in single layer on lined baking sheet. Bake at 350°F for 10 to 15 minutes, stirring often.
3. Cool completely before use. Can be stored airtight at room temperature for up to 3 days.

(d)

**Figure 5.1.** Example scanned document images: (a) text, (b) photo, (c) highlight, and (d) mixed. Note that in highlight documents, the highlighter marks are drawn manually by the user using a highlighter pen after the page is printed.

## 5.2 Feature Extraction

In this section, we will describe the detailed procedure for extracting the new features. Sec. Color Space presents the color space conversion. Sec. Printer Gamut-Based Features demonstrates how to retrieve and characterize the highlighter pixels based on the estimated gamut. Finally, the steps to obtain our new low-level color features are presented in Sec. Low-Level Color Features.

### 5.2.1 Color Space

The choice of the color space is essential for color image analysis [73]. $RGB$ is a widely used device-dependent space in imaging devices. However, it is not perceptually uniform. That is to say, the distance between the $RGB$ coordinates of two colors is not proportional to the human perception of such a difference [74]. For this reason, several perceptually uniform spaces have been developed, such as CIE $L^*a^*b^*$ and CIE $Lch(a^*b^*)$. In this paper, we will extract color features in the CIE $Lch(a^*b^*)$ color space.

We first convert gamma-corrected $RGB$ to linear $RGB$ using a 1D gamma uncorrection lookup table (LUT) provided by the organization sponsoring this research. This LUT is applied separately to each of the $R$, $G$, and $B$ channels of the MFP scanner output. The conversion is plotted in Fig. 5.2. Then a $3 \times 3$ matrix transformation is applied to the linearized $RGB$ values

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.5313 & 0.3519 & 0.1168 \\ 0.2742 & 0.7673 & -0.0415 \\ 0.0051 & 0.0510 & 0.9438 \end{bmatrix} \begin{bmatrix} R_L \\ G_L \\ B_L \end{bmatrix}. \tag{5.1}$$

This matrix describes the transformation from the device-dependent scanner $RGB$ of a particular MFP product to the device-independent CIE $XYZ$ color space. It was also provided by the organization sponsoring the research. Then, we convert to CIE $L^*a^*b^*$

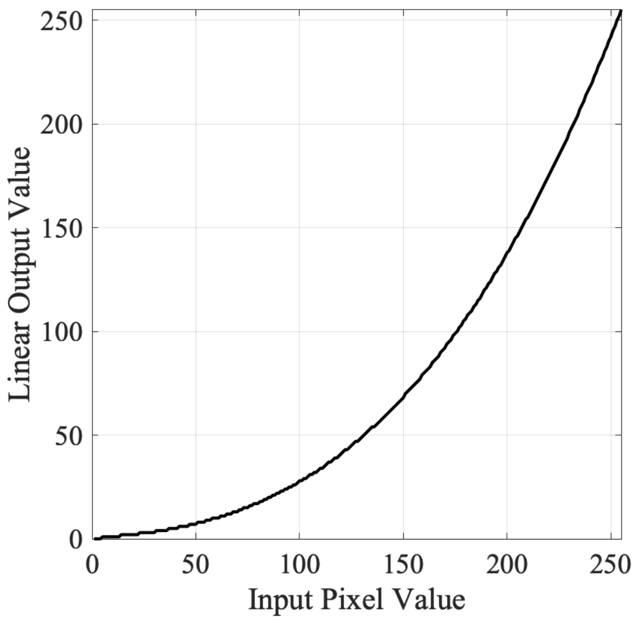


**Figure 5.2.** The gamma uncorrection conversion from gamma-corrected $RGB$ to linear $RGB$.

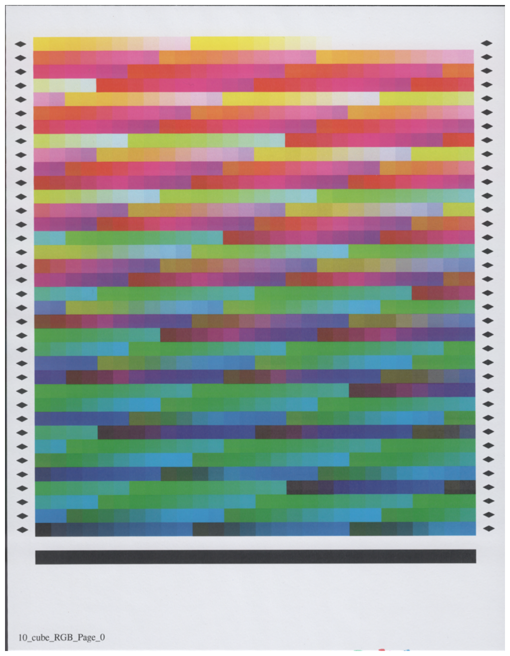
through the equations provided in [54]. Finally, the color attributes lightness ($L^*$), chroma ($C^*$), and hue ($h$) can be computed as

$$\begin{aligned} L^* &= L^*, \\ C^*(a^*, b^*) &= \sqrt{a^{*2} + b^{*2}}, \\ h(a^*, b^*) &= \arctan\left(\frac{b^*}{a^*}\right). \end{aligned} \tag{5.2}$$

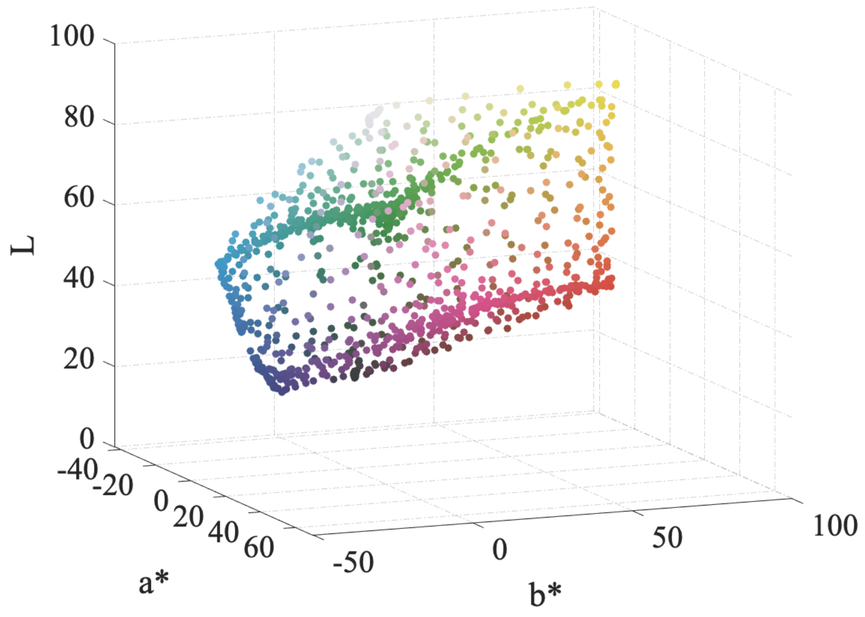
Here, $L^*$ corresponds to how light or dark a color is, $C^*$ represents the color intensity, and $h$ describes the appearance of color – color in its pure form, as in red, green, or blue.

### 5.2.2 Printer Gamut-Based Features

In our discussion here, a gamut is the range of colors that can be reproduced by the printer. In this paper, we only work with printed colors as sensed by a scanner, which has its own gamut of colors that can be uniquely sensed. However, generally, the scanner gamut is



(a)



(b)

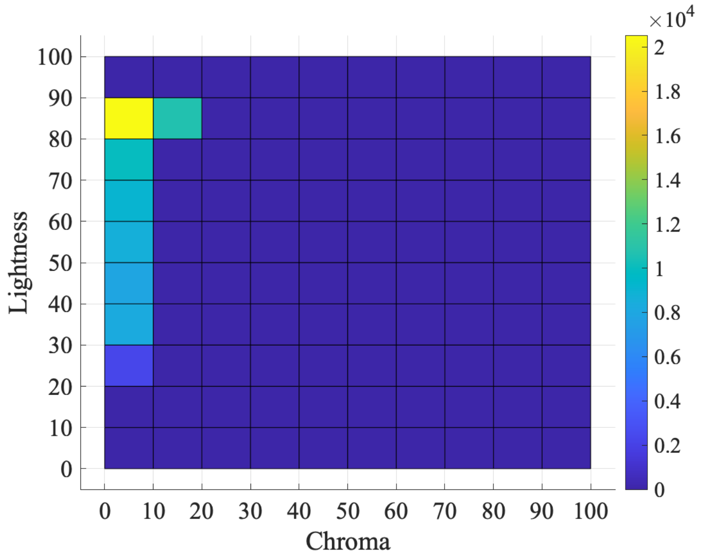
**Figure 5.3.** (a) The printed and scanned test page used to estimate the gamut. (b) The estimated gamut in CIE $L^*a^*b^*$ space. Each dot corresponds to the mean $L^*a^*b^*$ values of a patch.

larger than that of the printer. So we assume here that the printer gamut is strictly contained within the scanner gamut, and that the scanner gamut contains all highlighter colors.

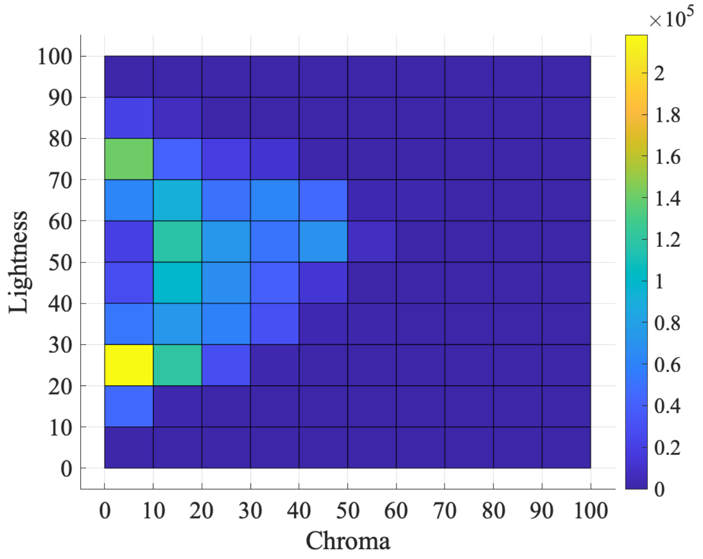
Given the fact that most colors painted by highlighter pens cannot be accurately reproduced by the printer, we will look at chromatic pixels in the scanned image lying outside of the printer gamut. We follow the procedure described in [54] to estimate the gamut using the test page shown in Fig. 5.3 (a), which has been printed with our target printer, and then scanned with our target scanner. The resulting gamut is shown in Fig. 5.3 (b). Inspired by [75], we segment the CIE $Lch(a^*b^*)$ space into 18 non-overlapping 20-degree hue slices. Then the gamut boundary of each hue slice is computed as the convex hull [76] that encompasses all points within the hue slice [77]. We refer to such a convex hull as a gamut hue sector. The vertices of the gamut sectors (convex hulls) are stored in counterclockwise order to facilitate later computations. We will soon compare each image and the gamut at every 20-degree hue slice.

Furthermore, by visual inspection, we note that some highlighter colors, such as yellow and orange, are softer than others, while others, such as magenta and purple, are more visible. From the plots of the four example highlighter patches in Fig. 5.7, one can see that the range of chroma and lightness varies from one hue slice to another. Thus, we propose to use hue-slice dependent chroma and lightness thresholds to improve the accuracy of highlighter pixel characterization. The thresholds are measured based on a scanned sheet of paper containing assorted colors of highlighter pen marks. It can also be seen from Fig. 5.7 that many highlight pixels are out of the printer gamut, adding credence to our claim that the highlighted regions often cannot be reliably reproduced. We assume that the chroma and lightness values of a highlight pixel should each fall between two thresholds. From the lightness and chroma 2D histogram in Fig. 5.4, we note that high chroma/low lightness values could be solid colors that are from a photo or some graphics, and low chroma/high lightness values could be the color of the media. Therefore, to speed up the subsequent computations, such colors can be excluded.

Now, we are going to compare each scanned image and the printer gamut at each hue slice. Specifically, we use the following procedure to extract the highlight pixels of interest. For each 20-degree hue slice $s$, we first compute a set of image pixels $\mathcal{S}^s = \{(C^*, L^*) \mid$



(a)



(b)

**Figure 5.4.** Lightness and chroma 2D histogram plots of a (a) text image and a (b) photo image. The original images are Fig. 1 (a) and (b).

$C^* \in [C^{*s}_{LB}, C^{*s}_{UB}], L^* \in [L^{*s}_{LB}, L^{*s}_{UB}]\}$, where $C^{*s}_{LB}, C^{*s}_{UB}, L^{*s}_{LB},$ and $L^{*s}_{UB}$ are the lower and upper bounds of the chroma and lightness, respectively, of hue slice $s$. Note that the set $\mathcal{S}^s$ depends on a specific image and we will repeat this process for each image in the dataset to compute their own feature values. Then, we check if each pixel in $\mathcal{S}^s$ is inside the gamut hue sector or not. Given an edge of the $s$-th gamut hue sector defined by the vertices $\mathbf{V}^s_{\mathrm{i}}(C^{*s}_{\mathrm{i}}, L^{*s}_{\mathrm{i}})$

and $\mathbf{V}^s_{\mathrm{i}+1}(C^{*s}_{\mathrm{i}+1}, L^{*s}_{\mathrm{i}+1})$, $0 \leq \mathrm{i} \leq N-2$, and a pixel $\mathbf{P}^s_{\mathrm{j}}(C^{*s}_{\mathrm{j}}, L^{*s}_{\mathrm{j}}) \in \mathcal{S}^s$, if that pixel lies in the exterior of the gamut hue sector as shown in Fig. 5.5, then based on [78], $\exists \mathrm{i} \in \{0, ...., N-2\}$ such that

$$(V^s_{\mathrm{i}+1} - V^s_{\mathrm{i}}) \times (P^s_{\mathrm{j}} - V^s_{\mathrm{i}}) < 0, \tag{5.3}$$

where $\times$ denotes cross product. Here $N$ is the total number of vertices of the gamut hue sector. Note that the sector is closed since we require that $\mathbf{V}^s_{N-1}(C^{*s}_{N-1}, L^{*s}_{N-1}) = \mathbf{V}^s_0(C^{*s}_0, L^{*s}_0)$, and again the vertices are labeled in the counterclockwise orientation.

Now that the highlight pixels of interest are recognized, we can design some features to describe them. We propose to use highlight hue count and maximum highlight strength. The first feature is designed to count the number of highlight colors marked on the document image. The second feature is designed to compute the average distance from each highlight color to the printer gamut boundary for each hue sector. Then, we take as our feature value, the maximum over all the hue sectors of this average distance.

To compute the first feature, we iterate through all hue slices, and count the number of highlight pixels in each slice. If there are a sufficient number of highlight pixels, i.e. at least 1% of the image pixels in the hue slice are highlight pixels, then the hue slice will be counted towards the total number of highlight hues. As for the second feature, we first need to calculate the distance from each highlight pixel to its corresponding gamut hue sector.

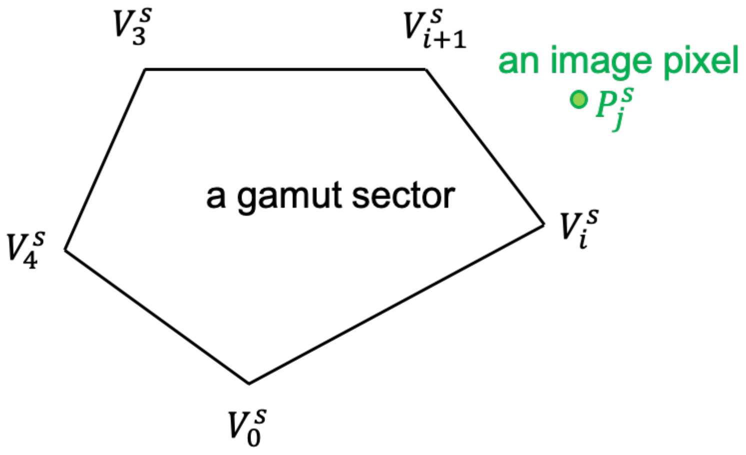


**Figure 5.5.** Visual aid for Equation (3).

For the hue sector $s$, let $\mathbf{P}_k^s$ be the $k$-th highlight pixel in the gamut sector $s$. As illustrated in Fig 5.6, there are three cases depending on the relative location of the pixel with respect to the gamut sector on the lightness-chroma plane. Let $r_{\mathrm{i},k}^s = \frac{(\mathbf{V}_{\mathrm{i}+1}^s - \mathbf{V}_{\mathrm{i}}^s)\cdot(\mathbf{P}_k^s - \mathbf{V}_{\mathrm{i}}^s)}{\|\mathbf{V}_{\mathrm{i}+1}^s - \mathbf{V}_{\mathrm{i}}^s\|^2}$, and $\mathbf{P}_k^{s\prime}$ be the projection of $\mathbf{P}_k^s$ to the edge $\mathbf{V}_{\mathrm{i}+1}^s - \mathbf{V}_{\mathrm{i}}^s$ of the hue sector $s$. Then, the distance from $\mathbf{P}_k^s$ to the edge [79] is

$$d_{\mathrm{i},k}^s = \begin{cases} \|\mathbf{P}_k^s - \mathbf{V}_{\mathrm{i}}^s\|, & \text{if} \quad r_{\mathrm{i},k}^s \leq 0 \\ \|\mathbf{P}_k^s - \mathbf{V}_{\mathrm{i}+1}^s\|, & \text{if} \quad r_{\mathrm{i},k}^s \geq 1 \\ \|\mathbf{P}_k^s - \mathbf{P}_k^{s\prime}\|, & \text{otherwise.} \end{cases} \tag{5.4}$$

Thus, the shortest distance from the highlight pixel to the periphery of the gamut sector $s$ is $d_{\min,k}^s = \min_{\mathrm{i}=0,\ldots,N-2} d_{\mathrm{i},k}^s$. Next, we compute the average distance $\bar{d}^s$ over all $K^s$ outlier pixels for the gamut hue sector $s$ according to $\bar{d}^s = \frac{1}{K^s}\sum_{k=0}^{K^s-1} d_{\min,k}^s$. Finally, the second gamut-based feature can be computed as

$$\bar{d}_{\max} = \max_{s=0,\ldots,17} \bar{d}^s. \tag{5.5}$$

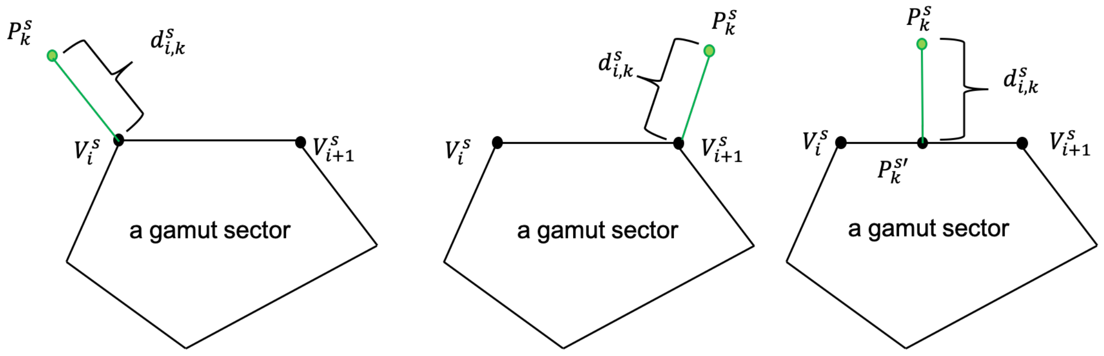


**Figure 5.6.** Visual aid for Equation (4). The three cases left to right correspond to the equations from top to bottom in Equation (4), respectively.

### 5.2.3 Low-Level Color Features

We use two properties of the highlighted regions in order to design the low-level color features. First, highlighter marks are typically bright and relatively translucent colorants drawn on a light-colored background [70]. So the average value of the lightness and chroma

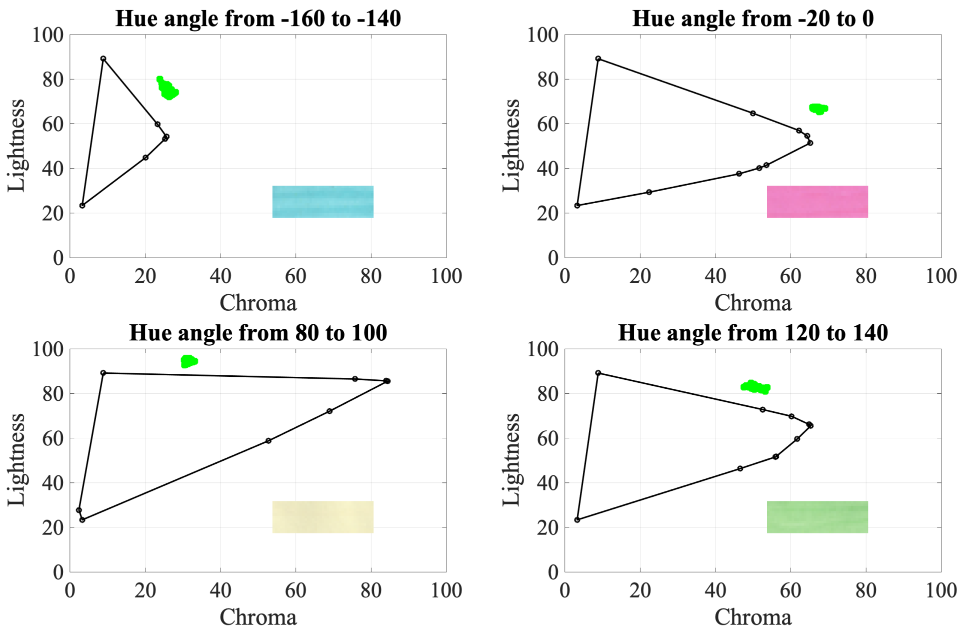


**Figure 5.7.** Four highlighter patches with their corresponding gamut hue sectors. The black convex hull indicates the gamut hue sector within the hue range. The green dots are the $(C^*, L^*)$ coordinates of the highlight pixels within the hue range.

in a highlighted image block should be higher than those of a non-highlighted block. Second, within a small region, i.e. $32 \times 32$ pixels in a document scanned at 75 dpi or 0.427 in $\times$ 0.427 in, there is usually a single highlighter color, and the fluctuations in chroma and lightness should therefore be smaller than those in the mixed and photo images that contain various colors. On the basis of these two properties, we developed six color-moment features [80], namely minimum block mean, maximum block standard deviation, and the minimum block unnormalized skewness of the lightness and chroma channels to describe the characteristics of the color distribution of the highlighted image blocks.

To be specific, we partition the query image into $32 \times 32$ non-overlapping blocks, and compute the color moments for both the $L^*$ and $C^*$ channels within each block. Let the pixel value ($L^*$ or $C^*$) of the i-th channel at the j-th image pixel in block $l$ be $I_{\mathrm{i,j}}$ and the number of image pixels in the block be $Q$, then the block color moments of channel i are defined as:

$$\begin{aligned} E_{\mathrm{i},l} &= \frac{1}{Q}\sum_{\mathrm{j}=0}^{Q-1} I_{\mathrm{i,j}} \\ \sigma_{\mathrm{i},l} &= \sqrt{\frac{1}{Q}\sum_{\mathrm{j}=0}^{Q-1}(I_{\mathrm{i,j}} - E_{\mathrm{i},l})^2} \\ s_{\mathrm{i},l} &= \sqrt[3]{\frac{1}{Q}\sum_{\mathrm{j}=0}^{Q-1}(I_{\mathrm{i,j}} - E_{\mathrm{i},l})^3} \end{aligned} \tag{5.6}$$

Finally, we compute the minimum and maximum color moment values across all blocks. These features are simple, yet effective. Intuitively, they summarize the chroma and lightness characteristics of the most prominent block in the image, which will be the highlighted block, if there is any.

## 5.3 Classification Model

Along the lines of [69], we use a DAGSVM model [72] to solve the multi-class classification problem. The DAGSVM model that we use has a tree structure, as shown in Fig. ??. It consists six 1-vs.-1 SVMs, one for each pair of the four classes. At the root level, the classifier decides if the image is in the mixed or highlight class. If it does not belong to the highlight class, then we go to the left child. If it does not belong to the mixed class, then we go to the right child. This procedure is repeated until the final decision is reached. The radial basis function (RBF) kernel is used for all the SVMs.

In our application, different misclassifications are weighted differently. For example, it is more problematic to misclassify text as photo than to misclassify text as highlight. If the text image is processed through the photo mode configuration, which has a smoothing effect, the text strokes will look too blurry. However, if the text image is processed using the highlight mode configuration, which will move the colors inside the printer gamut, the

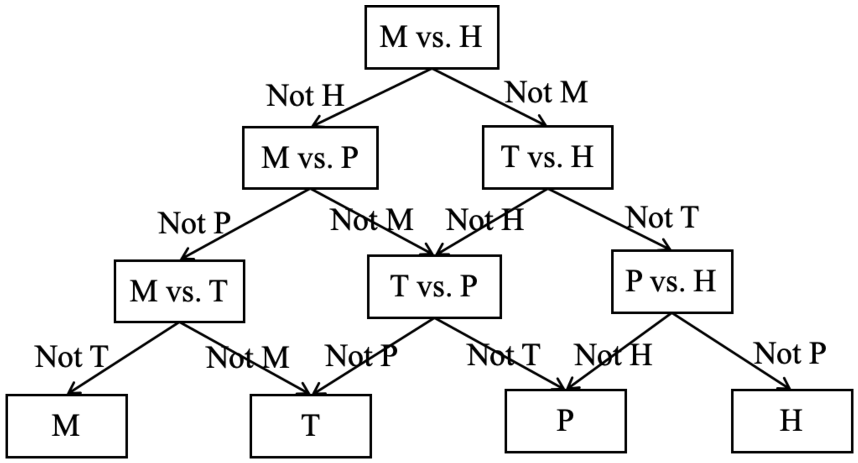


**Figure 5.8.** Illustration of the tree structure of the DAGSVM model. M = mixed, T= text, P = photo, and H = highlight.

reproduction will not be negatively impacted. Therefore, in the training process for the DAGSVM model, our goal is to minimize the weighted error

$$\mathcal{E} = \sum_{\mathrm{i,j}} W_{\mathrm{i,j}} U_{\mathrm{i,j}}, \tag{5.7}$$

where $W_{\mathrm{i,j}}$ is the weight of classifying the i-th class as the j-th class and $U_{\mathrm{i,j}}$ is the number of images in the i-th class being classified as the j-th class. The matrix $W$ is presented in Table 5.1 (a). The weights were chosen by engineers working for the organization sponsoring this research.

## 5.4 Experimental Results

Our dataset consists of the images in [69]. The images were labelled by engineers working for the organization sponsoring this research. There are in total 400 images, including 129 images in the mixed class, 84 images in the text class, 100 images in the photo class, and 87 images in the highlight class. The image contents include book and magazine pages, posters, portrait and natural pictures, handwritten notes, lecture slides, and application forms. Each image has a size of 825 × 638 pixels and a resolution of 75 dpi.

**Table 5.1.** (a) The error weight matrix $W$. It shows how different classification results weight differently towards the total cost. (b) The leave-one-out confusion matrix $U$. It summarizes the performance of our classification model. In both tables, M = mixed, T= text, P = photo, and H = highlight.

| | | Classifier Output | | | |
|---|---|---|---|---|---|
| | | M | T | P | H |
| Ground Truth | M | 0 | 3 | 5 | 4 |
| | T | 3 | 0 | 10 | 2 |
| | P | 3 | 10 | 0 | 15 |
| | H | 10 | 10 | 10 | 0 |

(a)

| | | Classifier Output | | | |
|---|---|---|---|---|---|
| | | M | T | P | H |
| Ground Truth | M | 114 | 4 | 6 | 5 |
| | T | 1 | 80 | 0 | 3 |
| | P | 4 | 0 | 96 | 0 |
| | H | 5 | 1 | 0 | 81 |

(b)

We evaluate the performance of the model using leave-one-out cross-validation (LOOCV). LOOCV repeatedly splits the data points into a training set containing all but one sample point, and a validation set containing only that remaining sample point. It provides a confusion matrix that we can use to compute the weighted error based on Equation (5.7). We employ SFFS [71] to select the best feature subset which has the minimal LOOCV weighted error. In the SFFS process, we start from an empty feature set and add one of the non-used features to the set to train the model. Then the cost function is evaluated on the validation dataset. The one feature that gives us the lowest cost will be included. After each inclusion, a number of exclusions will be performed to the current feature set if the cost can be further decreased. This process is iterated a number of times until there is no further decrease in the cost. Our final feature subset, selected by SFFS, contains 12 features, with

6 from [69] (all but the color variability score) and 6 new ones (2 gamut-based features and the first 4 color-moment features). The optimal LOOCV confusion matrix is shown in Table 5.1 (b).

Table 5.2 summarizes the results according to different document image types in terms of precision and recall, as well as the overall accuracy and weighted error. Here, precision and recall are defined [81] as

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \qquad \text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}}, \tag{5.8}$$

where TP, FP, and FN represent true positive, false positive, and false negative, respectively. It can be computed that the decrease in the weighted error is 61% and the increase in the accuracy is 10.8%. For the highlight class, the precision rises 11.5% and the recall rises 13%. These new results are relative to those reported in [69].

**Table 5.2.** Precision and recall (Equation (5.8)) for different document types and the overall accuracy and weighted error (Equation (5.7)). M = mixed, T= text, P = photo, and H = highlight.

| | Precision (%) | | | | Recall (%) | | | | Accur-acy(%) | $\mathcal{E}$ |
|---|---|---|---|---|---|---|---|---|---|---|
| | M | T | P | H | M | T | P | H | | |
| [69] | 80.6 | 76.2 | 89.0 | 81.6 | 83.9 | 81.0 | 84.0 | 78.0 | 82.0 | 3.6 |
| Ours | **88.4** | **95.2** | **96.0** | **93.1** | **91.9** | **94.1** | **94.1** | **91.0** | **92.8** | **1.4** |

## 5.5 Conclusion

A set of highlighter features is proposed. We utilize the characteristics of the highlighter colors and their distribution to describe the highlighted document images. The identification performance of the highlight class was evaluated by means of precision and recall. The overall performance of the model was measured by accuracy and the weighted cost. The newly added features significantly enhance the performance and substantially decrease the cost.

# 6. A COLOR IMAGE ANALYSIS TOOL TO HELP USERS CHOOSE A MAKEUP FOUNDATION COLOR

## 6.1 Introduction

With the rapid evolution of mobile phone technologies, people can do more and more things on their phones. Virtual makeup try-on is one of the popular mobile applications nowadays. It allows users to test out makeup shades via images or live camera, which makes cosmetic shopping a lot more convenient and fun. Many researches have focused on this field. Bhatti et al [82] developed a mobile image-based application to give women personalized cosmetic recommendations. Tong et al [83] extracted makeup information from before-and-after image pairs and transferred the makeup to a new face. Work in [84] is based on the similar idea. There are also some papers using deep neural networks [85], [86] to analyze makeup styles. Most of these papers are targeting the facial attribute analysis and style transfer. In this chapter, we will study the color change of the skin before-and-after the makeup is applied. We will focus on a specific line of foundation products.

Taking images with a mobile phone can be easy, but extracting reliable colors from the images remains a problem. Because of imperfect lighting conditions, different camera sensors sensitivity, and various post-processing in the image processing pipeline of the camera, the same product can look different on different images.

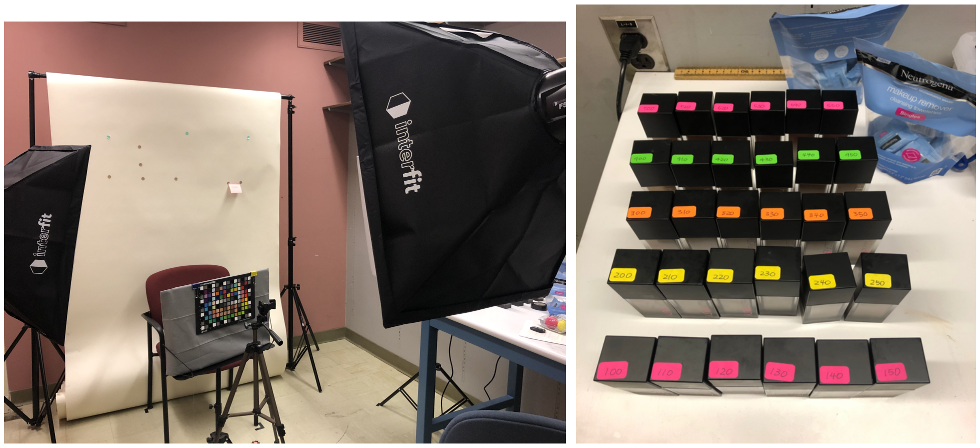
To address this color disparity issue, researchers have developed various color correction algorithms. These algorithms can be generally classified into two-fold: color constancy and image calibration. Color constancy is done by estimating and removing the influence of illuminations. Some popular color constancy algorithms include Gray World and Max RGB. A major limitation of these two algorithms is that they strongly rely on particular assumptions. If some of the assumptions do not hold, then the estimation can be inaccurate. Some more recent neural network algorithms [87]–[89] can achieve high accuracy but also require lots of computer resources, compared to some traditional methods. On the other hand, image calibration approaches are simple and effective. They directly map the device-dependent $RGB$ color values to some standard color values with the help of a calibration target. This mapping can be applied to any camera and makes very few assumptions about the characteristics of the images taken by the camera. The mapping includes three-dimensional look-up tables combined with interpolation and extrapolation [90], machine learning models [91], and neural networks [92], [93]. Despite the fact that a wide variety of calibration methods are available, researchers have favored regression-based approaches [54], [94], [95] due to their simplicity and feasibility.

In this chapter, we propose an image analysis tool that can predict the skin-with-foundation color based on the color information retrieved from calibrated selfie and foundation swatch images. To avoid illuminance inconsistency across multiple images of the same subject, we use a protocol to collect image data under controlled lighting conditions. To minimize the color correction errors, we group the color patches on the color checker into three sets and compute the mapping from the camera-dependent $RGB$ space to the standard CIE $XYZ$ for these three sets separately. Next, the CIE $L^*a^*b^*$ color coordinates of the skin pixels, foundation pixels, and the skin with foundation pixels are extracted. Finally, a linear regression model and a support vector regression (SVR) [96] model are trained using these color coordinates.

## 6.2 Data Collection

The image calibration method proposed in this chapter relies on a standard calibration target. We used the X-rite ColorChecker digital SG 140, which is shown in Fig. 6.2. The target foundation products are shown in Fig. 6.1 (b). The detailed procedure is as the following:

- Take a selfie photo with no foundation applied.
- Choose 3 or 4 foundation shades that are close to the actual skin tone by testing each possible match on the skin.
- Evenly apply the foundation on the skin using a new and clean sponge.
- Wait about three minutes for the foundation to dry, and then take a selfie photo.
- Remove the foundation completely using makeup removal wipes.



(a) (b)

**Figure 6.1.** Data collection experiment (a) lab settings and (b) the makeup foundation bottles.

- Wait about three minutes for the skin to calm down, and apply the next shade.
- Repeat the process until all chosen shades are applied and photographed.

The experiment was conducted under controlled lighting in a lab. The lab setting is shown in Fig. 6.1 (a). The light sources are three 4700K LED light bulbs. We installed diffuser panels to make the light as well-diffused as possible. The color checker and the mobile phone are mounted on tripods. The subject uses a Bluetooth remote button to take selfie photos. Each subject is instructed to sit at a fixed distance from the camera, so that the amount of the light reflected from his or her face is about the same for all images. The images in Fig. 6.3 are sample original photos that we collected from this experiment.

## 6.3 Image Calibration

In this section, we will present the details of the image calibration procedure, which will lay the groundwork for an accurate prediction model. We will start by introducing the skin pixel detection algorithm. And then we describe the two major steps in the proposed calibration framework. Finally, the calibration performance is evaluated by means of the color differences in CIE $L^*a^*b^*$ space.

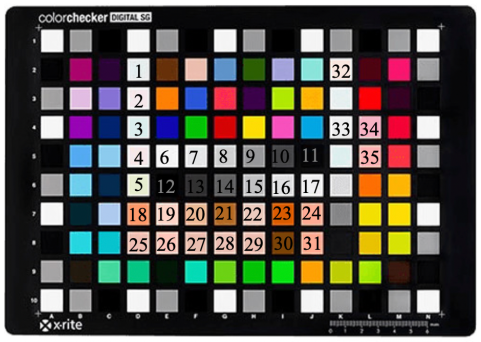


**Figure 6.2.** X-rite digital SG 140 color checker. Patches that are used in the color correction stage are numbered from 1 to 35.

### 6.3.1 Skin Detection

Since we mainly interested in analyzing and predicting skin colors, the facial skin is first segmented. A fast and efficient RGB-H-CbCr model [97] for this purpose is adopted. Under a uniform illumination condition, a skin pixel, defined by [97], should satisfy all of the following three criteria.

Criterion 1:

$$\begin{gathered} R > 95 \quad \wedge \quad G > 40 \quad \wedge \quad B > 20 \\ \max(R, G, B) - \min(R, G, B) > 15 \\ |R - G| > 15 \quad \wedge \quad R > G \quad \wedge \quad R > B \end{gathered} \tag{6.1}$$

Criterion 2:

$$\begin{aligned} C_r &\leq 1.5862 \cdot C_b + 20 \quad \wedge \\ C_r &\geq 0.3448 \cdot C_b + 76.2069 \quad \wedge \\ C_r &\geq -4.5652 \cdot C_b + 234.5652 \quad \wedge \\ C_r &\leq -1.15 \cdot Cb + 301.75 \quad \wedge \\ Cr &\leq -2.2857 \cdot Cb + 432.85 \quad \wedge \end{aligned} \tag{6.2}$$

Criterion 3:

$$H < 25 \quad \vee \quad H > 230 \tag{6.3}$$

Here, $\wedge$ denotes the logical AND operation, and $\vee$ denotes the logical OR operation. An example output of this skin detection algorithm is shown in Fig. 6.4. It can be seen from

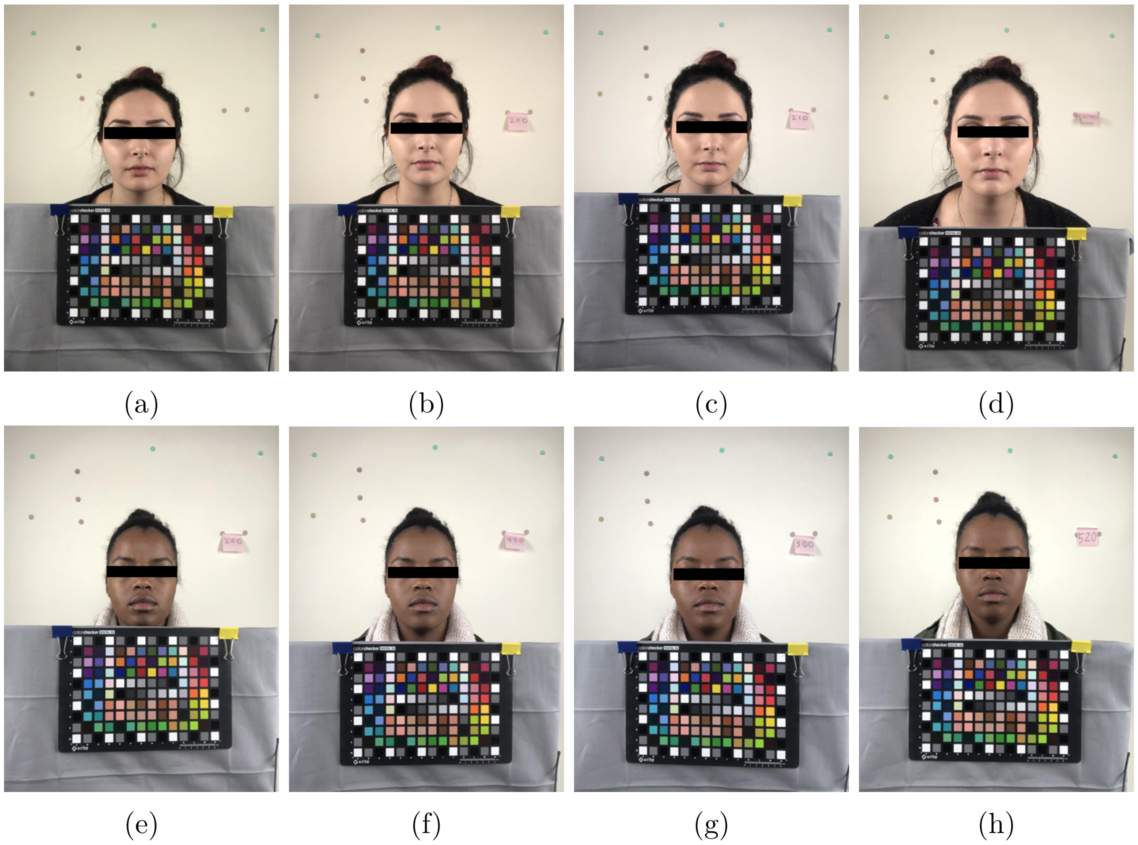


**Figure 6.3.** Sample original images. Subject No.1 (a light-skinned subject): (a) skin with no foundation, (b) skin with foundation shade No. 130, (c) skin with foundation shade No. 150, and (d) skin with foundation shade No. 200. Subject No. 2 (a dark-skinned subject): (a) skin with no foundation, (b) skin with foundation shade No. 450, (c) skin with foundation shade No. 500, and (d) skin with foundation shade No. 520.

Fig. 6.4, although some non-skin pixels are picked up by the segmentation mask, most of the skin pixels are correctly identified. Note that we will find the average value within the skin region, as discussed later, so a few non-skin pixels will not significantly degrade the estimation. It can also be seen from Fig. 6.4 that the algorithm can effectively handle various skin complexions across different ethnicities.

### 6.3.2 Gray Balancing and Polynomial Transformation

Image calibration is a process that converts device-dependent $RGB$ values into CIE $XYZ$ values. So we need to obtain the $RGB$ values of the target color patches in the

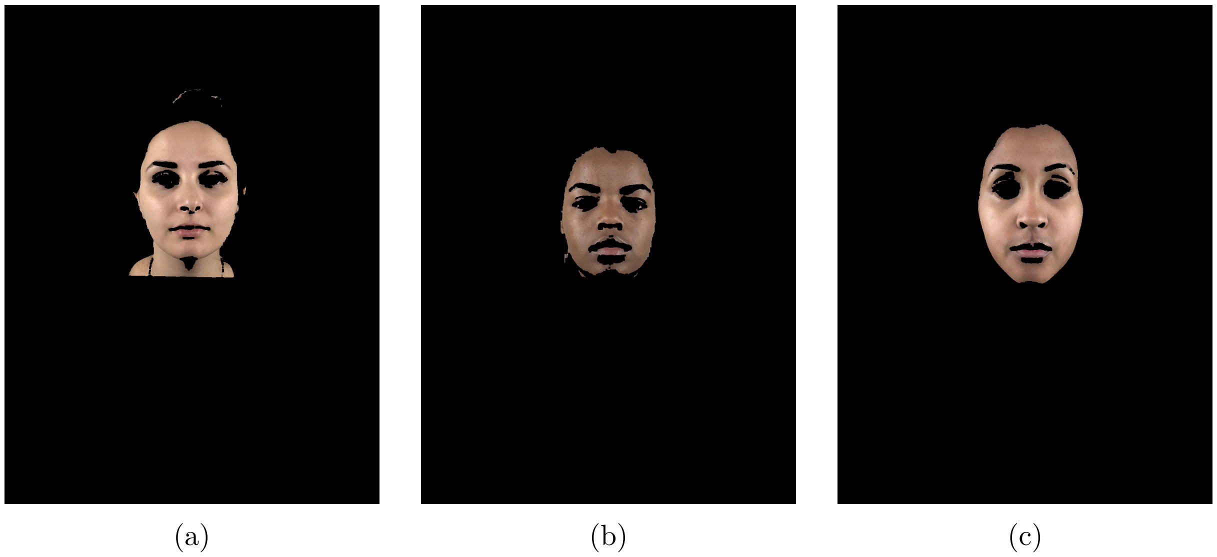


**Figure 6.4.** Skin detection output of the skin with no foundation images of (a) subject No.1 (light-skinned), (b) subject No. 2 (dark-skinned), and (c) subject No. 3 (tan-skinned).

original image as well as their corresponding CIE $XYZ$ values. The patches of interest are labelled 1 through 35 in Fig. 6.2. The CIE $XYZ$ values are measured with an X-Rite spectrophotometer under D50 illuminant, and the $RGB$ values are extracted by averaging over the center region of each patch for each color channel.

Now that we have the ($RGB$,CIE $XYZ$) pairs, we can estimate the color mapping between them. We will utilize a two-step process [54], namely gray balancing and polynomial regression. Gray balancing aims to remove the color cast and avoid having one particular dominant hue in the image. The methodology is based on [54]. We assume that the linear $RGB$ values of each patch and the CIE $Y$ (Luminance) value of the neutral gray patches are related by

$$R_l = G_l = B_l = Y. \tag{6.4}$$

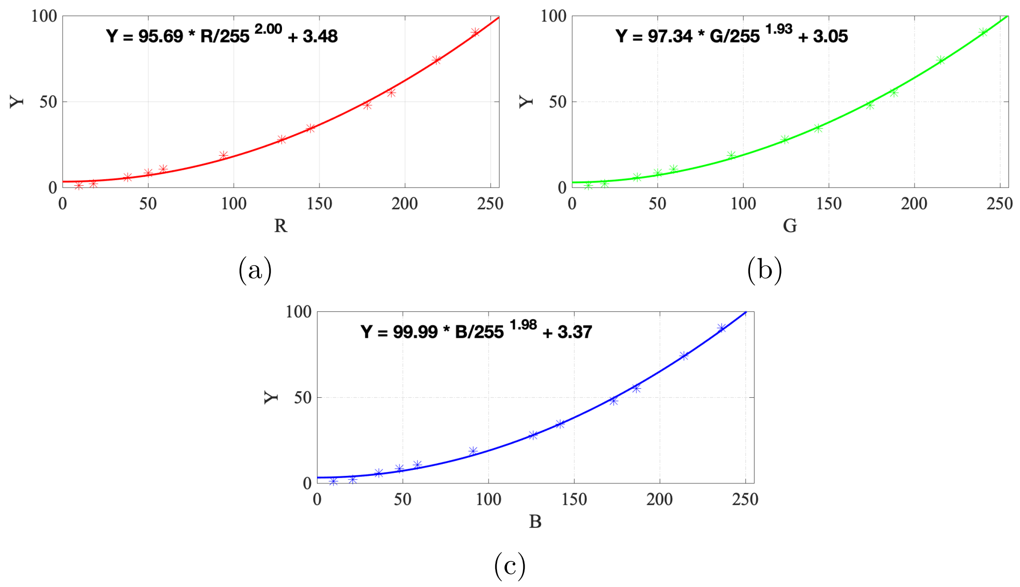


**Figure 6.5.** The gray balancing curves of the skin with no foundation image of subject No. 1. (a) The $R$ channel. (b) The $G$ channel. (c) The $B$ channel. Note that the gray balancing is based only on the color checker patches, and does not utilize the skin pixels.

Twelve neutral gray patches (patches No. 6 to No. 17 in Fig. 6.2) on the color checker are used in this step. This gives us twelve pairs of $(R_\gamma, R_l), (G_\gamma, G_l)$, and $(B_\gamma, B_l)$ for each image. We then fit a Gain-Gamma-Offset model [54] to them, such that

$$\begin{aligned} R_l &= \alpha_R \left(\frac{R_\gamma}{255}\right)^{\gamma_R} + o_R, \\ G_l &= \alpha_G \left(\frac{G_\gamma}{255}\right)^{\gamma_G} + o_G, \\ B_l &= \alpha_B \left(\frac{B_\gamma}{255}\right)^{\gamma_B} + o_B, \end{aligned} \tag{6.5}$$

where $\alpha_{\rm i}, \gamma_{\rm i}$, and $o_{\rm i}$ $({\rm i} = R, G, B)$ are the gain, gamma, and offset. The gray balancing curves of a sample image are shown in Fig. 6.5. The gamma and offsets values for this particular image are summarized in Table. ??.

After obtaining the linearized $RGB$ values, we can continue to the regression step. To maximize calibration accuracy, the target patches are classified into three sets, and the calibration mapping is trained separately on each set. We refer to these three training sets as Set 1, Set 2, and Set 3, respectively. We use the following procedure for patch grouping. To

**Table 6.1.** Gain, gamma, and offset values for each of the $R$, $G$, and $B$ channels for a sample image in the dataset.

| Channel | Gain | Gamma | Offset |
|---|---|---|---|
| R | 95.69 | 2.00 | 3.48 |
| G | 97.34 | 1.93 | 3.05 |
| B | 99.99 | 1.98 | 3.37 |

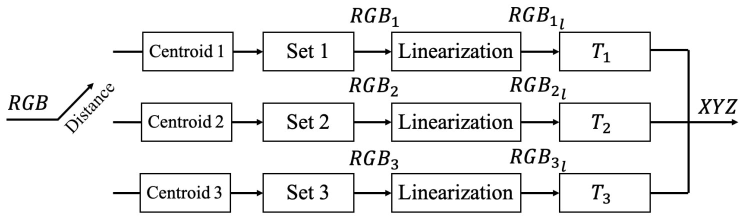


**Figure 6.6.** Block diagram of the calibration procedure.

form Set 1, we first compute the mean $RGB$ values of the pixels within the skin segmentation mask described in the previous section. We refer to this mean as the centroid of Set 1. Then, we find the patches from which the Euclidean distance is less than 80 to the centroid of Set 1. Similarly, the patches in Sets 2 and 3 are grouped based on the centroids determined by the K-means algorithm. The centroids of these three training sets will be used again when classifying image pixels for the entire image, as discussed later. Finally, we compute three different transformation matrices $T_\mathrm{i}$ using the polynomial regression. Let $Q$ denote the $N_\mathrm{i} \times 11$ matrix consisting of the polynomial terms of the linearized $RGB$ values of the target patches.

$$\begin{aligned} Q_\mathrm{i} = [&R_{\mathrm{i},l} \quad G_{\mathrm{i},l} \quad B_{\mathrm{i},l} \quad R^2_{\mathrm{i},l} \quad G^2_{\mathrm{i},l} \quad B^2_{\mathrm{i},l} \\ &R_{\mathrm{i},l}G_{\mathrm{i},l} \quad G_{\mathrm{i},l}B_{\mathrm{i},l} \quad R_{\mathrm{i},l}B_{\mathrm{i},l} \quad R_{\mathrm{i},l}G_{\mathrm{i},l}B_{\mathrm{i},l} \quad 1], \end{aligned} \tag{6.6}$$

where $N_\mathrm{i}$ indicates the number of patches in the i-th set. Let $P$ denote the $N_\mathrm{i} \times 3$ matrix consisting of the measured CIE $XYZ$ values of the patches.

$$P_\mathrm{i} = [X_\mathrm{i} \quad Y_\mathrm{i} \quad Z_\mathrm{i}]. \tag{6.7}$$

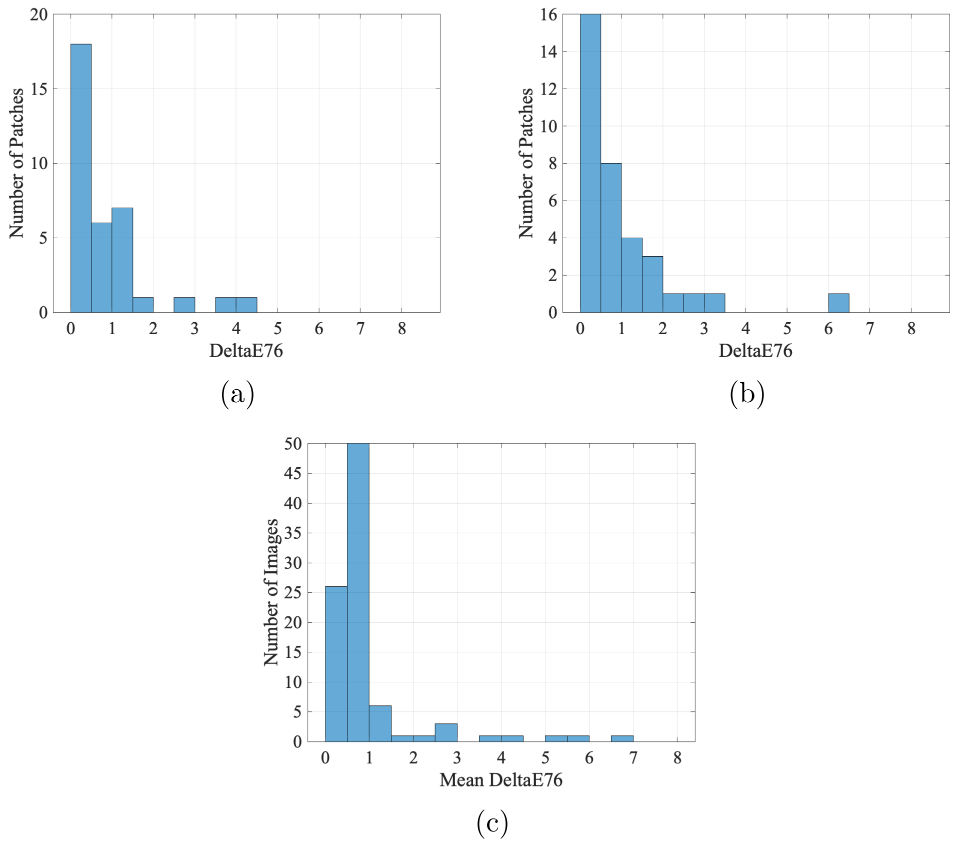


**Figure 6.7.** (a) The histogram of the calibration $\Delta E76$ across all 35 patches for the skin with no foundation image of subject No. 1 and (b) subject No. 2. The mean calibration $\Delta E76$ values over all 35 patches for subject No. 1 and subject No. 2 are 0.75 and 0.91, respectively. (c) The histogram of the mean calibration $\Delta E76$ values across all images of all subjects. The mean calibration $\Delta E76$ value across all images is 1.82; and the standard deviation is 3.50. Note that there are only five images with mean $\Delta E76 > 3$. These were eliminated from the subsequent analysis as outliers.

Then, the optimal $3 \times 11$ transformation matrix can be computed as

$$T_\mathrm{i} = (Q_\mathrm{i}^T Q_\mathrm{i})^{-1} Q_\mathrm{i}^T P_\mathrm{i}. \tag{6.8}$$

Now, we are ready to apply the transformation matrices to the entire image. As illustrated by Fig. 6.6, the image pixels will be classified into three sets based on their distance to the three pre-computed centroids, and then the linearized $RGB$ values will be converted to CIE $XYZ$ using the corresponding matrices.

### 6.3.3 Calibration Performance

The color correction stage is now completed, and we can evaluate the accuracy of it. In order to make the calibration more perceptually relevant, the color coordinates are converted from CIE $XYZ$ to CIE $L^*a^*b^*$ space. The conversion is defined as

$$\begin{aligned} L^* &= 116 \cdot f\left(\frac{Y}{Y_n}\right) - 16 \\ a^* &= 500 \cdot \left(f\left(\frac{X}{X_n}\right) - f\left(\frac{Y}{Y_n}\right)\right) \\ b^* &= 200 \cdot \left(f\left(\frac{Y}{Y_n}\right) - f\left(\frac{Z}{Z_n}\right)\right), \end{aligned} \tag{6.9}$$

where

$$f(t) = \begin{cases} \frac{1}{3}\left(\frac{116}{24}\right)^2 + \frac{16}{116}, & \text{if} \quad x \leq \frac{24}{116}, \\ x^{\frac{1}{3}}, & \text{if} \quad x > \frac{24}{116}. \end{cases} \tag{6.10}$$

Here, $(X_n, Y_n, Z_n)$ are the CIE $XYZ$ tristimulus values of the white point. With CIE standard D50 illumination, $X_n = 96.42, Y_n = 100$ and $Z_n = 82.52$.

Then, the color difference between the ground truth and the calibrated CIE $L^*a^*b^*$ coordinates of each of the patches are computed. The color difference is defined as the Euclidean distance between the color coordinates $(L_1^*, a_1^*, b_1^*)$ and $(L_2^*, a_2^*, b_2^*)$, such that

$$\Delta E76 = \sqrt{(L_1^* - L_2^*)^2 + (a_1^* - a_2^*)^2 + (b_1^* - b_2^*)^2}. \tag{6.11}$$

The resulting histograms for two sample images are shown in Figs. 6.7 (a) and (b). As can be seen from Figs. 6.7 (a) and (b), out of 35 patches, 33 patches have an $\Delta E76$ value less than 3 for both cases. So for these two images, color correction is quite effective. Figure 6.7 (c) shows the histogram of the mean $\Delta E76$ value over 35 patches for all images. It can be seen that the mean calibration error for the majority of the images is too small to be noticed, i.e. $\Delta E76 \leq 1$. This suggests that the proposed calibration algorithm works well on the dataset. The five images with a mean $\Delta E76$ value greater than 3 are considered to be outliers, and are therefore excluded from the subsequent analysis.

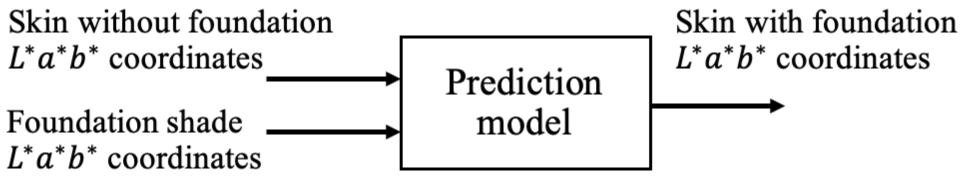


**Figure 6.8.** Illustration of the input and output of the prediction model.

## 6.4 Experimental Results

With the outliers taken out, the remaining images consist of 63 skin images with and without foundation pairs. These images are taken by 19 subjects. The distribution of the skin tone types is summarized in Table 6.2. After color correcting all the images using our

**Table 6.2.** Number of subjects and number of images in each skin tone category.

| Skin Tone Type | Fair | Light | Medium | Tan | Dark |
|---|---|---|---|---|---|
| Number of Subjects | 4 | 4 | 4 | 3 | 3 |
| Number of Image Pairs | 14 | 13 | 14 | 12 | 10 |

proposed method, we obtain 63 pairs of CIE $L^*a^*b^*$ coordinates. Apart from the selfie photos of the subjects, we also collected an image that contains the swatches of all the foundation shades. The foundation shades are applied on a white cardboard using makeup sponges. The original foundation swatches image is shown in Fig. 6.9. The same color correction procedure is performed on the image, except that the three sets are now all determined by K-means. The mean calibration $\Delta E76$ of this foundation swatches image is 0.34.

Now, we can develop a model to predict the CIE $L^*a^*b^*$ coordinates of the skin-with-foundation color, given the CIE $L^*a^*b^*$ coordinates of a skin-with-no-foundation color and the CIE $L^*a^*b^*$ coordinates of a foundation color, as illustrated in Fig. 6.8. We explored two common machine learning models. They are linear regression and SVR [96] with a linear kernel.

We denote the 6-dimensional input vector (i.e., the CIE $L^*a^*b^*$ coordinates of a skin-with-no-foundation color and the CIE $L^*a^*b^*$ coordinates of a foundation color) of the i-th

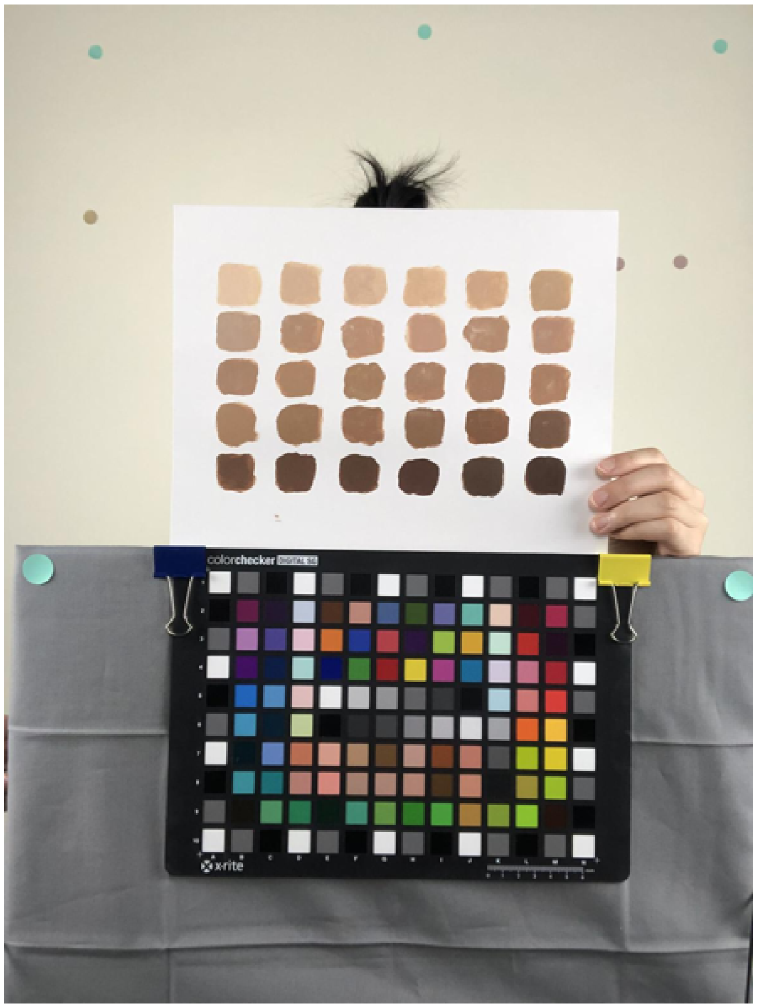


**Figure 6.9.** The original foundation swatches image. From top to bottom and from left to right, the shades are No. 100, 110, 120, 130, 140, 150, No. 200, 210, 220, 230, 240, 250, 300, 310, ... ..., 540, 550.

sample as $x_{\mathrm{i}}$ and the corresponding 3-dimensional ground truth label (i.e., the CIE $L^*a^*b^*$ coordinates of the skin-with-foundation color) as $y_{\mathrm{i}}$. Then, the linear regression problem can be formulated as

$$J(\theta) = \sum_{\mathrm{i}=1}^{N} \mathcal{L}(\theta^T x_{\mathrm{i}}, y_{\mathrm{i}}) = ||\theta^T x_{\mathrm{i}} - y_{\mathrm{i}}||^2, \tag{6.12}$$

where $\theta = [w, b]^T$, $\theta^T x_{\text{i}}$ is the prediction made by the model, and $N$ is the number of samples in the dataset. We further define a $N \times 7$ matrix

$$A = \begin{pmatrix} x_1^T & 1 \\ x_2^T & 1 \\ ... & ... \\ x_N^T & 1 \end{pmatrix}, \tag{6.13}$$

and a $N \times 3$ matrix

$$Y = \begin{pmatrix} y_1^T \\ y_2^T \\ ... \\ y_N^T \end{pmatrix}. \tag{6.14}$$

Then, (6.12) can be converted to the vector form such that $J(\theta) = ||A\theta - Y||^2$. Thus, the optimized solution can be expressed as

$$\hat{\theta} = \underset{\theta}{\operatorname{argmin}}\, J(\theta) = (A^T A)^{-1} A^T Y. \tag{6.15}$$

The SVR algorithm is based on the same concept as the support vector machine, except that it uses the support vectors for soft margins in the regression process rather than classification. The regression problem with a SVR model is defined as

$$\underset{w,b}{\operatorname{argmin}} \frac{1}{2}||w||^2 + C \sum_{\text{i}}^{N} (\xi_{\text{i}} + \xi_{\text{i}}^*), \tag{6.16}$$

subject to

$$\begin{gathered} y_{\text{i}} - w x_{\text{i}} - b \leq \epsilon + \xi_{\text{i}} \\ w x_{\text{i}} + b - y_{\text{i}} \leq \epsilon + \xi_{\text{i}}^* \\ \xi_{\text{i}}, \xi_{\text{i}}^* > 0, \end{gathered} \tag{6.17}$$

where $w$ and $b$ are model parameters, $C$ is the box constraint, $\epsilon$ is the margin around the decision boundary, and $\xi_{\text{i}}, \xi_{\text{i}}^*$ are the slack variables. The Python package *Scikit-learn* is used to implement linear regression and SVR.

A common way to evaluate the performance of a regression model is to use $K$-fold cross-validation. In $K$-fold cross-validation, the dataset is randomly split into $K$ equal-sized folds. We train the model on the data in $K-1$ of the folds and evaluate the model on the remaining one fold, namely, the validation fold. We then repeat this process $K$ times. Given the fact that we have limited data in the dataset, we choose $K = N$, where $N$ is the number of all data points. This method is referred to as the leave-one-out cross-validation (LOOCV) method. That is to say, in each trial, the predictor is trained on all but one data point, and the prediction is made for that retained point. Then the performance can be computed as the average over the $N$ trials. The advantage of using LOOCV is that each data point gets the chance to be allocated into both the testing set and $N-1$ of the training sets. The selection bias is therefore decreased.

To determine the goodness of model fit, the coefficient of determination, denoted by $R^2$, is computed. In the context of regression, it is a measure of the proportion of the prediction error that can be attributed to the variance in the independent input variables. It is defined as:

$$R^2 = 1 - \frac{\sum_{\mathrm{i}=1}^{N}(y_\mathrm{i} - \hat{y}_\mathrm{i})^2}{\sum_{\mathrm{i}=1}^{N}(y_\mathrm{i} - \bar{y})^2}, \tag{6.18}$$

where $y_\mathrm{i} - \hat{y}_\mathrm{i}$ is the error from the prediction $\hat{y}_\mathrm{i}$ to the ground truth $y_\mathrm{i}$ of the i-th data point, and $\bar{y}$ is the mean of the ground truth $y$ values. The $R^2$ score is in the range of $[0, 1]$. A score of 0 means that the dependent variable cannot be predicted from the independent variable, and a score of 1 means the dependent variable can be predicted with no error from the independent variable.

Besides, the mean squared error (MSE) and mean absolute error (MAE) are also used to evaluate the accuracy of prediction results. The MSE can be expressed as

$$MSE = \frac{\sum_{\mathrm{i}=1}^{N}(p_\mathrm{i} - g_\mathrm{i})^2}{N}, \tag{6.19}$$

where $p_\mathrm{i}$ and $g_\mathrm{i}$ are the predicted value and the ground truth value of the i-th sample point, respectively. Similarly,

$$MAE = \frac{\sum_{\mathrm{i}=1}^{N}|p_\mathrm{i} - g_\mathrm{i}|}{N}. \tag{6.20}$$

Table 6.3 summarizes the LOOCV results of the linear regression model and the SVR model in terms of $R^2$, average MSE, and average MAE. It can be seen that the average MSE and MAE values are less than 1.5 $\Delta E$ and the $R^2$ value is high for both of the models. This implies that the prediction models can accurately predict the skin with foundation color on the dataset.

**Table 6.3.** LOOCV $R^2$, MSE, and MAE results.

| Model | $R^2$ | Average MSE | Average MAE |
|---|---|---|---|
| Linear Regression | 0.83 | 1.50 | 0.91 |
| SVR with a Linear Kernel | 0.82 | 1.49 | 0.87 |

## 6.5 Conclusion

The selfie images are calibrated using a subset of color checker patches. The pixels are classified into three sets according to the Euclidean distance from the $RGB$ values of the pixel to the three designated centroids. Three different transformation matrices are computed separately and then applied to the corresponding pixels in the image. The calibration accuracy is measured by the color difference $\Delta E$ between the reference value and the calibrated value in CIE $L^*a^*b^*$ space. The $\Delta E$ results indicate that the error produced by the proposed method is almost not distinguishable for most of the images. A prediction model is then built upon the calibrated selfie images. The prediction performance is measured by $R^2$, $MSE$, and $MEA$. LOOCV results show that the prediction made by both linear regression and SVR with a linear kernel is reliable.

# 7. SUMMARY AND CONTRIBUTIONS

Let us review the contributions of this work.

In this thesis, we designed five image processing and/or learning-based vision algorithms that give better image quality, low complexity, and high accuracy.

In Chapter 1, we proposed a new parallel image processing path to accommodate the hardware architecture of a printer. Chapter 2 proposed an ink dot profile model to eliminate detail loss and noisiness in the print generated by the printers that exhibit irregular drop shape errors. Chapter 3 proposed an ink drop displacement model to reduce severe image quality degradation caused by ink drop misalignment and random displacement of the ink drops by the printer. Chapter 4 proposed a set of features to use along with a multi-class SVM so that highlighted documents can be more accurately classified. Chapter 5 designed a new color calibration method to better study the color change in skin complexion before-and-after applying makeup foundation products.

To sum up, the main contributions of this thesis are:

4-Row Serpentine Tone Dependent Error Diffusion

- Designed a novel multi-row serpentine scan path for the tone dependent fast error diffusion algorithm that realizes parallel processing of the original algorithm.
- Designed a set of new error weighting matrices so that the worm-like artifacts can be dispersed.
- Adopted a new loss function in the training system which results in more reliable and smoother renderings.

Dot Profile Model-Based Direct Binary Search

- Designed a test page to capture the ink drop shape of the target printer.
- Developed an analytical dot profile model that characterizes dot shape irregularity errors of the printer.
- Explored the printer model at two different resolutions, i.e. SD (printer resolution), HD (3$\times$printer resolution).

- Embedded both the SD and HD dot profile models into the direct binary search algorithm and achieved better output quality.

Ink Drop Displacement Model-Based Direct Binary Search

- Designed a test page to capture the ink drop displacement of the target printer.
- Analyzed the stochastic behaviors of printer ink drops using image processing techniques.
- Developed a statistical model to predict ink drop displacements, and embedded the prior knowledge of the statistics of the printer defects into the halftoning algorithm.
- Designed a Python GUI and conducted mean opinion score experiments to quantify the perceived image quality.

Highlighted Document Image Classification

- Designed features to characterize the content, color distribution and highlighter marks on the document images.
- Developed machine learning models to classify scanned document images according to the content of the document. The proposed method achieved significant improvement in document classification accuracy compared to previous method.
- Translated Python code to C for use in the embedded system.

A Color Image Analysis Tool to Help Users Choose a Makeup Foundation Color

- Designed and fully implemented a laboratory and protocol for capturing ground truth selfie images of subjects with and without makeup and an image of the makeup foundations applied to a white board.
- Developed a color correction algorithm to reduce color distortion in mobile phone images.
- Implemented machine learning models to predict makeup foundation color with given mobile phone images.

# REFERENCES

[1] R. W. Floyd and L. Steinberg, “An adaptive algorithm for spatial gray-scale,” in *Proc. of the Society for Information Display*, vol. 17, 1976, pp. 75–77.

[2] I. H. Witten and R. M. Neal, “Using peano curves for bilevel display of continuous-tone images,” *IEEE Computer Graphics and Applications*, vol. 2, no. 3, pp. 47–52, 1982.

[3] D. E. Knuth, “Digital halftones by dot diffusion,” *ACM Trans. on Graphics*, vol. 6, no. 4, pp. 245–273, 1987.

[4] R. Ulichney, *Digital Halftoning.* MIT press, 1987.

[5] C. Billotet-Hoffmann and O. Bryngdahl, “On the error diffusion technique for electronic halftoning,” *Proc. of the Society for Information Display*, vol. 24, no. 3, pp. 253–258, 1983.

[6] R. L. Miller and C. M. Smith, *Image processor with error diffusion modulated threshold matrix*, US Patent 5,150,429, Sep. 1992.

[7] J. Sullivan, R. Miller, and G. Pios, “Image halftoning using a visual model in error diffusion,” *J. Opt. Soc. Amer. A*, vol. 10, no. 8, pp. 1714–1724, 1993.

[8] R. Eschbach, “Error diffusion algorithm with homogenous response in highlight and shadow areas,” *J. of Electron. Imaging*, vol. 6, no. 3, pp. 348–356, 1997.

[9] P. W. Wong, “Adaptive error diffusion and its application in multiresolution rendering,” *IEEE Trans. Image Processing*, vol. 5, no. 7, pp. 1184–1196, 1996.

[10] B. W. Kolpatzik and C. A. Bouman, “Optimized error diffusion for image display,” *J. of Electron. Imaging*, vol. 1, no. 3, pp. 277–293, 1992.

[11] P. W. Wong and J. P. Allebach, “Optimum error-diffusion kernel design.,” in *Proc. SPIE Color Imaging: Device-Independent Color, Color Hard Copy, and Graphic Arts*, 1997, pp. 236–242.

[12] R. Eschbach, “Reduction of artifacts in error diffusion by means of input-dependent weights,” *J. of Electron. Imaging*, vol. 2, no. 4, pp. 352–359, 1993.

[13] J. Shu, “Adaptive filtering for error-diffusion quality improvement,” in *Symposium Digest of Technical Papers*, vol. 26, 1995, pp. 833–836.

[14] Y.-H. Fung and Y.-H. Chan, “Tone-dependent error diffusion based on an updated blue-noise model,” *J. of Electron. Imaging*, vol. 25, no. 1, p. 013 013, 2016.

[15] T.-C. Chang and J. P. Allebach, “Memory efficient error diffusion,” *IEEE Trans. Image Processing*, vol. 12, no. 11, pp. 1352–1366, 2003.

[16] P. Li and J. P. Allebach, “Block interlaced pinwheel error diffusion,” *J. of Electron. Imaging*, vol. 14, no. 2, p. 023 007, 2005.

[17] P. Li and J. P. Allebach, “Tone-dependent error diffusion,” *IEEE Trans. Image Processing*, vol. 13, no. 2, pp. 201–215, 2004.

[18] S. W. Han, “Training based error diffusion and halftone quality quantification,” Ph.D. dissertation, Dept. Elect. Comput. Eng., Purdue Univ., West Lafayette, IN, USA, 2009.

[19] J. F. Jarvis, C. N. Judice, and W. Ninke, “A survey of techniques for the display of continuous tone pictures on bilevel displays,” *Computer Graphics and Image Processing*, vol. 5, no. 1, pp. 13–40, 1976.

[20] J.-N. Shiau and Z. Fan, “Set of easily implementable coefficients in error diffusion with reduced worm artifacts,” in *Proc. SPIE Color Imaging: Device-Independent Color, Color Hard Copy, and Graphic Arts*, International Society for Optics and Photonics, vol. 2658, 1996, pp. 222–226.

[21] R. Nasanen, “Visibility of halftone dot textures,” *IEEE Trans. Syst. Man and Cybern. Syst.*, vol. 14, no. 6, pp. 920–924, 1984.

[22] V. Torczon, “On the convergence of pattern search algorithms,” *SIAM J. optimization*, vol. 7, no. 1, pp. 1–25, 1997.

[23] J. P. Allebach, “DBS: retrospective and future directions,” in *Proc. SPIE Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts VI*, vol. 4300, 2001, pp. 358–376.

[24] P. G. Roetling and T. M. Holladay, “Tone reproduction and screen design for pictorial electrophotographic printing,” *J. Appl. Photo. Eng.*, vol. 15, no. 4, pp. 179–182, Sep. 1979.

[25] T. N. Pappas and D. L. Neuhoff, “Model-based halftoning,” in *Proc. SPIE Human Vision, Visual Processing, and Digital Display II*, vol. 1453, 1991, pp. 244–255.

[26] T. N. Pappas and D. L. Neuhoff, “Printer models and error diffusion,” *IEEE Trans. Image Processing*, vol. 4, no. 1, pp. 66–80, 1995.

[27] J. P. Allebach, “Binary display of images when spot size exceeds step size,” *J. Appl. Opt.*, vol. 19, no. 15, pp. 2513–2519, 1980.

[28] P. Stucki, "MECCA–A multiple-error correcting computation algorithm for bilevel image hardcopy reproduction," IBM Research Laboratory, Zurich, Switzerland, Res. Rep. RZ1060, 1981.

[29] P. Stucki, "Advances in digital image processing for document reproduction," in *VLSI Engineering: Beyond Software Engineering*, T. L. Kunii, Ed., vol. 163, 1984, pp. 256–302.

[30] R. L. Stevenson and G. R. Arce, "Binary display of hexagonally sampled continuous-tone images," *J. Opt. Soc. Amer. A*, vol. 2, no. 7, pp. 1009–1013, July 1985.

[31] T. N. Pappas and D. L. Neuhoff, "Least-squares model-based halftoning," in *Proc. SPIE Human Vision, Visual Processing, and Digital Display III*, vol. 1666, 1992, pp. 165–176.

[32] T. N. Pappas and D. L. Neuhoff, "Least-squares model-based halftoning," *IEEE Trans. Image Processing*, vol. 8, no. 8, pp. 1102–1116, 1999.

[33] M. Analoui and J. P. Allebach, "Model-based halftoning using direct binary search," in *Proc. SPIE Human Vision, Visual Processing, and Digital Display III*, vol. 1666, 1992, pp. 96–108.

[34] F. A. Baqai and J. P. Allebach, "Halftoning via direct binary search using analytical and stochastic printer models," *IEEE Trans. Image Processing*, vol. 12, no. 1, pp. 1–15, 2003.

[35] T. N. Pappas, C.-K. Dong, and D. L. Neuhoff, "Measurement of printer parameters for model-based halftoning," *J. Electron. Imaging*, vol. 2, no. 3, pp. 193–204, 1993.

[36] J.-H. Lee and J. P. Allebach, "Inkjet printer model-based halftoning," *IEEE Trans. Image Processing*, vol. 14, no. 5, pp. 674–689, 2005.

[37] S. Wang, "Aerodynamic effect on inkjet main drop and satellite dot placement," in *Proc. Int. Conf. Digit. Printing Technol.*, 1998, pp. 5–8.

[38] P. D. Fleming, J. E. Cawthorne, F. Mehta, S. Halwawala, and M. K. Joyce, "Interpretation of dot fidelity of ink jet dots based on image analysis," *J. Imaging Sci. Technol.*, vol. 47, no. 5, pp. 394–399, 2003.

[39] Y. Ju, T. Kashti, T. Frank, D. Kella, D. Shaked, M. Fischer, R. Ulichney, and J. P. Allebach, "Black-box models for laser electrophotographic printers– Recent progress," in *Proc. 29th Int. Conf. Digit. Printing Technol.*, 2013, pp. 66–71.

[40] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 9, no. 1, pp. 62–66, Jan. 1979.

[41] R. Näsänen, “Visibility of halftone dot textures,” *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 14, no. 6, pp. 920–924, 1984.

[42] C. Lee, “Hybrid screen design and automatic portrait image enhancement,” Ph.D. dissertation, Dept. Elect. Comput. Eng., Purdue Univ., West Lafayette, IN, USA, Dec. 2008.

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

[44] F. Ruckdeschel and O. Hauser, “Yule-Nielsen effect in printing: A physical analysis,” *Applied Optics*, vol. 17, no. 21, pp. 3376–3383, 1978.

[45] J. Yule and W. Nielsen, “The penetration of light into paper and its effect on halftone reproduction,” in *Proc. TAGA*, vol. 3, 1951, pp. 65–76.

[46] R. P. Loce, W. L. Lama, and M. S. Maltz, “Modeling vibration-induced halftone banding in a xerographic laser printer,” *J. Electron. Imaging*, vol. 4, no. 1, pp. 48–62, 1995.

[47] D. Kacker, T. Camis, and J. P. Allebach, “Electrophotographic process embedded in direct binary search,” *IEEE Trans. Image Processing*, vol. 11, no. 3, pp. 243–257, 2002.

[48] P. A. Torpey, “Multipass printing in an ink-jet device,” in *Proc. SPIE Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts II*, vol. 3018, 1997, pp. 344–347.

[49] A. D. Parkhurst, R. Padmanabhan, S. D. Mueller, and K. A. Winter, “Connectivity of the hp deskjet 1200c printer,” *Hewlett Packard Journal*, vol. 45, pp. 85–85, Feb. 1994.

[50] J. Yen, M. Carlsson, M. Chang, J. M. Garcia, and H. Nguyen, “Constraint solving for inkjet print mask design,” *J. Imaging Sci. Technol.*, vol. 44, no. 5, pp. 391–397, 2000.

[51] J. W. Boley, J. P. Allebach, and G. T.-C. Chiu, “Direct binary search for print mask design in inkjet printing,” in *Proc. Int. Conf. Digit. Printing Technol.*, vol. 2011, 2011, pp. 616–619.

[52] “Technical white paper HP PageWide Technology,” HP Inc. USA, 4AA4-4292ENUS, 2012. [Online]. Available: http://www.hp.com/hpinfo/newsroom/press_kits/2012/FallBizPrinting/HPPageWideTechnologyWhitePaper.pdf.

[53] N. Otsu, “A threshold selection method from gray-level histograms,” *IEEE Trans. Syst. Man Cybern. Syst.*, vol. 9, no. 1, pp. 62–66, 1979.

[54] S. A. Gindi, “Color characterization and modeling of a scanner,” Masters thesis, Dept. Elect. Comput. Eng., Purdue Univ., West Lafayette, IN, USA, July 2008.

[55] M. A. Stephens, “EDF statistics for goodness of fit and some comparisons,” *J. Opt. Soc. Amer. A*, vol. 69, no. 347, pp. 730–737, 1974.

[56] N. M. Razali and B. W. Yap, “Power comparisons of Shapiro-Wilk, Kolmogorov-Smirnov, Lilliefors and Anderson-Darling tests,” *J. Stat. Model. Analytics*, vol. 2, pp. 21–33, 2011.

[57] S. H. Kim and J. P. Allebach, “Impact of HVS models on model-based halftoning,” *IEEE Trans. Image Processing*, vol. 11, no. 3, pp. 258–269, 2002.

[58] Y. Mao, L. Abello, U. Sarkar, R. Ulichney, and J. Allebach, “4-row serpentine tone dependent fast error diffusion,” in *Proc. of 2018 25th IEEE International Conference on Image Processing*, 2018, pp. 3973–3977.

[59] W. Jiang, “Color halftoning based on Neugebauer primary area coverage and novel color halftoning algorithm for ink savings,” Ph.D. dissertation, Dept. Elect. Comput. Eng., Purdue Univ., West Lafayette, IN, USA, May 2019.

[60] M. Bevilacqua, A. Roumy, C. Guillemot, and M. L. Alberi-Morel, “Low-complexity single-image super-resolution based on nonnegative neighbor embedding,” *Proc. British Machine Vision Conference*, 2012.

[61] R. Zeyde, M. Elad, and M. Protter, “On single image scale-up using sparse-representations,” in *Proc. Springer International Conference on Curves and Surfaces*, 2010, pp. 711–730.

[62] X. Dong, K.-L. Hua, P. Majewicz, G. McNutt, C. A. Bouman, J. P. Allebach, and I. Pollak, “Document page classification algorithms in low-end copy pipeline,” *J. of Electron. Imaging*, vol. 17, no. 4, p. 043 011, 2008.

[63] H. Cheng and C. A. Bouman, “Document compression using rate-distortion optimized segmentation,” *J. of Electron. Imaging*, vol. 10, no. 2, pp. 460–475, 2001.

[64] R. L. de Queiroz, “Compression of compound documents,” in *Proc. 1999 International Conference on Image Processing*, vol. 1, 1999, pp. 209–213.

[65] S. J. Simske and S. C. Baggs, “Digital capture for automated scanner workflows,” in *Proc. of the 2004 ACM Symposium on Document Engineering*, 2004, pp. 171–177.

[66] W. Wang, I. Pollak, T.-S. Wong, C. A. Bouman, M. P. Harper, and J. M. Siskind, "Hierarchical stochastic image grammars for classification and segmentation," *IEEE Trans. Image Processing*, vol. 15, no. 10, pp. 3033–3052, 2006.

[67] A. Das, S. Roy, U. Bhattacharya, and S. K. Parui, "Document image classification with intra-domain transfer learning and stacked generalization of deep convolutional neural networks," in *Proc. of 2018 24th International Conference on Pattern Recognition (ICPR)*, 2018, pp. 3180–3185.

[68] C. Lu, J. Wagner, B. Pitta, D. Larson, and J. P. Allebach, "SVM-based automatic scanned image classification with quick decision capability," in *Proc. SPIE Color Imaging XIX: Displaying, Processing, Hardcopy, and Applications*, vol. 9015, 2014, 90150G.

[69] S. Xu, C. Lu, M. Shaw, P. Bauer, and J. P. Allebach, "Page classification for print imaging pipeline," in *Proc. SPIE Color Imaging XXII: Displaying, Processing, Hardcopy, and Applications*, vol. 2017, 2017, pp. 137–142.

[70] C. Schmid, J. L. Stoffel, and B. Sperry, *Ink compositions for use in highlighter markers and associated methods*, US Patent 8,007,096, Aug. 2011.

[71] P. Pudil, J. Novovičová, and J. Kittler, "Floating search methods in feature selection," *Pattern Recognition Letters*, vol. 15, no. 11, pp. 1119–1125, 1994.

[72] J. C. Platt, N. Cristianini, and J. Shawe-Taylor, "Large margin DAGs for multiclass classification," in *Proc. of the 12th International Conference on Neural Information Processing Systems*, MIT Press, 1999, pp. 547–553.

[73] G. Paschos, "Perceptually uniform color spaces for color texture analysis: An empirical evaluation," *IEEE Trans. Image Processing*, vol. 10, no. 6, pp. 932–937, 2001.

[74] G. Wyszecki and W. Stiles, *Color Science: Concepts and Methods, Quantitative Data and Formulae.* New York, USA: Wiley, 1982.

[75] M. Shaw, "Gamut estimation using 2D surface splines," in *Proc. SPIE Color Imaging XI: Processing, Hardcopy, and Applications*, vol. 6058, 2006, p. 605 807.

[76] R. L. Graham, "An efficient algorithm for determining the convex hull of a finite planar set," *Info. Pro. Lett.*, vol. 1, pp. 132–133, 1972.

[77] W. Kress and M. Stevens, "Derivation of 3-dimensional gamut descriptors for graphic arts output devices," in *Proc. of the Technical Association of the Graphic Arts*, 1994, pp. 199–199.

[78] J. F. Whitney and H. M. Whitney, "The right-hand rule," in *A Handbook of Mathematical Methods and Problem-Solving Tools for Introductory Physics*, ser. 2053-2571, Morgan & Claypool Publishers, 2016, 7-1 to 7–3.

[79] B. Kolman and D. Hill, *Elementary Linear Algebra with Applications*. Boston MA: Pearson, 2007.

[80] M. A. Stricker and M. Orengo, "Similarity of color images," in *Storage and Retrieval for Image and Video Databases III*, vol. 2420, 1995, pp. 381–392.

[81] D. L. Olson and D. Delen, *Advanced Data Mining Techniques*. Berlin Heidelberg: Springer, 2008.

[82] N. Bhatti, H. Baker, H. Chao, S. Clearwater, M. Harville, J. Jain, N. Lyons, J. Marguier, J. Schettino, and S. Süsstrunk, "Mobile cosmetics advisor: An imaging based mobile service," in *Proc. SPIE Multimedia on Mobile Devices 2010*, vol. 7542, 2010.

[83] W.-S. Tong, C.-K. Tang, M. S. Brown, and Y.-Q. Xu, "Example-based cosmetic transfer," in *15th Pacific Conference on Computer Graphics and Applications (PG'07)*, 2007, pp. 211–218.

[84] T. V. Nguyen and L. Liu, "Smart mirror: Intelligent makeup recommendation and synthesis," in *Proc. of the 25th ACM International Conference on Multimedia*, 2017, pp. 1253–1254.

[85] T. Alashkar, S. Jiang, S. Wang, and Y. Fu, "Examples-rules guided deep neural network for makeup recommendation," in *Proc. of the AAAI Conference on Artificial Intelligence*, vol. 31, 2017.

[86] J. Li, C. Xiong, L. Liu, X. Shu, and S. Yan, "Deep face beautification," in *Proc. of the 23rd ACM International Conference on Multimedia*, 2015, pp. 793–794.

[87] D. Cheng, D. K. Prasad, and M. S. Brown, "Illuminant estimation for color constancy: Why spatial-domain methods work and the role of the color distribution," *J. Opt. Soc. Amer. A*, vol. 31, no. 5, pp. 1049–1058, 2014.

[88] J. T. Barron and Y.-T. Tsai, "Fast fourier color constancy," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 886–894.

[89] Y. Hu, B. Wang, and S. Lin, "FC4: Fully convolutional color constancy with confidence-weighted pooling," in *Proc. of the IEEE Conference on Computer Vision and Pattern Recognition*, 2017, pp. 4085–4094.

[90] P.-C. Hung, "Colorimetric calibration in electronic imaging devices using a look-up-table model and interpolations," *J. of Electron. Imaging*, vol. 2, no. 1, pp. 53–62, 1993.

[91] C. Zhao, J. Niu, G. Li, H. Wang, and C. He, "Facial color management for mobile health in the wild," *IEEE Trans. NanoBioscience*, vol. 15, no. 4, pp. 316–327, 2016.

[92] V. Cheung and S. Westland, "Color camera characterisation using artificial neural networks," in *Proc. of 10th Color Imaging Conference: Color Science and Engineering Systems, Technologies, Applications*, vol. 2002, 2002, pp. 117–120.

[93] H. R. Kang and P. G. Anderson, "Neural network applications to the color scanner and printer calibrations," *J. of Electron. Imaging*, vol. 1, no. 2, pp. 125–136, 1992.

[94] R. S. Berns and M. J. Shyu, "Colorimetric characterization of a desktop drum scanner using a spectral model," *J. of Electron. Imaging*, vol. 4, no. 4, pp. 360–373, 1995.

[95] G. D. Finlayson and M. S. Drew, "Constrained least-squares regression in color spaces," *J. of Electron. Imaging*, vol. 6, no. 4, pp. 484–494, 1997.

[96] H. Drucker, C. J. Burges, L. Kaufman, A. Smola, and V. Vapnik, "Support vector regression machines," *Advances in Neural Information Processing Systems*, vol. 9, pp. 155–161, 1996.

[97] N. A. A. Rahman, K. C. Wei, and J. See, "RGB-H-CbCr skin colour model for human face detection," *Proc. of Mulitimedia University International Symposium on Information & Communications Technologies*, 2006.