DIGITAL IMAGE PROCESSING AND MACHINE LEARNING RESEARCH: DIGITAL COLOR HALFTONING, PRINTED IMAGE ARTIFACT DETECTION AND QUALITY ASSESSMENT, AND IMAGE DENOISING

by

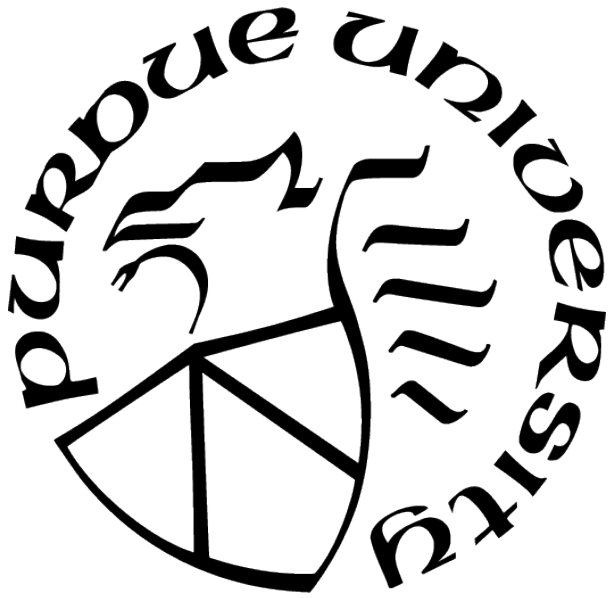
Yi Yang

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

May 2022

THE PURDUE UNIVERSITY GRADUATE SCHOOL
STATEMENT OF COMMITTEE APPROVAL

Prof. Jan P. Allebach, Chair

School of Electrical and Computer Engineering

Prof. George T. Chiu

School of Mechanical Engineering

Prof. Fengqing M. Zhu

School of Electrical and Computer Engineering

Prof. Mary L. Comer

School of Electrical and Computer Engineering

Approved by:

Dr. Dimitrios Peroulis

To my family.

ACKNOWLEDGMENTS

I would like to express my gratitude to every family member, teacher, and friend who has supported, guided and assisted me throughout my life. The words that follow will never be able to adequately express my heartfelt gratitude.

First and foremost, I would like to thank my advisor, Prof. Allebach, for his encouraging words, professional advice, and unwavering patience. Joining his group not only provided me with the opportunity to conduct exciting research, but also made me a better person. I will always consider myself lucky to have had the opportunity to learn under Prof. Allebach.

Next, I would like to thank the members of my Ph.D. advisory committee, Prof. Chiu George T. C., Prof. Comer Mary L., and Prof. Zhu Fengqing M., for their assistance and insightful comments. I would also like to thank every teacher who has taught and assisted me throughout my academic life.

Furthermore, I would like to express my gratitude to my family members: Tianrui, Xiulan, Yufang, Taishan, Zhao, and Ruiyi. When I decided to pursue my Ph.D. in the USA and I've never spent enough time with them since. However, their unconditional love and unwavering support inspired me to embark on this adventure. Many thanks, and I love you all!

Additionally, I would like to thank my friends for their support and friendship, as well as my EISL labmates. Thank you for enriching my life.

Finally, I would like to thank Dr. Shaw Mark, Dr. Maggard Eric, Dr. Peng Liang, and Dr. Chou Chih-Hsien for their helpful discussions and feedback during my internships at HP Inc. and Futurewei Inc.

TABLE OF CONTENTS

LIST OF FIGURES

ABBREVIATIONS

HVS	Human Visual System
DBS	Direct Binary Search
PARAWACS	Parallel Random Weighted Area Coverage Selection
IQ	Image Quality
IQA	Image Quality Assessment
CSF	Contrast Sensitivity Function
JND	Just Noticeable Difference
MSE	Mean Square Error
MAE	Mean Absolute Error
RMS	Root Mean Square
RMSE	Root Mean Square Error
SSIM	Structural Similarity Index
PSNR	Peak Signal-to-Noise Ratio
IJ	InkJet
SVM	Support Vector Machines
FM	FrequencyModulation
AM	Amplitude Modulation
DPI	Dots Per Inch
PPI	Pixel Per Inch
INCITS	The International Committee for Information Technology Standards
ROI	Region Of Interest
RANSAC	Random Sample Consensu
MLESAC	Maximum Likelihood Estimation Sample Consensus
SSD	Sum of Squared Differences
CIE	International Commission on Illumination
ANOVA	Analysis of Variance
CPR	Color Plane Misregistration
AWGN	Additive White Gaussian Noise

CNNs/ConvNets	Convolutional Neural Networks
DCT	Discrete Cosine Transform
LUT	Look-Up Table

COLOR SPACE:

CMYK	Cyan, Magenta, Yellow, Black
sRGB	standard Red Green Blue
HSL	Hue, Saturation, Lightness
CIE XYZ	
CIE LAB	

ABSTRACT

In the thesis, we study several problems related to digital image processing and machine learning.

Digital halftoning is an essential preprocessing step in printing. It is used to represent a continuous-tone image as an image that contains a limited number of colorants and with minimal loss of image quality. The primary goal of digital color halftones is to reproduce the closest possible color representation to the original image. To begin with, we describe a project in which three screens for Cyan, Magenta, and Yellow colorants were designed jointly using the Direct Binary Search algorithm (DBS). The screen set generated by the algorithm can be used to halftone color images easily and quickly. The halftoning results demonstrate that by utilizing the screen sets, it is possible to obtain high-quality color halftone images while significantly reducing computational complexity.

The demands for printing speed and quality are increasing as imaging technology advances. Our next research focuses on defect detection and quality assessment of printed images, which are critical for designing and improving high-quality printing systems. We measure and analyze macro-uniformity, banding, and color plane misregistration, which are thought to have the greatest influence on printed image quality. For these three defects, we designed different pipelines for them and developed a series of digital image processing and computer vision algorithms for the purpose of quantifying and evaluating these printed image defects. Additionally, we conduct a human psychophysical experiment to collect perceptual assessments and use machine learning approaches to predict image quality scores based on human vision.

Due to the limitations of various recording devices, images are sensitive to random noise during acquisition. Noise is a signal distortion that impedes image observation and information extraction. Thus, as a fundamental topic of image analysis and processing, image noise suppression aids our understanding of image statistics and processing. We study modern deep convolutional neural networks for image denoising and focus on blind, bias-removed, mix loss optimized, and perceptually oriented image denoising tasks. We propose a network designed for AWGN image denoising. Our network removes the bias at each layer to achieve

the benefits of scaling invariant network; additionally, it implements a mix loss function to boost performance. We train and evaluate our denoising results using PSNR, SSIM, and LPIPS, and demonstrate that our results achieve impressive performance evaluated with both objective and subjective IQA metrics.

1. INTRODUCTION

In this thesis, we study several problems related to digital image processing and machine learning. Our research is concentrated on the following areas:

- Color digital halftones.
- Print image macro-uniformity prediction.
- Print image defect detection and quality analysis.
- Image denoising.

Chapter 2 of this thesis will focus on the design and development of joint FM screen sets for halftone color images. Digital color halftoning represents a continuous-tone color image with a limited number of fixed-tone inks. When viewed from a distance, the halftone image produces the same visual perception as the continuous image due to the low-pass nature of the human eye. We discuss how we use the Direct Binary Search algorithm (DBS) to design and build plane-dependent Frequency Modulation (FM) joint screen sets. The algorithm generates a set of screens that can be easily and quickly used to halftone color images. We demonstrate that our results are of higher quality than those obtained using traditional screening halftoning, as well as being easier to use and requiring less computational complexity than those obtained using direct color DBS halftoning.

In Chapters 3–5, we will discuss measurement and assessment of image quality, particularly for printed images. We detect and quantify print image artifacts, assess macro-uniformity, and assess image quality both objectively and subjectively.

Chapter 3 will delve into the macro-uniformity of printed images, one of the most critical factors affecting the overall visual perception of prints. While quantitative analysis of a single artifact is possible, due to the nonlinear nature of the human eye's response, it is difficult to assess the overall perceived quality when multiple artifacts are present simultaneously. We conduct psychophysical experiments to pool perceptual assessments of our printed test images, using the International Council for Information Technology Standards (INCITS)

W1.1-designed macro-consistency quality scale as our experimental reference. Then, we compute artifact features that characterize the severity of artifacts in our test sample, and we use these features to train a predictive model. On the basis of human judgment, the predictor can automatically calculate the macro-uniformity score in JND units. Our results show that the predictor is accurate and that the predicted score corresponds to the subjective visual score (ground-truth). Thus, our predictor helps researchers gain a better understanding of the perceptual macro-uniformity of print quality without conducting subjective experiments.

In Chapters 4 and 5, we will discuss the detection and measurement of two important artifacts in printed images: banding and color plane misregistration. These two artifacts are considered to be among the most serious and common defects affecting the overall image quality within the printing industry. There has been some research on it before, but most have focused on measuring them on uniform color pages or specific test images. We propose measurement and detection methods that directly operate effectively on customer content pages. Therefore, it can be more convenient and widely used than the previous method. Also, we build artifact classification and quality predictors using machine learning methods.

In Chapter 6, we will discuss deep convolutional neural networks for image denoising. Deep learning has been extensively studied in a variety of aspects of image denoising, including blind, unsupervised, universal, and perceptually oriented. Our research focuses on blind, bias-removed, mix loss optimized, and perceptually oriented image denoising task. We propose a network designed for AWGN image denoising. Our network removes the bias at each layer to achieve the benefits of scaling invariant network. Additionally, it implements a mix loss function to boost performance. We train and evaluate our denoising results using PSNR, SSIM, and the perceptual metric LPIPS, and demonstrate that our results achieve impressive performance evaluated with both objective and subjective IQA metrics.

Finally, in Chapter 7, we will summarize the contributions of this work.

2. JOINT DESIGN OF PLANE-DEPENDENT FM SCREENS SETS USING DBS ALGORITHM

2.1 Background and Previous Work Review

2.1.1 Digital Halftoning

The majority of image reproduction devices, especially printing devices, are limited to few colors, while digital images can consist of millions of colors. Therefore, when printing a digital image, the process of transforming a digital image (also called a continuous-tone image) that contains a full range of values between 0 and 255, to a representation that is either binary, 0 (white hole or black dot) or 255 (black dot or white hole), or that contains only a small number of levels between 0 and 255, is digital halftoning [1], (For simplicity, we will only consider binary output devices in our work). Digital halftoning is an especially critical step in the imaging pipeline for most printing technologies.

2.1.2 AM and FM Halftoning

Adjusting the tone value by changing the size of black dots is called Amplitude Modulation (AM) halftoning. An increase in the size of a single black dot means that the tone value becomes darker, and a decrease in size means that the tone value becomes brighter.

Adjusting the tone value by keeping the size of the black dots fixed but changing the spacing between black dots is called Frequency Modulation (FM) halftoning. Both AM and FM can indicate tone value by computing the total area of black dots in a halftone cell.

Many previous halftone research works based on AM and FM can be found in the references [1],[2], [3] and [4]. AM and FM techniques have their advantages and limitations. For a long time, AM halftoning has been the most commonly used halftone technology in the printing industry. The main reason is that the printing device cannot produce small single micro-dots. But later, FM halftoning began to compete with AM halftoning because it can reduce printing costs. FM halftoning works better when dealing with prints with sharp transitions between the tone values, such as heavy texture and image details. Moreover, when the mechanical limitations of the printing press limit the screen frequency from reaching

the desired high frequency, FM halftoning is far superior to AM halftoning. So now, FM halftoning is more widely used in the printing industry [1].

2.1.3 Color Halftoning

Digital color halftoning is an essential pre-processing of printing, which represents a continuous-tone color image in a limited amount of fixed-tone inks. Although at first glance one could think that the color halftoning only means repeating the same monochrome halftoning multiple times in different color channels, some problems arise in the process of color representation, such as color artifacts and poor color rendition when this approach is taken.

To thoroughly study color halftoning, we need a comprehensive understanding of colors and the correlation between color or colorant planes, which is beyond the scope of this thesis. Therefore, my thesis only focuses on those parts that help to understand the color halftoning. Also, in this thesis, the print mechanism places colorant dots, for example, Cyan (C), Magenta (M), or Yellow (Y) for three-color printers, on the media and the perceived colors are the results of subtractive interaction between these colorants.

2.1.4 Dependent FM Color Halftoning

The most common printer color channels are Cyan (C), Magenta (M), and Yellow (Y). FM halftoning technology can be practiced for each color channel of a color image. Normally, the color channels can be halftoned independently, but the image quality will be largely improved by conjunctively applying the halftoning algorithm to different color channels, which is also described as halftoning in a dependent mode. Previous studies [5] show that dependent halftoning can reduce color noise. It has also been shown that dependent halftoning can improve print quality, and relatively reduce the ink requirement.

2.1.5 Color Halftoning Methods

Digital halftoning algorithms can be broadly classified into one of three groups [7], according to their computational complexity.

In the first group, known as *screening* [8], for each colorant plane, the binary value in the halftone pattern at each pixel is determined by comparing the colorant amount at that pixel with a spatially varying threshold. The matrix of these thresholds is called the screen. Thus, the halftone value at each pixel only depends on the colorant value at that pixel location. This approach is very widely used in printers due to its computational simplicity.

At the next level of complexity, we have methods in the second group, which are based on *error diffusion*[9]. Here, the binary value of each colorant is determined by thresholding a colorant amount that has been modified by the diffusion of errors made at previously binarized pixels. Thus, the halftone value at a given pixel depends on the values of the continuous-tone input in a causal neighborhood of the current pixel. Despite its increased computational complexity, error diffusion is also widely used in printers.

Finally, the halftoning methods in the third group are *search-based methods* [7] and [10]. These methods make several passes through the image to minimize a metric of the perceived error between the continuous-tone and halftone images. While such methods can arguably yield the best overall image quality, their computational complexity has thus far prevented their use in printers to directly halftone images. However, these methods have been frequently used as the basis for design of the parameters for simpler methods, such as screening or error diffusion that are used in printers. The Direct Binary Search or DBS algorithm is perhaps the best-known of these methods [7] and [10].

2.1.6 Screening

In this chapter, we shall only consider halftoning by screening. In particular, for color printers, we assume that the printer independently processes each colorant plane by performing comparisons pixel-by-pixel with a screen designed specifically for that colorant plane. This architecture is a limitation imposed by many printers, ranging from low-end devices to very high-end digital presses used for commercial printing. Although the work reported in this chapter is only targeted to halftoning by screening, it is important to note that there are other variants in this class that appear promising. In particular, the recently proposed PARAWACS algorithm explicitly controls the placement of Neugebauer primaries

on a pixel-by-pixel basis [11] and [12], via a single *selection matrix* that is analogous to a screen. However, this method fails the test of independently processing each colorant plane to determine the halftone output at each pixel.

In addition, we only consider 3-color printers that can place a dot of one or more of the three colorants C, M, or Y at each pixel. Such devices are also commercially important. At the low-end, this restriction lowers the cost of the printer mechanism. At the high-end, the use of ink can be reduced, and the throughput is increased, by not using the black (K) colorant, even when it is available. This reduces the overall cost of a print job.

Since the design of the screens is an offline process, we are free to use extensive computational resources in the design process. Here, we use DBS to jointly design a set of three screens. The halftoned image using the jointly designed screens set has better quality than the halftoned image using independently designed screens.

Joint screen design using DBS was first proposed in references [2] and [3]. The authors introduced a cost metric that was based on a full spatiochromatic model for the human visual system[13] and [14]. The metric was quite general in that at any level, it could account for quality of the halftone patterns consisting of any desired subset of the constituent Neugebauer primaries comprising the composite halftone pattern. Then, references [5] and [15] proposed a different strategy for color halftoning based on DBS. Here, the cost function was simply based on Nasenen's monochrome human visual system response [6] and [16]; and the interaction of colorants was controlled by constraints that varied based on the target color value. However, these two references did not propose a screen design method, and only considered joint design of the C and M halftone patterns.

In the present chapter, we propose a method for joint design of CMY color screen sets based on DBS that is philosophically most similar to the approach taken in references [5] and [15]. In addition to proposing a detailed process for screen design, which is not given by above references, the present chapter explicitly considers all three colorants C, M, and Y in the design process, and also more completely controls the printing of secondary colors and composite black (CMY) across the tone scale than do references [2] and [3].

2.2 Color DBS Halftoning Algorithm

The color DBS halftoning algorithm is an excellent algorithm in the category of iterative methods to do color halftoning. The purpose of the color DBS halftoning algorithms is to iteratively modify the halftone to minimize the perceived difference between the continuous-tone color image and its halftoned image.

The overall idea is to start with an initial halftone, apply the human visual filter to the continuous-tone original image and the initial halftoned image, compute the error metric based on their perceived difference, and sequentially visit each pixel of the halftone image, by swapping two different status pixels (such as exchange C with M) or toggling a pixel with its opposite-phase pixel (such as change 0 to 1 or change 1 to 0) to find operations that reduce the error metric, and modify the halftone image by accepting operations that reduce the error metric. We then repeat in this manner and make several traverses over the halftone image until the error metric reaches a local minimum or meets certain constraints.

The color DBS halftoning algorithm generally can generate halftoned images with noticeably superior visual quality than the dot processes methods (such as screening) and the neighborhood methods (such as error diffusion). However, this superior performance requires a certain cost.

The color DBS halftoning algorithm usually is much more computationally complex than point processing or neighborhood algorithms. Due to its computational cost, the color DBS halftoning algorithm cannot be realized in current printing devices for real-time halftoning of pages. However, it provides a gold standard for assessing the quality of color halftone images and assist in designing simpler, more efficient pixel-based or neighborhood-based halftoning algorithms.

Our work is based on this foundation. It combines the simplicity and convenience of the pixel-based method with the premium quality of color DBS halftoning. The screens set is jointly designed according to the main idea and algorithm of color DBS halftoning, and the designed screens set can be directly compared with the continuous-tone image, and make the color halftoning become a pixel-based method. Also, the DBS algorithm guarantees the image quality of the final halftoned image.

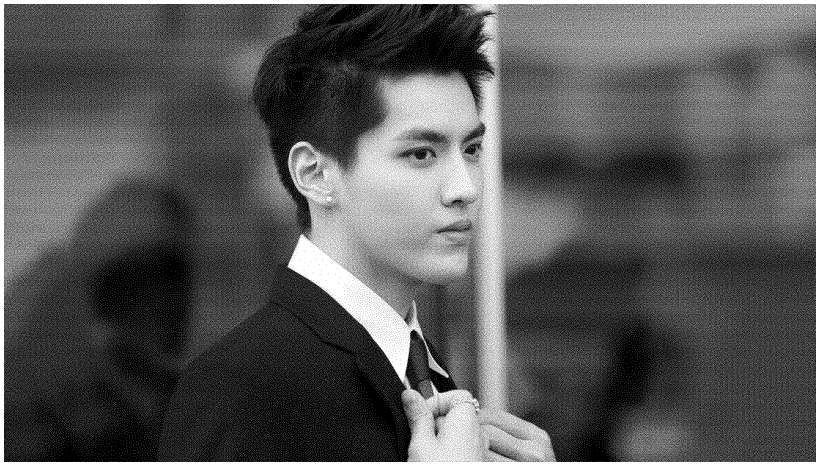


Figure 2.1. An image halftoned by monochrome DBS algorithm. To properly assess image quality, the reader should zoom in until individual dots are visible.

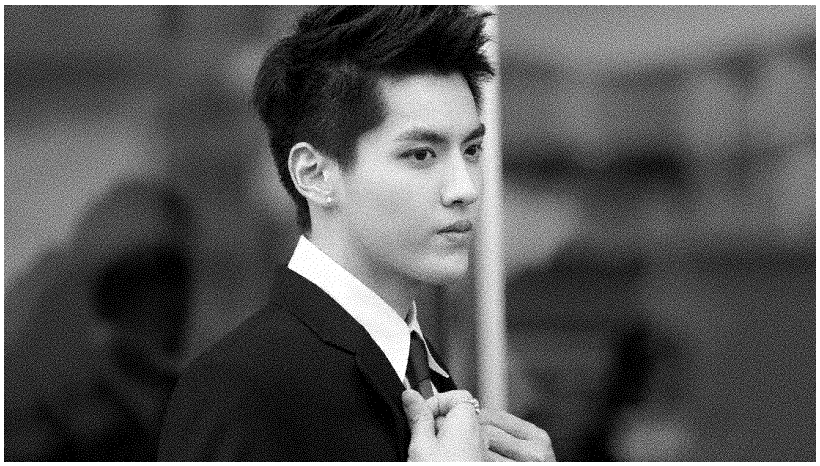


Figure 2.2. The image halftoned by monochrome screening algorithm. Here, the screen was designed using the DBS algorithm, as reported in reference [3]. To properly assess image quality, the reader should zoom in until individual dots are visible.

Figures 2.1 and Figures 2.2 show the comparison of a monochrome DBS halftoned image and a monochrome screen halftoned image. We can clearly see that the halftoned image generated by DBS is more uniform and the distribution of the dots is more smooth.

2.3 Joint Design Algorithm

To design a screen level-by-level, we select a certain level as the starting level, generate an initial pattern, and then apply our DBS-based optimization algorithm to obtain a homogeneous halftone pattern for this level. Then, based on the starting level, we design lighter levels by gradually removing dots and optimizing patterns, and design darker levels by gradually adding dots and optimizing patterns.

In this chapter, the patterns for each level need to satisfy the following conditions [5] and [15] for obtaining homogeneous halftone images.

(1). If we consider dots of each colorant (C, M, or Y) individually, we would like these dots to be arranged as uniformly as possible.

(2). We also would like the overall composite pattern that consists of all three colorants dots to be as uniform as possible.

(3). We need to reduce the dot-on-dot (overlapping dots) as much as possible. That is, keep dot-off-dot as much as possible during the process of screen design.

2.3.1 Optimization Procedure for Designing the Initial Level

We assume that the halftone screens have size $M \times M$, and the total number of levels of the screens is L. For all the examples shown in this chapter, the size of the screens is 256×256, and the number of levels is $L = 256$. To simplify the notation, and without loss of generality, we will assume that the number of levels L is divisible by 3. If this requirement is not met, the procedure described below can be executed with minor modifications to the notation.

We consider a CMY color pattern as a combination of three different C, M and Y patterns. To meet the conditions in the preceding paragraph, we propose our algorithm with the following steps:

First, select a starting level. We choose $\frac{1}{3}L$ as the starting level in this chapter since we consider three colorants: C, M and Y. The size of the screen determines the total number of dots for the three colorants (C+M+Y). With starting level $\frac{1}{3}L$, each colorant pattern receives $\frac{256^2}{3}$ dots for a total number of 256^2 dots. We generate an initial pattern based on

the total number of dots (all C, M and Y dots are treated as black dots, and others are treated as white holes). Then, we apply the monochrome DBS algorithm to this pattern to achieve a uniform pattern. However, if the starting level is $\frac{1}{3}L$, every pixel in the initial halftone pattern will have been assigned a colorant dot, either C, M, or Y. Thus, there will be no white holes; and this first stage of optimization is not needed. It will only be required if the initial level is $< \frac{1}{3}L$.

Secondly, we randomly and evenly divide the positions of black dots in the uniform pattern into three same-size patterns representing the C pattern, M pattern and Y pattern, respectively, guaranteeing that the three patterns each have the same number of dots. By this step, we can prevent dot overlap. Especially in lighter levels, this step enforces the three patterns to be totally dot-off-dot. Figure 2.3 indicates the combination of C, M, and Y colors. Figure 2.4 illustrates the difference between a dot-on-dot pattern and a dot-off-dot pattern.

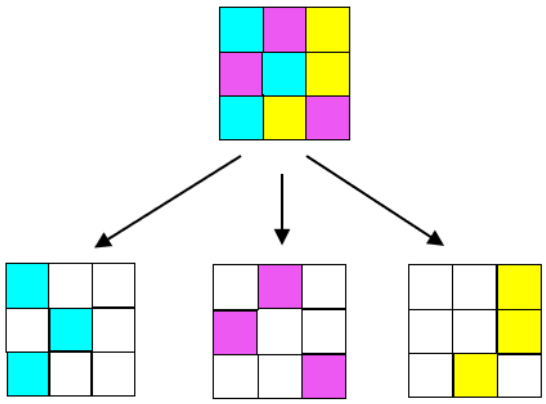


Figure 2.3. A CMY color pattern as a combination of three different C, M and Y patterns.

Then, we apply the swap-only DBS algorithm to iteratively compute the joint $\Delta\epsilon$ between two different patterns, adjusting the position of dots to minimize the error. Since every pixel is assigned one of the three colorants C, M, and Y, the only option is to swap one colorant pixel with another pixel of a different colorant. We start by scanning the C pattern until we find a C dot, then we scan the M pattern until we find an M dot. We swap these two

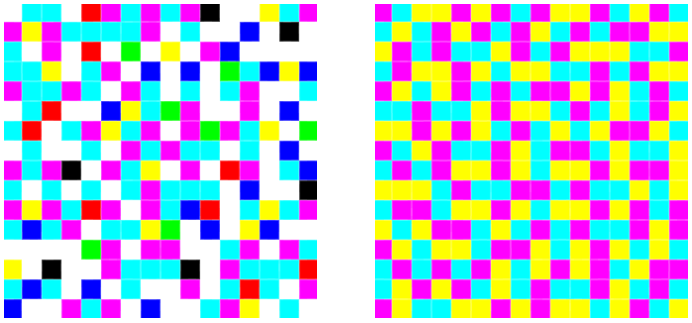


Figure 2.4. The zoomed-in details (16 ×16) of a dot-on-dot pattern and a dot-off-dot pattern consisting of a superposition of Cyan, Magenta, and Yellow dots.

pixels(exchange the position indices of these two pixels), and evaluate the change in error. We repeat this process until we have considered swapping every M pixel with this C pixel. If one or more of these trial swaps decreased the error, we keep the one that decreased the error the most. Otherwise, we don't accept any swaps with this C pixel. Then we go on to the next Cyan pixel, and repeat the process. After we have scanned through the entire halftone pattern, we repeat the process, and scan the halftone pattern again. When no change is accepted during an entire scan of the halftone pattern, this phase of the optimization has converged.

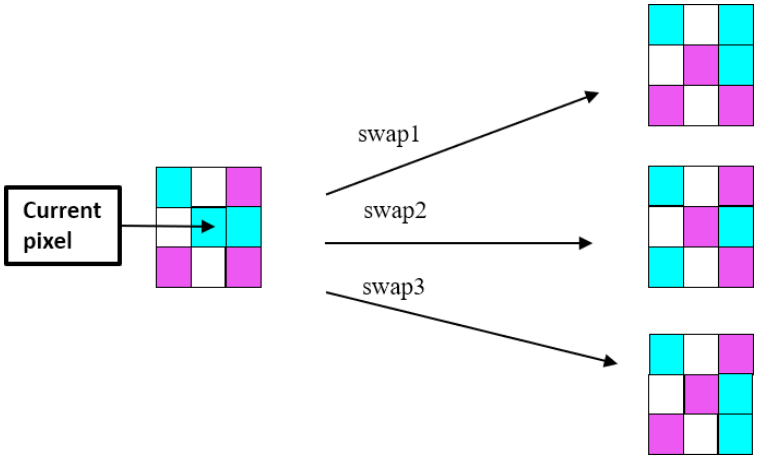


Figure 2.5. Possible swap trials between two colorant: C and M.

Next, we consider swaps between C pixels and Y pixels, following the procedure described in the preceding paragraph. Finally, we consider swaps between M pixels and Y pixels, again

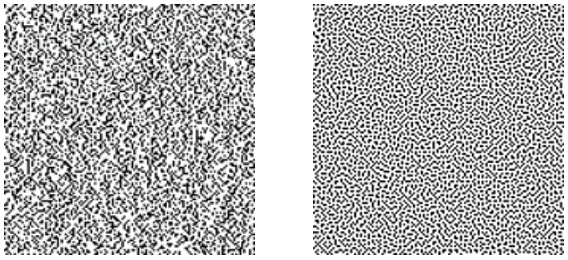


Figure 2.6. The initial and optimized pattern of Cyan.

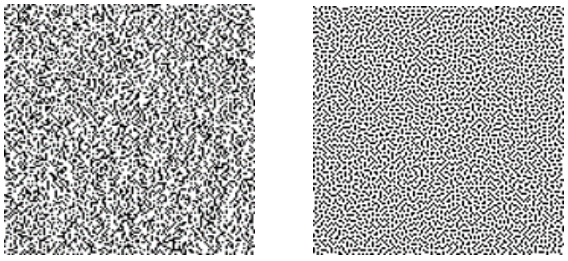


Figure 2.7. The initial and optimized pattern of Magenta.

following the procedure described in the preceding paragraph. Once we have completed these three sequential optimization processes, we repeat the entire cycle: C with M, C with Y, and M with Y. When no changes are kept during an entire cycle, the overall optimization of the starting level of the three halftone patterns is complete.

In summary, the first step ensures that the halftone pattern superimposed by three colorants is uniform, and the next steps ensure that each of the three halftone patterns is uniform. Thus, patterns that satisfy those conditions can be obtained.

To illustrate the pattern uniformity, we present the initial and optimized pattern of C and M in Figure 2.6 and Figure 2.7, the initial and optimized combination of C and M patterns in Figure 2.8, and the entire initial and optimized pattern consisting of the combination of all three colorants in Figure 2.9, which is the uniform starting pattern.

2.3.2 Error Metrics

The Human Visual System (HVS) can be modeled as a linear, shift-invariant system with a point spread function given by the inverse Fourier transform of the contrast sensitivity function of the human viewer [14]. Kim and Allebach compared Nasanen's model with

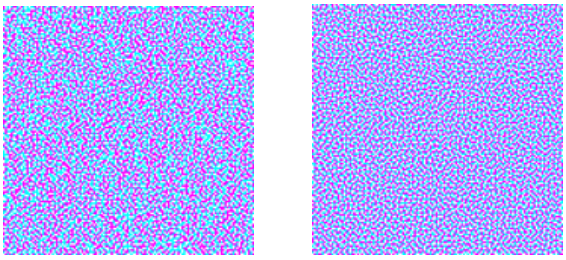


Figure 2.8. The initial and optimized pattern of combination of Cyan and Magenta.

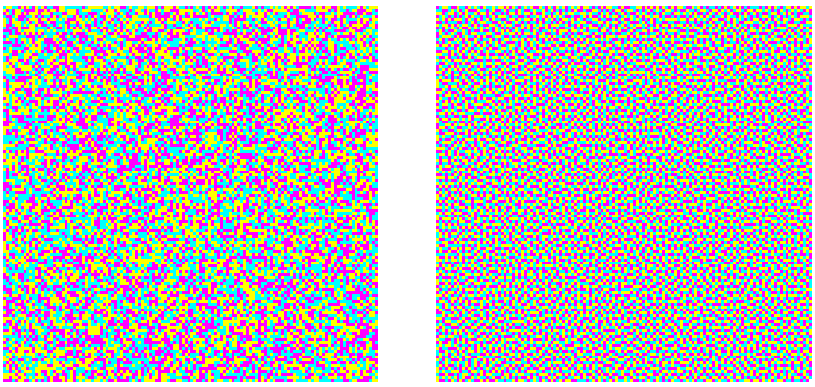


Figure 2.9. The initial and optimized patterns of Cyan, Magenta and Yellow for the level $\frac{1}{3}L$. The reader is advised to zoom in to this pattern to see the individual colorant dots.

three other HVS models and concluded that Nasanen's model gave the best overall halftone quality when incorporated in the DBS algorithm [6]. In this chapter, we use the contrast sensitivity function based on Nasanen's model [16] to filter the continuous tone image and halftone image.

Nasanan's HVS model

For the Nasanan's HVS model [16], the spatial frequency domain response is given by:

$$H_Y(u,v) = a\Gamma^b \exp\left(-\frac{\sqrt{u^2+v^2}}{\lfloor c\ln\Gamma + d\rfloor}\right) \tag{2.1}$$

where Y is the luminance channel, $a = 131.6$, $b = 0.3188$, $c = 0.525$, $d = 3.91$, and Γ is the average luminance of the light reflected from the print in cd/m^2, usually set to 11, and u and v are the spatial frequency coordinates in $cycles/degree$ subtended at the retina.

We use $(\boldsymbol{x}) = (x,y)$ and $[\boldsymbol{m}] = [m,n]$ to denote continuous and discrete spatial coordinates, respectively. Then, we let $f_{\mathrm{i}}(\mathbf{x})$ denote the continuous-tone image, and $g_{\mathrm{i}}[\mathbf{m}]$ denote the halftone images of $f_{\mathrm{i}}(\mathbf{x})$, where, $i = C,\ M,\ Y$ denotes the colorant of the Cyan, Magenta, and Yellow components. We assume $f_{\mathrm{i}}(\mathbf{x})$ takes values between 0 and 1 $(0, \frac{1}{L-1}, \frac{2}{L-1}, ..., 1)$, where L indicates the total number of grey levels mentioned previously. The digital halftone image $g_{\mathrm{i}}[\mathbf{m}]$ takes binary values $\mathbf{0}$ or $\mathbf{1}$.

We consider the patterns of Cyan, Magenta and Yellow to be monochrome, and the contrast sensitivity function based on Näsänen's model is used as the HVS filter. For an ideal printer which produces a square dot, the perceived rendered image $\tilde{g}_{\mathrm{i}}(\mathbf{x})$ is

$$\tilde{g}_{\mathrm{i}}(\mathbf{x}) = \sum_{m} g_{\mathrm{i}}[\mathbf{m}]\tilde{p}(\mathbf{x\text{-}mX}) \tag{2.2}$$

where $\mathbf{X}$ is a diagonal matrix that represents the distance between printer addressable pixels on a rectangular grid, and $\tilde{p(\mathbf{x})} = p_{hvs}(\mathbf{x}) * p_{dots}(\mathbf{x})$ indicates the effect of cascading the printer rendering and HVS models. So the perceived error is the filtered $\mathrm{e}[\mathbf{m}]$, which can be written as

$$\tilde{\mathrm{e}}(\mathbf{x}) = \sum_{\mathbf{m}} \mathrm{e}[\mathbf{m}]\tilde{p}(\mathbf{x\text{-}mX}) \tag{2.3}$$

where $\mathrm{e}[\mathbf{m}] = g[\mathbf{m}] - f[\mathbf{m}]$ is the error image.

In a monochrome halftone image, when we swap two pixels at indices $\mathbf{m_0}$ and $\mathbf{m_1}$, we can write the change in error due to the swap as:

$$\begin{aligned} \Delta\varepsilon =& (a_0^2 + a_1^2)c_{\tilde{p}\tilde{p}}[\mathbf{0}] \\ &+ 2(a_0 c_{\tilde{p}\tilde{\mathrm{e}}}[\mathbf{m_0}] + a_1 c_{\tilde{p}\tilde{\mathrm{e}}}[\mathbf{m_1}]) \\ &- 2(a_0 a_1)c_{\tilde{p}\tilde{p}}[\mathbf{m_1} - \mathbf{m_0}] \end{aligned} \tag{2.4}$$

where

$$c_{\tilde{p}\tilde{\mathrm{e}}}(\mathbf{x}) = \int \tilde{p}(\mathbf{y})\tilde{\mathrm{e}}(\mathbf{y+x})d\mathbf{y} \tag{2.5}$$

$$c_{\tilde{p}\tilde{p}}(\mathbf{x}) = \int \tilde{p}(\mathbf{y})\tilde{p}(\mathbf{y+x})d\mathbf{y} \tag{2.6}$$

and $c_{\tilde{p}\tilde{e}}[\mathbf{m}] = c_{\tilde{p}\tilde{e}}(\mathbf{mX})$, $c_{\tilde{p}\tilde{p}}[\mathbf{m}] = c_{\tilde{p}\tilde{p}}(\mathbf{mX})$.

When $g[\mathbf{m}]$ is changed from 1 to 0, $a_0 = -1$, and when $g[\mathbf{m}]$ is changed from 0 to 1, $a_0 = 1$. The change rule of a_1 is the reverse of that for a_0, so $a_1 = -a_0$.

When we apply a swap-operation between a pixel that is Cyan and a pixel that is Magenta, we consider the dual metric,

$$\Delta\varepsilon = \alpha\Delta\varepsilon_{Cyan} + \beta\Delta\varepsilon_{Magenta} \tag{2.7}$$

We need to ensure that the patterns for the different colorants have the same quality, so we use equal weights:

$$\alpha = \beta = 1$$

Assume that we have an acceptable swap trial when we exchange a Cyan dot ($g_C[\mathbf{m}_0]$) with a Magenta dot ($g_M[\mathbf{m}_1]$). This swap indicates $g_C[\mathbf{m}_0]$ will be changed from 1 to 0, and $g_C[\mathbf{m}_1]$ will be changed from 0 to 1. Meanwhile, $g_M[\mathbf{m}_0]$ will be changed from 0 to 1 and $g_M[\mathbf{m}_1]$ will be changed from 1 to 0.

The pair of swapping dots need to have different binary states. So,

$a_0^C = a_1^M, \quad a_0^C = -a_1^C, \; a_1^M = -a_0^M$

More generally, if we are considering swaps between colorant i and colorant j, where $\mathrm{i}, \mathrm{j} \subset C, M, Y$, then the parameters can be written as,

$a_0^{\mathrm{i}} = a_1^{\mathrm{j}}, \quad a_0^{\mathrm{i}} = -a_1^{\mathrm{i}}, \quad a_0^{\mathrm{j}} = -a_1^{\mathrm{j}}$

where $\mathrm{i}, \mathrm{j} \subset C, M, Y$

Substituting these relationships into Eq. (6), we can get the swap-only dual metric of DBS:

$$\begin{aligned}\Delta\varepsilon =& 2(\alpha+\beta)c_{\tilde{p}\tilde{p}}[\mathbf{0}] - 2(\alpha+\beta)c_{\tilde{p}\tilde{p}}[\mathbf{m_1} - \mathbf{m_0}] \\ &+ 2a_0^{\mathrm{j}}(\alpha c_{\tilde{p}\tilde{e}}^{\mathrm{j}}[\mathbf{m_0}] - \beta c_{\tilde{p}\tilde{e}}^{\mathrm{i}}[\mathbf{m_0}]) \\ &+ 2a_1^{\mathrm{j}}(\alpha c_{\tilde{p}\tilde{e}}^{\mathrm{j}}[\mathbf{m_1}] - \beta c_{\tilde{p}\tilde{e}}^{\mathrm{i}}[\mathbf{m_1}])\end{aligned} \tag{2.8}$$

2.3.3 Design of Levels Below and Above the Initial Level

The process of generating halftone patterns for all levels is not the same. However, all L levels will depend on the uniform starting pattern for $\frac{1}{3}L$ that we designed in the preceding section. The color pattern at this level is full dot-off-dot without any holes or overlaps. Then, we generate lighter levels by gradually removing the dots and generate darker levels by gradually adding dots. The number of dots added or deleted per level is equal for all levels and each colorant. Since we need to keep the stacking constraint, only the newly added or deleted dots can be moved during the DBS optimization, as we design each succeeding level.

To design the first lighter level, we first choose at random M^2/L colorant dots to remove from each of the three starting patterns, which takes us to the next lighter level $\frac{1}{3}L - 1$. Then, we have a composite halftone pattern consisting of an equal number $M^2(1 - \frac{1}{L})$ of C, M, and Y pixels, plus $3M^2/L$ White pixels. At this point, to satisfy the stacking constraint, we may only consider swaps between a given colorant pixel and a White pixel that was previously of the same colorant value as the colorant pixel to be swapped. As before, we scan through all C pixels, considering possible swaps of each C pixel with every White pixel that was previously Cyan. For each C pixel, we retain that swap, if any, that most reduces the error. We repeat this process, scanning through all the C pixels until no swap is retained during an entire iteration through the halftone pattern. We then conduct the same process considering swaps of M pixels with White pixels that were previously Magenta, and then again, considering swaps of Y pixels with White pixels that were previously Yellow. Once this cycle of three sequential optimization processes is complete, we repeat it until no changes are accepted during an entire cycle. Then, the optimization of the first lighter level ($\frac{1}{3}L - 1$) is complete.

To design the second lighter level ($\frac{1}{3}L - 2$), we again choose at random M^2/L colorant dots to remove from each of the three halftone patterns. At this point, to again satisfy the stacking constraint, we can only swap a colorant pixel with a White pixel that had that colorant value at the previous level. Pixels that were White at the previous level cannot be changed. Once again, we repeatedly go through the cycle of three sequential optimizations until no change is accepted.

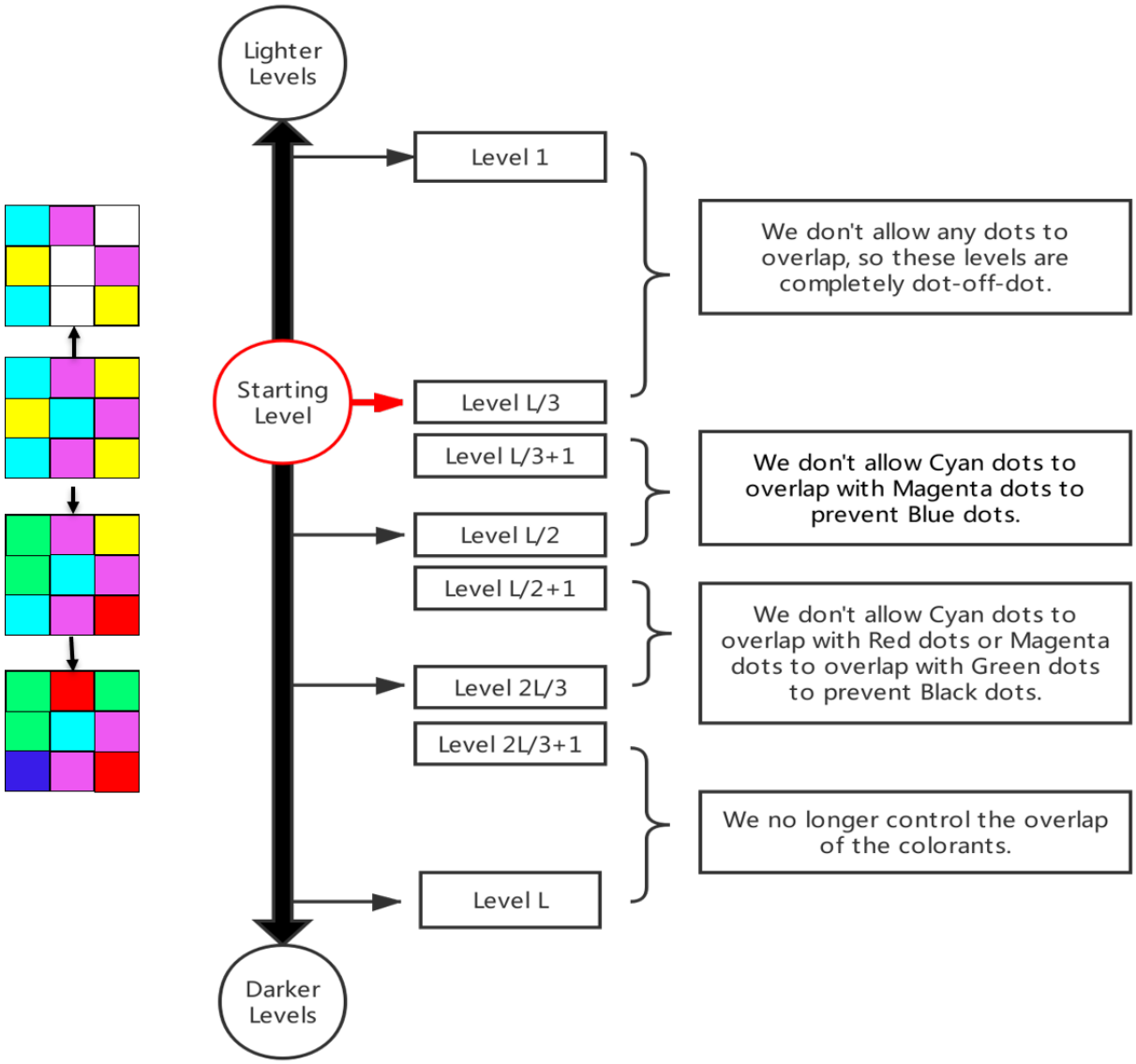


Figure 2.10. Design order and colorants control.

Figure 2.11. The uniform patterns for Magenta colorant in the 1st, 2nd, and 3rd levels.

Then, we continue designing the next lighter levels $\frac{1}{3}L-3, \frac{1}{3}L-4, ..., 1, 0$ by repeating this process until there are no colorant dots to be removed. At this point, we have completed the design of all the levels lighter than the starting level $\frac{1}{3}L$. For all levels that lighter than $\frac{1}{3}L$, we can guarantee patterns are completely dot-off-dot, and also, the three colorants dots are uniformly distributed in the halftone patterns at each level, as illustrated in Figure 2.11 for the Magenta colorant.

From the levels greater than $\frac{1}{3}L$, secondary colors will appear due to overlaps between colorants. Our algorithm directly operates only on the Cyan, Magenta and Yellow colorants; so we need to control the secondary colors such as Blue, Green, and Red in our processing.

We know that for the printer color space:

$$\text{Cyan} + \text{Magenta} = \text{Blue}$$
$$\text{Cyan} + \text{Yellow} = \text{Green}$$
$$\text{Magenta} + \text{Yellow} = \text{Red}$$

We hypothesize that the secondary colors Red and Green blend well with the primaries that contribute most strongly to their lightness and hue [5] [15]. That is, Red harmonizes well with Magenta, and Green harmonizes well with Cyan. However, Blue dots are more problematic than Red and Green dots. Figure 2.12 illustrates this phenomenon, although it is important to note that here the colorants and colors are generated by additive combinations of Red, Green and Blue primaries, not subtractive combinations of actual printed colorants Cyan, Magenta and Yellow. Therefore, when we design levels darker than $\frac{1}{3}L$, we prefer to let the Yellow dots first overlap with the other two colorant dots, but we don't allow Cyan dots to overlap with Magenta dots to prevent the Blue dots.

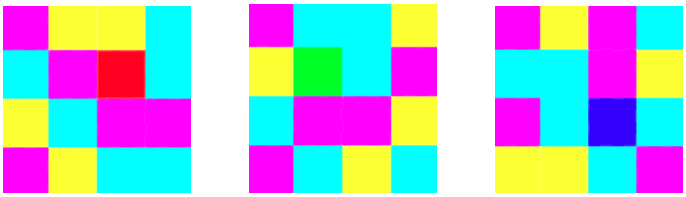


Figure 2.12. The secondary colors Red, Green and Blue with the primary colors Cyan, Magenta and Yellow.

Figure 2.13. The uniform patterns for the Magenta colorant in the 85th, 130th and 160th levels. Here, the first Magenta pattern is designed under the constraint that there are no Blue dots in the composite halftone pattern; the second and third Magenta patterns are designed under the constraint that there are no Black dots in the composite halftone pattern. When viewing these patterns, the reader is advised to zoom in until the individual dots become visible. Otherwise, there may be sampling artifacts.

To design the level $\frac{1}{3}L+1$, we start by adding, at random, M^2/L colorant dots to each of the three starting patterns. Secondary colors appear when we add new colorant dots. However, until we pass the level $\frac{1}{2}L$, we can prevent the occurrence of Blue dots, by not allowing Cyan and Magenta dots to be placed at the same pixel location. Thus every new Cyan dot or new Magenta dot will have to be placed on a Yellow dot, resulting in either a Green or a Red pixel, respectively. And each new Yellow dot must be placed on either a Cyan or Magenta dot. After the initial random assignment is made, we follow the same three-stage sequential optimization described in the preceding paragraphs, but only swap Green and Yellow pixels or Red and Yellow pixels. When this process is completed, we have the optimized design of the three halftone patterns for level $\frac{1}{3}L+1$.

To design the halftone patterns for level $\frac{1}{3}L+2$, we again add, at random M^2/L colorant dots to each of the three halftone patterns for level $\frac{1}{3}L+1$, following the same constraints as we did for level $\frac{1}{3}L+1$. However, during each of the three stages of the optimization process,

we may only swap dots that were newly added at the beginning of the design procedure for this level. Again, we do not allow Cyan and Magenta dots to be placed at the same location.

Once we reach level $\frac{1}{2}L$, half of the M^2 pixels in the composite halftone pattern contain Cyan dots; and the other half of the pixels contain Magenta dots. Of these pixels, $\frac{1}{4}$ will be Green, $\frac{1}{4}$ will be Red, $\frac{1}{4}$ will be Cyan, and $\frac{1}{4}$ will be Magenta. Thus, when we design the halftone patterns for levels $\frac{1}{2}L + 1$ and above, we have to allow the occurrence of Blue dots. In fact, for level $\frac{1}{2}L + 1$ each new Cyan dot can only be placed on an existing Magenta pixel and each new Magenta dot can only be placed on an existing Cyan dot. With this constraint, we can avoid the occurrence of Black pixels, which would result if a new Cyan dot is placed on a Red pixel, or if a new Magenta dot is placed on a Green pixel. We can avoid having any Black pixels until we reach the level $\frac{2}{3}L$. At this point, every pixel will be either Red, Green, or Blue.

2.4 Results

In this section, we show sample halftone images obtained using a CMY screen set that has been jointly designed following the procedures described in previously in this chapter. Each screen contain $M \times M = 256 \times 256$ pixels and $L = 256$ levels. The halftoning is implemented in CMY color space. We convert RGB to CMY components by using the formula:

$$\begin{aligned} \text{C} &= 255 \text{ - R} \\ \text{M} &= 255 \text{ - G} \\ \text{Y} &= 255 \text{ - B} \end{aligned}$$

and use the halftone rule below:

$$\begin{aligned} &\text{If } \text{C}_{\text{image}}[\text{i},\text{j}] < \text{C}_{\text{screen}}[\text{i},\text{j}], \text{ print a Cyan dot at } [\text{i},\text{j}]; \\ &\text{If } \text{M}_{\text{image}}[\text{i},\text{j}] < \text{M}_{\text{screen}}[\text{i},\text{j}], \text{ print a Magenta dot at } [\text{i},\text{j}]; \\ &\text{If } \text{Y}_{\text{image}}[\text{i},\text{j}] < \text{Y}_{\text{screen}}[\text{i},\text{j}], \text{ print a Yellow dot at } [\text{i},\text{j}]; \end{aligned}$$

Using the set of Cyan, Magenta and Yellow screens, the color halftone processing can be simplified to a channel-independent comparison operation on a pixel-by-pixel basis. Figure 2.14 illustrates this process.

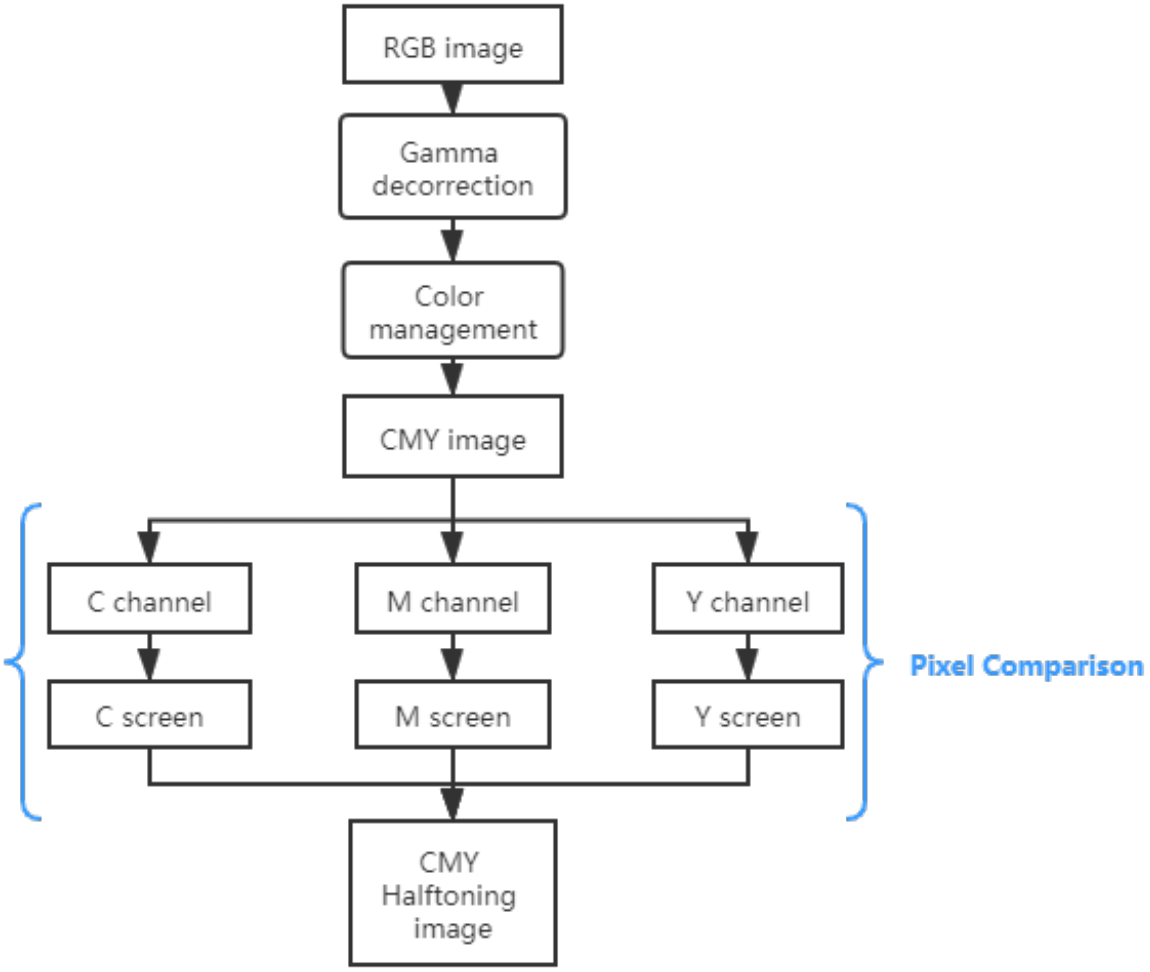


Figure 2.14. Halftoning framework.

We present a continuous-tone image in Figure 2.15 and the halftone image obtained using our jointly designed screen set in Figure 2.16 to illustrate the halftone quality. Also, we compare our result with a halftone image obtained using independently designed screens in Figure 2.17. To better compare the uniformity of the halftone images, we zoom in on the same parts of the two halftoned images. From the detailed comparison in Figure 2.19, it is clear that the color image halftoned by the jointly designed C, M and Y screen set greatly reduces the occurrence of secondary colors and white holes. Hence, it appears less grainy.

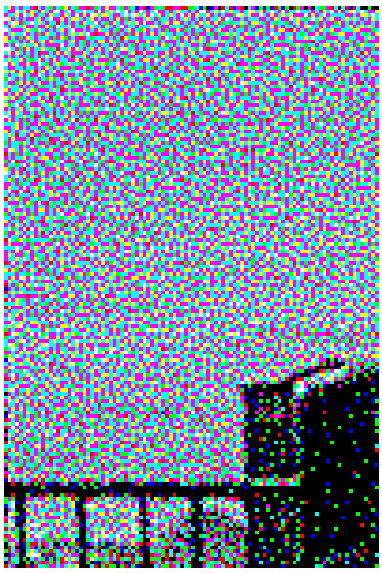
2.5 Conclusion

This chapter describes an algorithm to jointly design screen sets for Cyan, Magenta and Yellow colorants using the DBS algorithm. The results show that the color images halftoned using the jointly designed screen set are more uniform than those halftoned using independently designed screens, due to the reduction in secondary colors and white holes. Also, compared with halftoning a color image directly using DBS, jointly designed screen sets can greatly reduce the computational complexity.

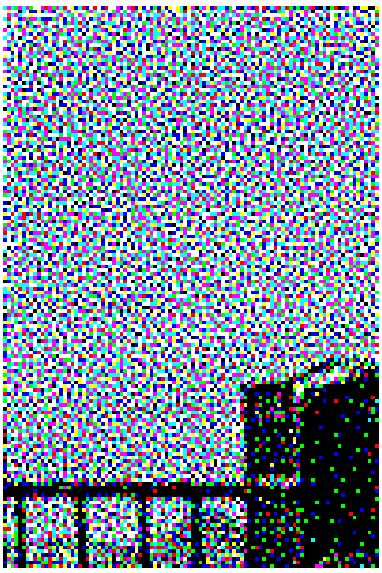
Figure 2.15. Continuous-tone color image.

Figure 2.16. Image halftoned using jointly designed screen set.

Figure 2.17. Image halftoned using three independently designed screens.



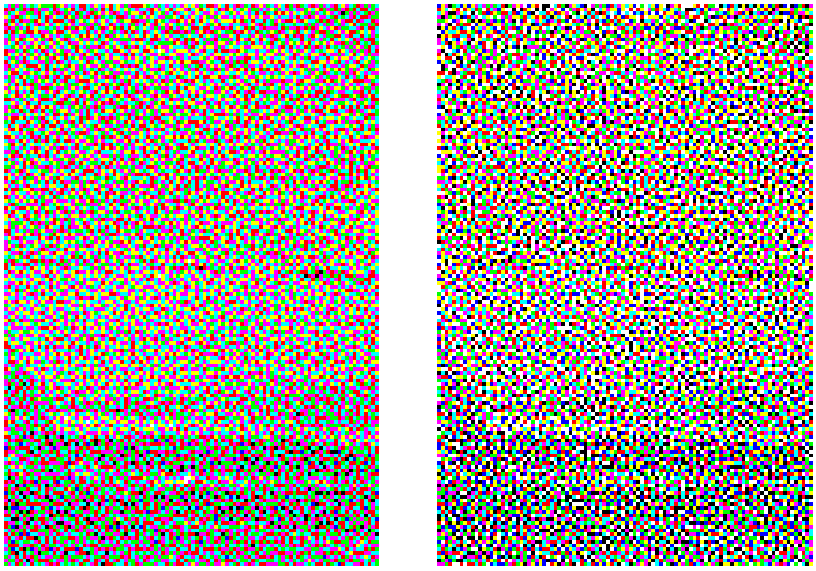
(a) Part a in Figure 2.16.



(b) Part a in Figure 2.17.

(c) Part b in Figure 2.16.

(d) Part b in Figure 2.17.



(a) Part c in Figure 2.16. (b) Part c in Figure 2.17.

Figure 2.19. Comparison of the zoomed-in detail of three regions in the two halftone images shown in Figure 2.16 and Figure 2.17. (The right is halftoned by independently designed screens, and the left is halftoned by jointly designed screens).

3. IMAGE QUALITY RULER EXPERIMENTS AND PRINT UNIFORMITY PREDICTOR

3.1 Background and Previous Work Review

Generally, the end-user of a printed image is a human, so the human judgment of the printed image quality is extremely important. Therefore, the only accurate way to assess printed image quality should be based on subjective criteria. However, subjective assessment is usually complicated, time-consuming, and expensive. In addition, subjective experiments are difficult to replicate or verify. Therefore, it is also critical to use objective methods to assess the quality of printed images.

Considering the many advantages of objectively evaluating image quality, in the past few decades, many objective image quality assessment (IQA) indicators have been proposed to predict the human perceptive image quality. Based on the accessibility of reference images (almost perfect original images), objective image quality evaluations can be divided into three categories [17]. The image quality can be objectively predicted based on comparison with the reference image, which is called full-reference (FR), and there are many evaluation factors related to full-reference, such as root mean square error (RMSE), structural similarity index (SSIM), and peak signal-to-noise ratio (PSNR) [18]. However, in many practical applications, there is no reference image available, therefore, there is also reduced-reference (RR) quality assessment, which requires only partial information of the reference image. If the reference images totally cannot be reached, no-reference (NR) quality assessment is demanded. Our work focuses on the quality assessment of non-reference images.

There are many ways to conduct no-reference image quality assessment, but the inadequacy of these methods is that it is very hard to verify the final result. Therefore, the $INCITS\ W1.1$ macro-uniformity team has proposed a new method to evaluate image quality called the quality ruler method [19], [20], [21] and [22]. They designed test pages with simulated defects, and used them as a quality ruler in the subjective evaluation. In this chapter, we describe the process of a psychophysical experiment using the quality ruler method following the guidelines of the $INCITS\ W1.1$ team and ISO 20462 [21].

The $INCITS\ W1.1$ activity recognizes that printed image quality can be well described by a small set of attributes, including gloss uniformity, macro-uniformity, and micro-uniformity. Numerous recent papers show that among these attributes, macro-uniformity draws the most attention [19] and [20].

Macro-uniformity (ISO 19751 macro-uniformity) refers to the subjective impression of color uniformity across a large image area that is intended to have a uniform color. There are several kinds of print quality defects that influence the perception of macro-uniformity [19] and [20]. They are:

- Banding: one-dimensional, periodic lightness and/or chromatic variations.
- Streaks: one-dimensional, isolated lightness and/or chromatic variations.
- Graininess: two-dimensional, fine-scale, random texture with a sand-like appearance.
- Mottle: two-dimensional, medium-scale, random lightness variations.
- Large area variation: two-dimensional, random lightness variations, for which the spatial region is larger than for mottle.
- Large-scale non-uniformity: one-dimensional, low-frequency lightness variations.

These defects are very important to print quality, yet it is difficult to evaluate the overall macro-uniformity when they occur simultaneously.

The $INCITS\ W1.1$ macro-uniformity team developed a method to measure overall macro-uniformity. They created macro-uniformity quality ruler samples by imposing increasing levels of non-uniformity in a synthetic defect pattern, which consists of a multitude of normally occurring defect types. Experiments were conducted to calibrate the quality ruler in terms of just noticeable differences (JND) [20].

Quality rulers labeled 3, 6, 9, 12, 15, 18, 21, 24, 27, and 30 can be generated by the macro-uniformity software [22]. The smaller the label (JND) is, the better the print quality is. The first quality ruler ($JND = 3$) would appear nearly perfect, with only minor defects. The perceived defect level is approximately logarithmic with the amplitude of the defects, and all levels are visually equidistant according to the quality of the print samples.

With quality rulers, according to the ISO 20462-3 international standard, a psychophysical experiment for estimating printing quality can be designed [21]. The quality ruler

method is superior to many other psychophysical methods, because it can assess a large range of printing quality levels with relatively few resources.

In this chapter, we complete the following three tasks:

1. Compute a value for each defect in the macro-uniformity set, which can represent the severity of the defect.

2. Design and conduct a psychophysical experiment according to the ISO 20462-3 international standard, which includes the selection of test samples, the calibration of the printer and scanner, and the environmental preparation and detailed guidance during the experiment.

3. Analyze the data and built prediction models. We use Linear Regression and SVM to build prediction models which can predict the subjective assessment of the JND based on the objective defect values calculated for the test print.

The prediction model we built can automatically predict the perceived print image quality based on the viewer's opinion. The predictor has many applications in printing systems, which can be used not only to predict the quality of printed images but also to evaluate and optimize the calculation parameters of print defects.

Our overall method follows that of [25]; and we used macro-uniformity generation software provided by [22].

3.2 Methodology

3.2.1 Macro-Uniformity Quality Ruler

To print the samples for the image quality ruler, we used an *Epson Stylus Pro* 3880 color inkjet printer[1], which is a photo quality inkjet printer, as recommended by the $INCITS$ $W1.1$ macro-uniformity team. Printer calibration is required before generating and printing the quality ruler. We first used the W1.1 macro-uniformity software [22] to generate the printer calibration file. An image of the calibration page is shown in Figure 3.1.

[1]Epson America, Inc., Long Beach, CA.

A companion file for the calibration page contains a table of input values for the patches in the generated calibration test pattern. The first column in this file contains an arbitrary index. The second column is the row-index of the test patch. The third column is the column-index of the test patch. The fourth column is the input value to the printer, as a CIE Y value from 0 to 100, before mapping to an 8 *bit* value from 0 to 255.

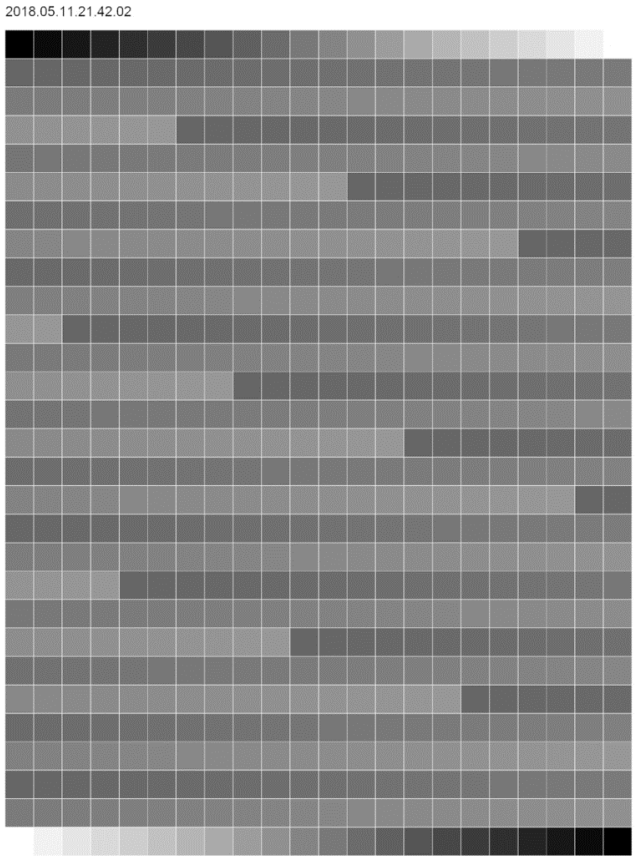


Figure 3.1. The calibration pattern.

The file containing the calibration page needs to be printed using the same printer that is going to be used to print quality rulers. We then measured the CIE XYZ values of each patch on the printed file using an *X-Rite* DTP-70 [2], color measurement instrument. The measuring process can be repeated several times to minimize the effect of noise. The final

[2]X-Rite, Inc., Grand Rapids, MI.

step to complete printer calibration is to arrange all the patches' average CIE XYZ values into a specific format file and input it into the $W1.1$ macro-uniformity software.

The software automatically calibrates the CIE Y data according to the input file, and then generates quality ruler samples with specific defect scales.

Each quality ruler sample includes a 170 mm × 170 mm defect region and a test target surrounding the defect region. As shown in Figure 3.2, the quality rulers are created at fixed quality levels ranging from highest to lowest in steps that are 3 $JNDs$ apart. Quality ruler samples labeled 3, 6, 9, 12, 15, 18, 21, 24, 27, and 30 can be generated by the software. But according to the severity of the defects in the test samples, the appropriate range should be chosen. (Quality rulers with JND from 3 to 21 were used in our work.)

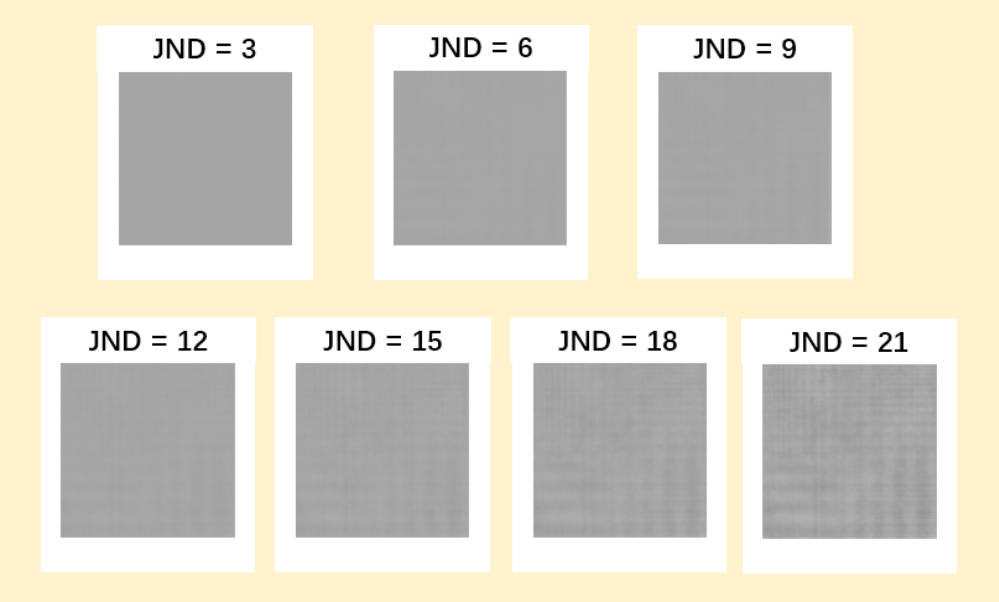


Figure 3.2. Quality ruler samples labeled with JND = 3, 6, 9, 12, 15, 18, and 21. The smaller the label (JND) is, the better the print quality is. The first quality ruler (JND = 3) would appear nearly perfect

3.2.2 Compute Defect Features in the Macro-Uniformity Test Set

Our test samples were printed at four different tint levels with a prototype large-format printer using a page-wide array inkjet print bar. Print defect features such as banding, streaks, graininess, etc., could be seen in the print samples. The framework for computing the defect features includes scanning hard-copy test samples, removing halftoning pattern using a descreening processing and then computing values that represent the severity of defect features as described below.

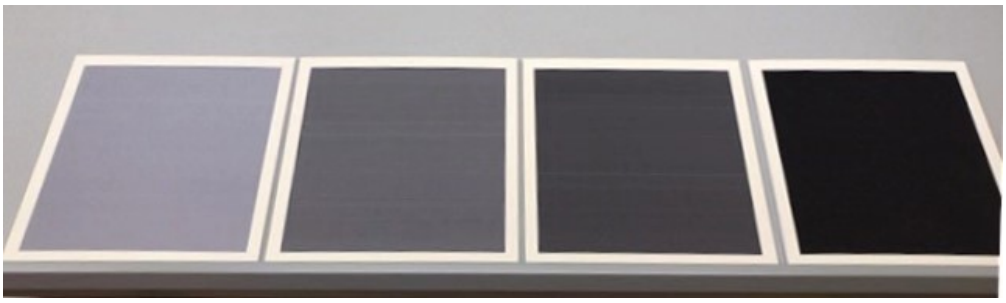


Figure 3.3. Print samples with tint levels 30%, 50%, 70%, and 100% were printed with a prototype large-format printer using a page-wide array inkjet print bar. We selected 12 samples from levels 30%, 50% and 70%, and 6 samples from level 100%, resulting in a total of 42 test samples.

First, scanner calibration needs to be performed before scanning the test samples. All test samples were scanned at 600 *dpi* resolution, as recommended by a previous study [23]. We used an *Epson Expression* 10000XL scanner. The scanner calibration was performed as described in [24] and [25]. A *Kodak Q*60 reflective target was scanned, and an *X-Rite DTP*-70 was used to determine the CIE XYZ values of the patches. The gray patches were used to determine the gray-balance curves for each of the R, G, and B scanner channels. Then, the 240 color patches were used to determine the elements of a 3×3 matrix used to transform from linear scanner RGB to CIE XYZ. Finally, we transform from CIE Y to CIE $L*$ to complete the transformation that is applied to the monochrome test pages.

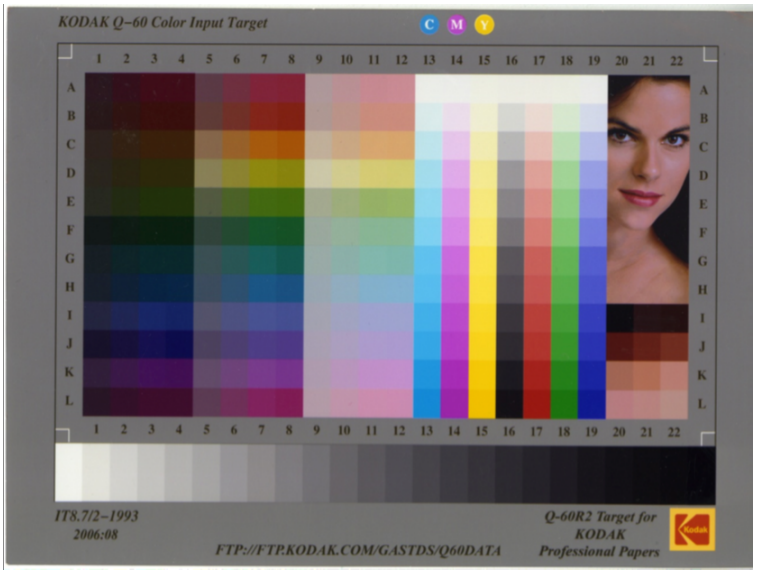


Figure 3.4. The Kodak Q-60 target for scanner calibration.

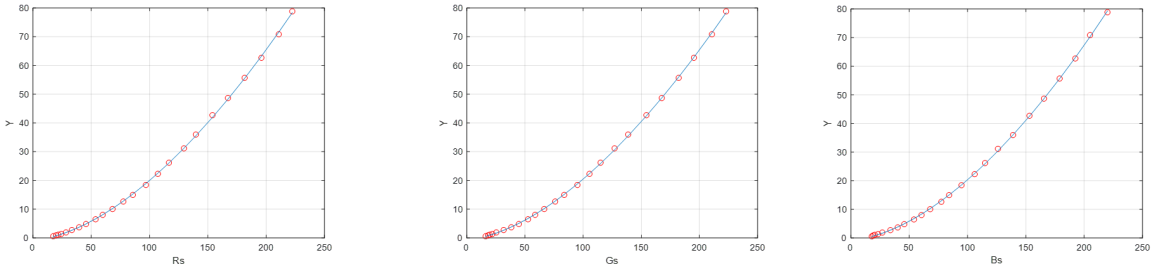


Figure 3.5. Gray balancing curves for the R, G, B channels.

The Gray balancing results are:

$$R_l = 99.4942(\frac{R_s}{255})^{1.6821} - 0.6268$$
$$G_l = 98.5282(\frac{G_r}{255})^{1.6542} - 0.4967$$
$$B_l = 102.6936(\frac{B_r}{255})^{1.7003} - 0.5815$$

The transformation from Linear RGB to CIE XYZ is :

$$\begin{bmatrix} X \\ Y \\ Z \end{bmatrix} = \begin{bmatrix} 0.3628 & 0.3310 & 0.1875 \\ 0.1137 & 0.1407 & 0.1901 \\ 0.1819 & 0.1846 & 0.1197 \end{bmatrix} \begin{bmatrix} R_l \\ G_l \\ B_l \end{bmatrix}$$

A human vision model is then applied to the scanned samples to measure defects as perceived by a human subject. The contrast sensitivity function (CSF) we used in the human vision model was proposed by Mannos and Sakrison [26]. The viewing distance of the CSF is set to 15.7 $inches$ (approximately 40 cm), which is also the viewing distance recommended by the $INCITS\ W1.1$ working group for conducting psychophysical experiments using the quality ruler method.

After applying the human vision model, we converted the samples from the RGB color space to the CIE $L^*a^*b^*$ space. Since our test samples are all printed in grayscale, we only use the lightness channel (L^* channel) to compute the defect features in the macro-uniformity test set. Figure 3.6 illustrates the pipeline of applying human vision model. Figure 3.7 shows a 70% tint test sample before and after filtering with our human vision model.

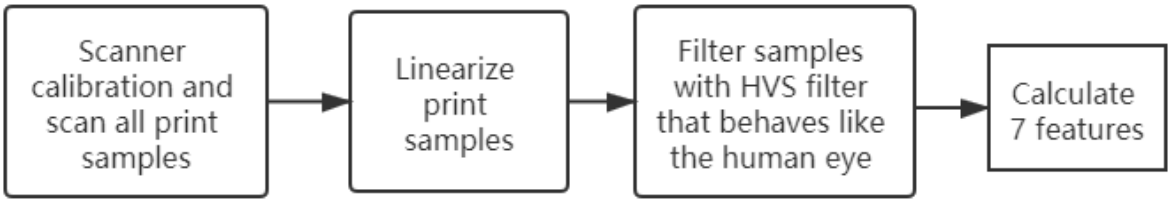


Figure 3.6. A flow chart for applying the human vision model.

To compute values that can represent the severity of the defects in the macro-uniformity test set, spatial variations including one-dimensional, two-dimensional, periodic, aperiodic, localized, large-scale, and small-scale variations were considered. In the remainder of this section, we describe the attributes of each defect feature, and how it is computed. The

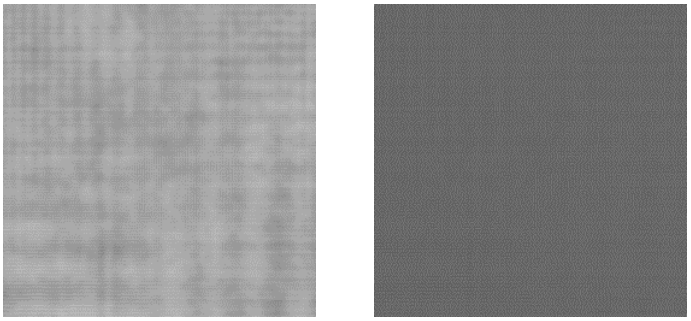


Figure 3.7. A test image before and after HVS filtering.

methods we used to compute values for the defect features are mainly inspired by ISO image quality standards [21], and previous work [25].

Graininess refers to the image fluctuation in both the horizontal and vertical directions of the image. We use the root mean square fluctuation ($RMSF$) of L^* value to measure Graininess.

$$G = \sqrt{\frac{1}{MN}\sum_{\mathrm{i}=1}^{M}\sum_{\mathrm{j}=1}^{N}({L^*}_{\mathrm{ij}} - \overline{L^*})^2} \tag{3.1}$$

where M and N are the number of image pixels in the vertical and horizontal directions, and $\overline{L}^*$ is the average value of L^* in the print sample.

Mottle refers to random lightness variations in both the horizontal and vertical directions of the image. To compute its value, an averaging window of $2 \times 2\ mm^2$ (47×47 pixels at the scanning resolution of 600 dpi) is convolved with the sample. Then, the standard deviation of the resulting array is computed as the Mottle.

Large Area Variation also refers to two-dimensional random lightness variations, but the spatial region is larger than for Mottle. We use an averaging window of $4 \times 4\ mm^2$ (95×95 pixels at the scanning resolution of 600 dpi), and convolve it with the sample. Then the difference between the largest and smallest array entries is computed as the Large Area Variation.

Banding and Streaks represent high-frequency lightness variations in different directions. So their computational methods are the same except for the processing direction. First, the $1D$ (one-dimensional) projection in both the horizontal and vertical directions is performed. Then the average of the signal is subtracted from the signal to exclude the DC component. After that, its DFT (discrete Fourier transform) is computed; and the strengths of signal peaks are measured. Peaks with frequencies less than 10 $cycles/inch$ are filtered out, and the energy of the remaining peaks is integrated to generate the banding and streaks features.

Large-scale Non-uniformity represents two-dimensional low-frequency lightness variations. First, we perform $1D$ projection in the horizontal and vertical directions. Then, we smooth the projections with an averaging window of length 3 mm (80 pixels at the scanning resolution of 600 dpi). After that, we use a piecewise linear spline fit to iteratively add knots until the maximum error between two adjacent knots is less than $0.5\Delta E$ units. The Large-scale Non-uniformity is obtained by computing the mean absolute slope of those line segments for which $\frac{\Delta L*}{\Delta d} > 0.5$, where d is the distance in $inch$.

3.2.3 Psychophysical Experiment

The psychophysical experiments were conducted under controlled viewing conditions in a laboratory at Purdue University dedicated for this purpose. The viewing booth used was the *Graphiclite* $CVX2$ [3]. For viewing, the quality ruler samples and test samples were placed in frames. The frames were fabricated by *MatShop* [4]. For the ruler samples, the frames were 9×9 $inch^2$ in size with a 6×6 $inch^2$ opening. The frame border was sufficiently large to hide the test target around the border. For the test samples, the frames were 9×9 $inch^2$ in size with an 8×8 $inch^2$ opening. These frames can be seen in Figure 3.3. For both the ruler samples and the test samples, the frame color was cream with a white color core, which was chosen to be as neutral as possible.

[3]GTI Graphic Technology, Inc., Newburgh, NY.
[4]MatShop, Victoria, BC, Canada.

Figure 3.8. Two views of the lab environment and viewing booth. In the far left of the left image, some of the wooden platforms used to adjust viewing height can be seen standing on end.

A total of 26 subjects participated in the entire experiment. 14 of these subjects came from our research group; and most of them had an image processing background. The other 12 subjects were from non-engineering departments at Purdue, and did not have an image processing background. We conducted a pilot experiment before the official experiment to double-check that the procedure worked well. The data from the pilot experiment was not used to train or evaluate the predictors.

Before each subject's experiment, we tested their visual acuity and color vision, and adjusted their viewing distance based on their height by having them stand on an appropriate number of stacked wooden platforms. We showed all quality ruler samples as well as the test samples, and briefly explained the experimental process.

During the experiment, the subjects walked along the viewing booth, and slid the test sample in front of them, comparing it with the hard-copy quality ruler samples, until finding a suitable location based on overall visual uniformity (The test sample's location meets the condition that each ruler sample farther to the right is lower in quality than the test sample, and each ruler sample farther to the left is higher in quality). Since the reference stimuli are labeled 3, 6, 9, etc., if the test sample's location fell in between two adjacent ruler images, as it often did, the subject then selected an intermediate integer value from the ruler scale. For example, if the location was between the ruler prints "12" and "15", but was closer to

"12", the value of the test sample was assigned to be "13". Figure 3.9 illustrates the pipeline of psychophysical experiment.

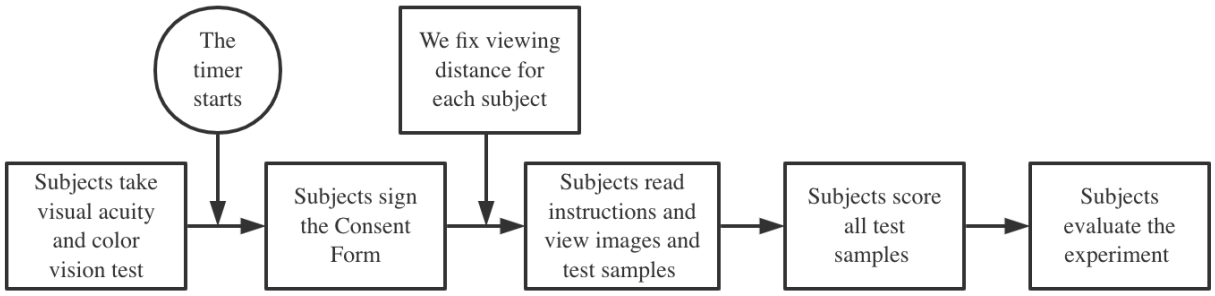


Figure 3.9. A flow chart of psychophysical experiment.

3.2.4 Outlier Analysis

Before using the subjects' assessments, we pre-processed the data and removed the outliers. In our work, we define a subject's data to be an outlier if it meets both of the following conditions:

1). Weak consistency.

2). Large average absolute deviation.

We conducted the consistency test as part of the experimental process. We selected 4 samples from different tint levels and repeatedly added these 4 samples into the waiting list in the early, middle, and end stages of each participant's experiment. So each subject assessed the uniformity of these four samples 3 times throughout the experiment.

We performed the same procedure for each subject to collect data to analyze their score consistency. For a given test sample, if the difference between the highest and lowest scores

given by a subject was greater than 6 JND, the subject's score was considered to be weakly consistent. Thus, the weak consistency condition is defined as:

$$X^{\mathrm{i}}_{max} - X^{\mathrm{i}}_{min} \geq 6 \tag{3.2}$$

where X^{i}_{max} and X^{i}_{min} represent the highest and lowest scores given to the same print sample i by the subject at different stages of the experiment.

The second condition for the subject's data to be an outlier is that it exhibits a large average absolute deviation, which is:

$$\frac{1}{N}\sum_{\mathrm{i}=1}^{N} |S_{\mathrm{i}} - \overline{S_{\mathrm{i}}}| > 3 \tag{3.3}$$

where N denotes the total number of samples, S_{i} denotes the score assigned by the subject to the print sample i, and $\overline{S_{\mathrm{i}}}$ denotes the average score assigned by all subjects to the print sample i. If the subject failed both these tests, then all their scores were eliminated from further analysis of the dataset.

Of our 26 subjects, 6 participated in a pilot experiment to ensure the procedure went smoothly. Thus, 20 subjects participated in the formal experiment. As shown in Figure 3.10, the x-axis represents the average absolute deviation of the 20 participants who participated in the formal experiment. According to formulas (3.2) and (3.3), the values of Nos. 3, 6, and 12 exceed the maximum tolerance value of average absolute deviation, and they also fail to meet the requirement of consistency. Therefore, the data from these three subjects was excluded as outliers.

3.2.5 Prediction Models

We have collected 26 participants' perceptual assessments (JND) of 42 samples. After removing the outliers as described in the previous section, we obtained the average perceptual assessments of all samples, which was the ground-truth used to build the prediction model. For each sample, the seven features described previously, which represent the severity of

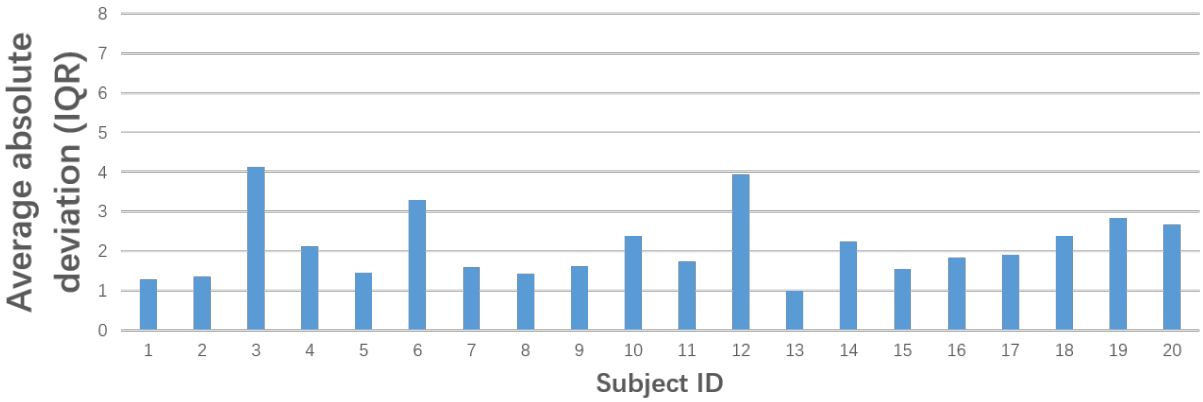


Figure 3.10. Subject's absolute deviation results (20 participants who participated in the formal experiment).

printing defects in the macro-uniformity set were computed. This data was also the input for building the prediction model. Our goal was to find the relation between defects and the overall perceptual uniformity. Using this correspondence, we can predict the overall JND score for prints in the future.

In this chapter, we used linear regression (LR) and support vector regression (SVR) to build predictors. Linear regression is a simple algorithm that models the relationship between a scalar response and one or more explanatory variables, and then predicts future data response based on that relationship. The support-vector machine (SVM) is a supervised learning model used for classification and regression analysis. The learning strategy of SVM is to identify the hyperplane that maximizes the margin, so the task is transformed into a convex quadratic programming problem.

Linear Regression

Linear regression [32] is a method of modeling the relationship between the variable y and one or more variables X. In linear regression, linear functions are used for modeling the relationship, and the unknown parameters of linear functions can be estimated by known data [14]. The common LR form is:

$$y = \theta_1 x_{i1} + \theta_2 x_{i2} + ... + \theta_j x_{ij} + \epsilon_i, \quad i = 1, ...n \tag{3.4}$$

Linear regression models are usually solved by the least-squares method or by minimizing the least-squares loss function (such as $LASSO$).

Support Vector Machine (SVM) Regression

In SVM regression (SVR) [33], the input feature x is first mapped to high-dimensional feature space using a fixed non-linear mapping. Then, an ε-insensitive loss and $L2$ regularization are used to construct a linear regression model in the high-dimensional feature space.

The SVR is expressed as a minimization problem by the following formula:

$$\begin{aligned} \underset{w,\xi,\xi^*}{minimize} \quad & \frac{1}{2}||w||^2 + C\sum_{i=1}^{N_p}(\xi_i + \xi_i^*), \\ subject \;\; to \quad & y_i - f(x_i, w) \leq \varepsilon + \xi_i^*, \\ & f(x_i, w) - y_i \leq \varepsilon + \xi_i, \\ & \xi_i, \;\; \xi_i^* \geq 0, \quad i = 1, ..., N_p \end{aligned} \tag{3.5}$$

This optimization problem can be converted to its dual problem and the solution is given by:

$$f(x) = \sum_{i=1}^{n_{SV}}(\alpha_i - \alpha_i^*)K(x_i, x) + b \tag{3.6}$$

where $0 \leq \alpha_i \leq C$, $0 \leq \alpha_i^* \leq C$, n_{SV} is the number of support vectors and the kernel function $K(x, x_i)$ is defined as:

$$K(x, x_i) = \sum_{j=1}^{M} g_j(x) g_j(x_i) \tag{3.7}$$

In our work, we used the Least-squares support-vector machines ($LS-SVM$) to solve the regression problem. It has the same principles as the SVM for classification, with only a few minor differences. For the kernel function K, typical choices are the linear kernel, polynomial

kernel, and radial basis function (RBF) kernel. We use the RBF kernel represented by the following formula:

$$K(x, x_{\mathrm{i}}) = exp(\frac{-||x - x_{\mathrm{i}}||^2}{\sigma^2}) \tag{3.8}$$

We built the SVM predictor with the help of these package and references [27] and [28].

In the process of building the model, we used k-fold cross-validation [29]. Cross-validation is a verifying method used to evaluate how the results of a statistical analysis will generalize to an independent data set. It is mainly used for assessing the ability of a predictive model to predict new data. Also, it can help to find problems such as overfitting or selection bias.

In k-fold cross-validation, the original sample set is randomly divided into k equal-sized subsets. Among the k subsets, one is used as testing data, and the remaining k-1 subsets are used as training data. Then, the cross-validation process is repeated k times, and each of the k subsets is used as the testing data only once. The k results can then be averaged to produce the model estimation. The advantage of this method is that all observations are used for training and testing, and each observation is only used once for testing.

For our data, setting $k = 6$ will result in 6 folds. We randomly shuffle the samples into 6 folds indicated by d_0 to d_5 so that each set is equal in size, which is with 7 samples in each fold. Then, we train on $d_0, ..., d_4$ and verify on d_5, then train on $d_1, ..., d_5$ and verify on d_0, ... This process is repeated 6 times. The average prediction is the model estimation.

3.3 Results

The graphs in Figure 3.11 and Figure 3.12 illustrate our linear regression and Support vector regression results. On 42 samples, we compared subjects' visual scores (orange bars) and our predicted scores (blue bars) in JND units.

To evaluate our models, we use mean absolute error (MAE) and mean squared error (MSE). The standard deviation of MAE is a measure of the robustness of the predictions.

$$MAE = \frac{\sum_{\mathrm{i}=1}^{n} |p_{\mathrm{i}} - a_{\mathrm{i}}|}{n} \tag{3.9}$$

$$MSE = \frac{\sum_{\mathrm{i}=1}^{n}(p_{\mathrm{i}} - a_{\mathrm{i}})^2}{n} \tag{3.10}$$

where p_{i} represents predicted data and a_{i} represents actual data.

For the predictor built by LR, we have:

$MAE = 0.9100\ JND$ $\quad$ $Std.Dev.$ of $MAE = 0.6340$

$MSE = 1.2399\ JND$

For the predictor build by SVR, we have:

$MAE = 0.8305\ JND$ $\quad$ $Std.Dev.$ of $MAE = 0.5469$

$MSE = 0.9463\ JND$

It is a quite encouraging that the MAE between the predicted scores and the subjects' scores for both models is less than 1. As mentioned earlier, the interval between two adjacent quality rulers used as a reference is $JND = 3$ so for an accurate model, its MAE should be less than 3 JND. Also, the standard deviation shows that the error distribution is stable.

We were particularly interested in some samples with a relatively large absolute error. We found that most of these samples were from the 100% tint level. We feel that this is understandable, because the 100% tint level means the sample is totally black. It was more difficult for subjects to notice the defects; and thus these samples were given a relatively better quality evaluation than was predicted.

3.4 Conclusion

In this chapter, we designed and conducted a psychophysical experiment to collect the perceptual assessment of print macro-uniformity. The experiment worked well, and showed that the quality ruler method proposed by the $INCITS\ W1.1$ macro-uniformity team can provide a reliable method to assess macro-uniformity of print samples. Also, we developed models for predicting the overall macro-uniformity as judged by humans. We confirmed the efficacy of the predictors using 6-fold cross-validation. Also, the model evaluation metrics MAE, MSE and standard deviation of MAE indicated that the models can perform accurate prediction.

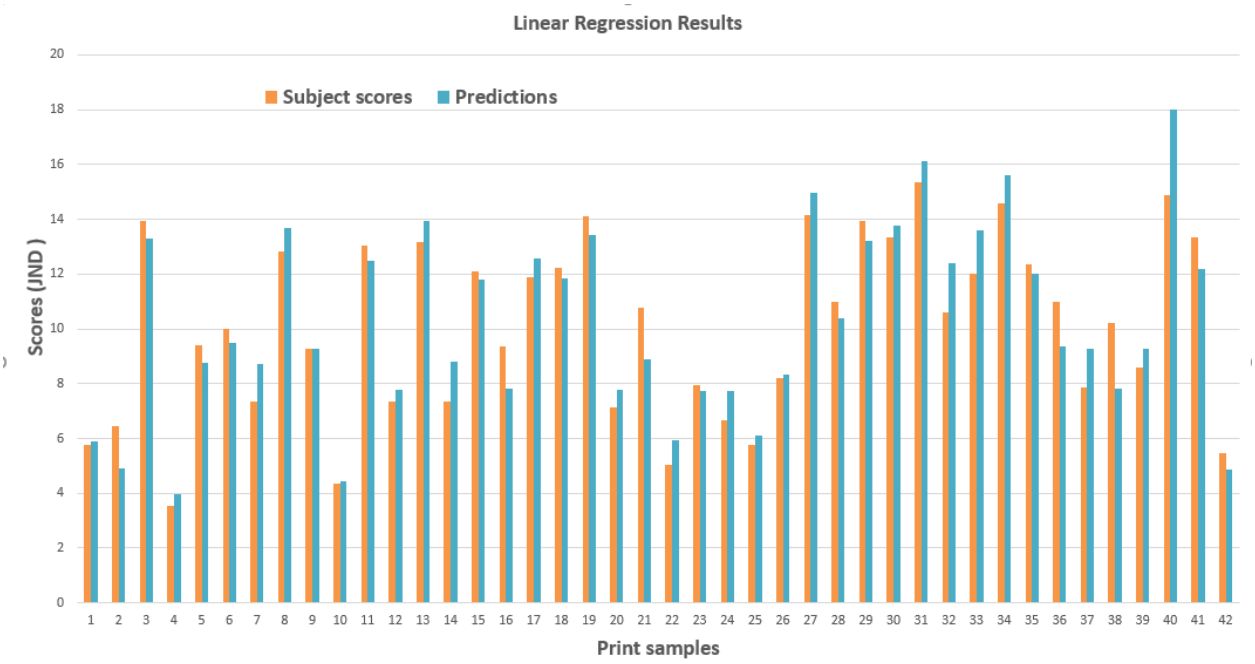


Figure 3.11. The results of the Linear Regression predictor. The abscissa is the sample ID and the ordinate is the JND score. The orange bars indicate the predicted score for each sample. The blue bars indicate the subjects' mean score for each sample.

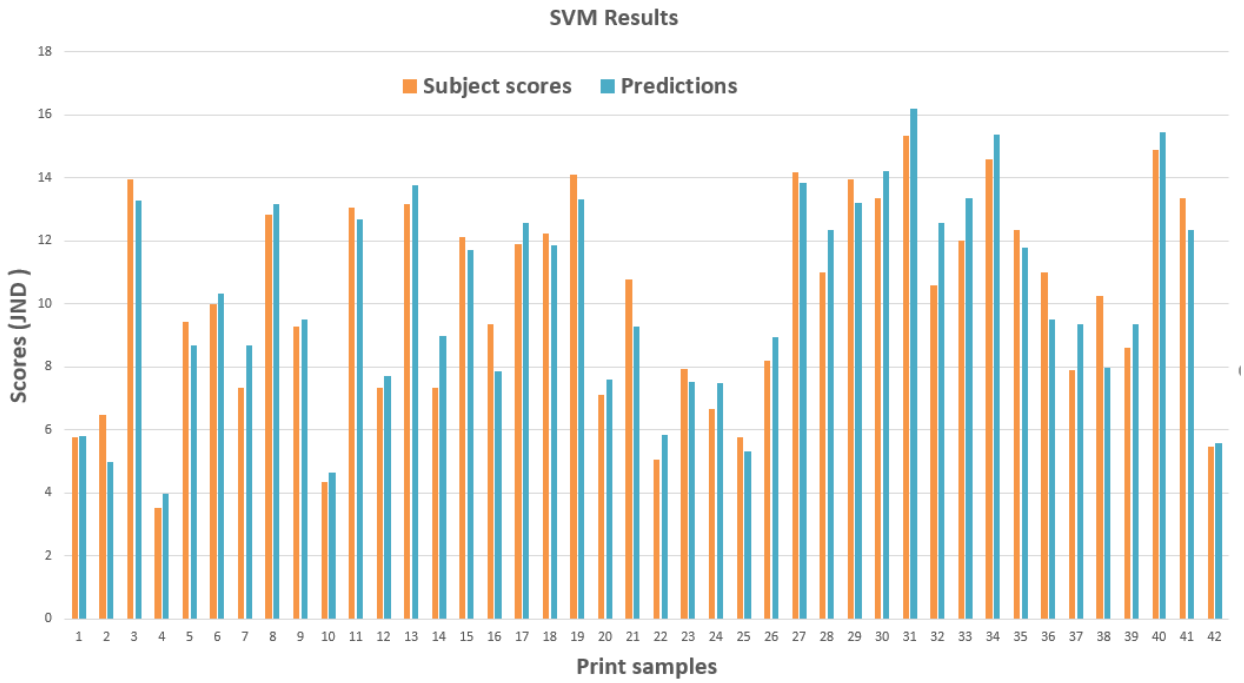


Figure 3.12. The results of the SVR Predictor. The abscissa is the sample ID and the ordinate is the JND score. The orange bars indicate the predicted score for each sample. The blue bars indicate the subjects' mean score for each sample.

4. BANDING DEFECT DETECTION AND IMAGE QUALITY CLASSIFICATION

4.1 Background and Previous Work Review

Banding is one of the most difficult image defects, which is a one dimensional, isolated or periodic, luminance and/or chromatic variation induced by the vibration of different printer components [34]. This defect was categorized under the macro uniformity image quality attribute in the paper [35], which evaluates the overall image quality of the printed image and considers banding as one of the most severe defects that affect the overall perceived image quality.

Much previous work has been done to study various aspects of banding defects, including defect stimulation, visual analysis, Fourier domain analysis, and banding reduction. A common approach to assess the visibility of banding is to conduct psychophysical experiments and collect the subject's perceptual evaluation [36], [37] and [38]. Some methods analyzed banding in the Fourier domain, implemented in one dimension [39] or two dimensions [40]. Several works have also been done at the printer mechanism level to reduce banding, and have produced encouraging results for banding reduction [40] and [41].

We draw inspiration from the banding detection work of Zhang et al.[42], [43] and [44]. The algorithm that they developed can detect banding on printed uniform color pages, and can classify periodic and aperiodic banding. However, this work cannot detect banding defects on customer's content pages. We remedy this with a new design that allows automatic detection of banding defects on customer's content pages. Furthermore, we design a scheme for predicting the overall quality based on banding features and perceptual assessment, which yields promising predictions.

4.2 Methodology

The overall goal of our work is to automatically detect aperiodic and periodic banding on the printed customer's content pages and automatically analyze the print quality according to the banding severity.

As shown in Figure 4.1, the input is the master - test image pair. The master image is the digital original customer's content images, and the test image is the scanned printed customer's content images.

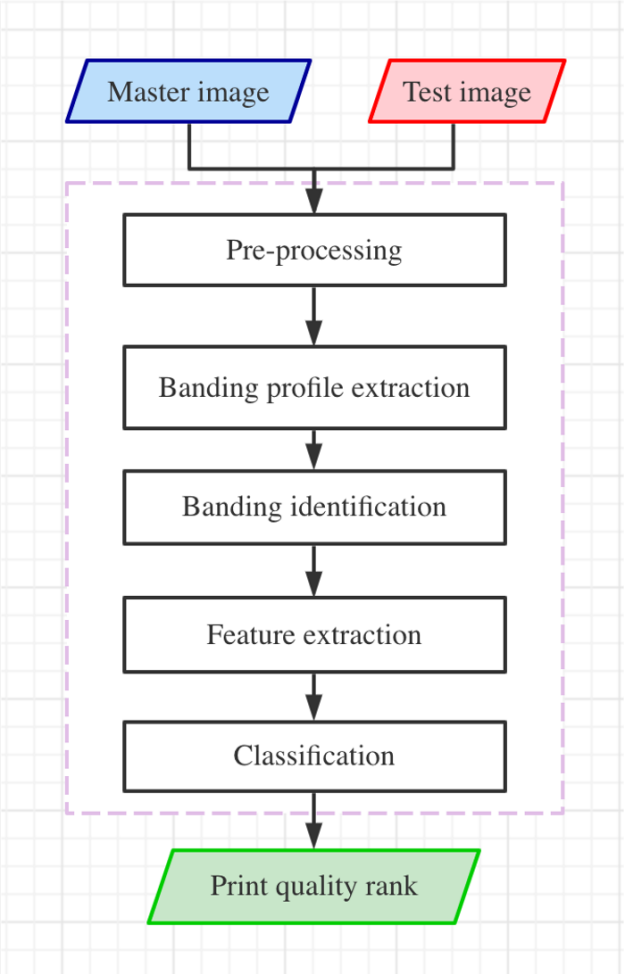


Figure 4.1. Overall pipeline of banding detection and analysis.

We first pass the master-test image pair into the pre-processing pipeline, which includes image registration, region of interest (ROI) extraction, and ROI alignment. Next, we apply banding profile extraction and banding identification to localize and identify banding defects. We extract meaningful features of banding defects, which can indicate the severity of the

defects. Then, we use machine learning models to classify these features into four categories from A to D (A symbolizes almost perfect print quality and D symbolizes the worst print quality. The ground truth is labeled by human subjects).

4.2.1 Pre-processing

Image registration is the first step of pre-processing, which aligns the test image with its master image, to eliminate the misalignment caused by the scanning process.

For our test images, we assume that the geometric transformation involves only a small skew angle rotation and a small translation along the x-axis and the y-axis, so we would like to find several matched key points to compute the best transformation. We convert images from RGB to grayscale to reduce the image dimension, and then resample the image using a 1/3 downsampling rate to save computation. Next, we apply histogram matching to the test image based on the master image to match the image gray values, and then we use Harris Corner Detection [45] to extract key points on the master image and test image, respectively. The feature descriptor is constructed using the 31×31 local areas surrounding each key point. By minimizing the sum of squared differences (SSD) and computing the ratio SSD, we find the best-matched key point pair. We then use the Maximum Likelihood Estimation Sample Consensus (MLESAC) [46] to take advantage of multiple matched point pairs. The transformation estimated by MLESAC is more robust than by random sample consensus (RANSAC) [47].

After obtaining the aligned test image, we would like to generate the object map, and divide the entire image into different ROIs according to the object map. The details of this part were explained in our previous work [56].

In this chapter, all the following processes are implemented on the ROI, but they can also be implemented on the entire image.

4.2.2 Banding Defect Detection

In this section, we extract the banding profile, and then localize and identify banding based on the profile. Next, we refine the banding result, measuring the amount per unit area. For each band, we get its height, width, and interband distance, to describe the severity of the banding. In addition, periodic banding is identified and its repetitive interval is estimated.

Banding Profile Extraction

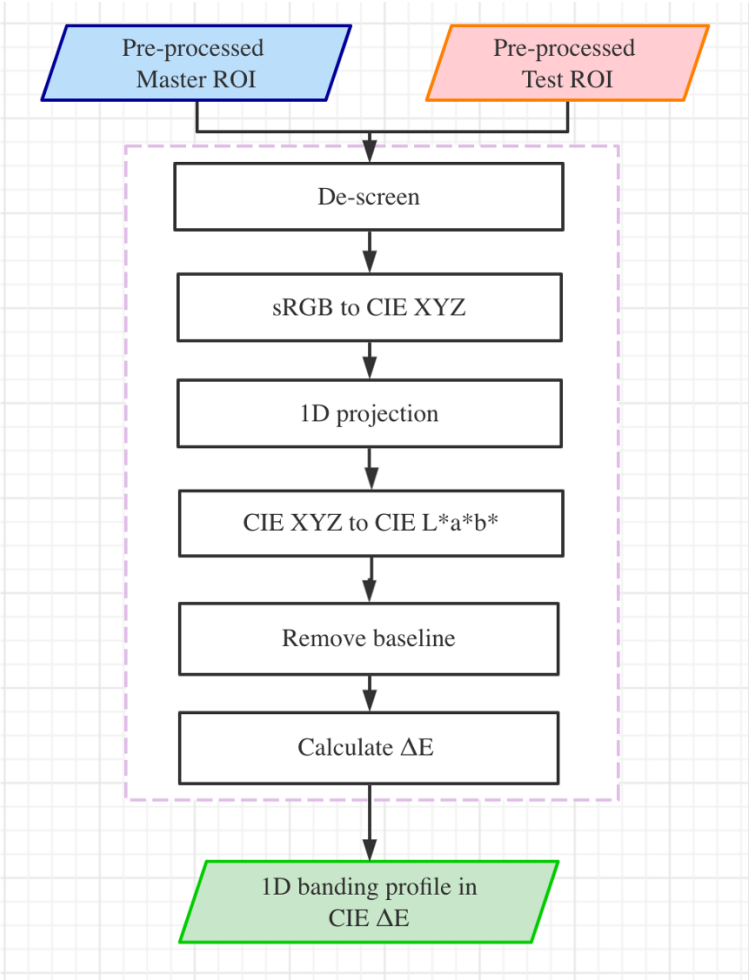


Figure 4.2. Pipeline of banding profile extraction.

Figure 4.2 shows the steps of the banding profile extraction. First, we de-screen the input image or ROI to remove the halftone patterns. A previous work [44] used a median filter with a window size of 0.05 inches (15×15 pixels for an image scanned at 300 dpi) to de-screen for uniform color pages and achieved good performance. But for our customer's

content pages, after several tests, we found that a Gaussian filter with a size of 0.02 inches (7×7 pixels for our pages scanned at 300 dpi) is the best approach to do de-screening.

Then, we convert the image or ROI from the sRGB color space to the CIE 1931 XYZ color space (hereinafter abbreviated as CIE XYZ). Next, in the CIE XYZ color space, the one dimensional projection of the image along the scan direction is computed by calculating the mean value of each line in this direction.

Since the banding may fade along the scanning direction, the 1D projection of the banding may become inconspicuous if we compute it across the entire image. Accordingly, we divide the image into three parts along the scanning direction and perform three separate projections, respectively. Independent analysis of each projection can yield more accurate results. For each part, we compute 1-D projections of the X, Y, and Z channels using the following formula:

$$projection_{\mathrm{i}}[m] = \frac{1}{N}\sum_{n=1}^{N} image_{\mathrm{i}}[m,n], \mathrm{i} = 1, \ldots, M. \tag{4.1}$$

where M is the number of pixels in the process direction and N is the number of pixels in the scan direction, which is perpendicular to the process direction; and $[m, n]$ represents the coordinates along the process direction and the perpendicular direction, respectively.

Then, the 1-D projections of the X, Y, and Z channels are converted to the CIE 1976 $L^*a^*b^*$ color space (with a 2° observer and D65 illumination). Following this, to make the banding more distinct from the image texture, we deduct its baseline from each 1-D projection signal. The baselines are obtained by applying a 1-D median filter with a size of 0.02 inches (7 pixels at 300 dpi) to each 1-D projection, respectively. Then, we compute the $CIE\ \Delta E$ and signed $CIE\ \Delta E$ using the three baseline-removed projections in CIE $L^*a^*b^*$ color space.

The formula for calculating $CIE\ \Delta E$ is:

$$CIE\ \Delta E = \sqrt{\sum_{c=L^*,a^*,b^*} (Original_c - Baseline_c)^2} \tag{4.2}$$

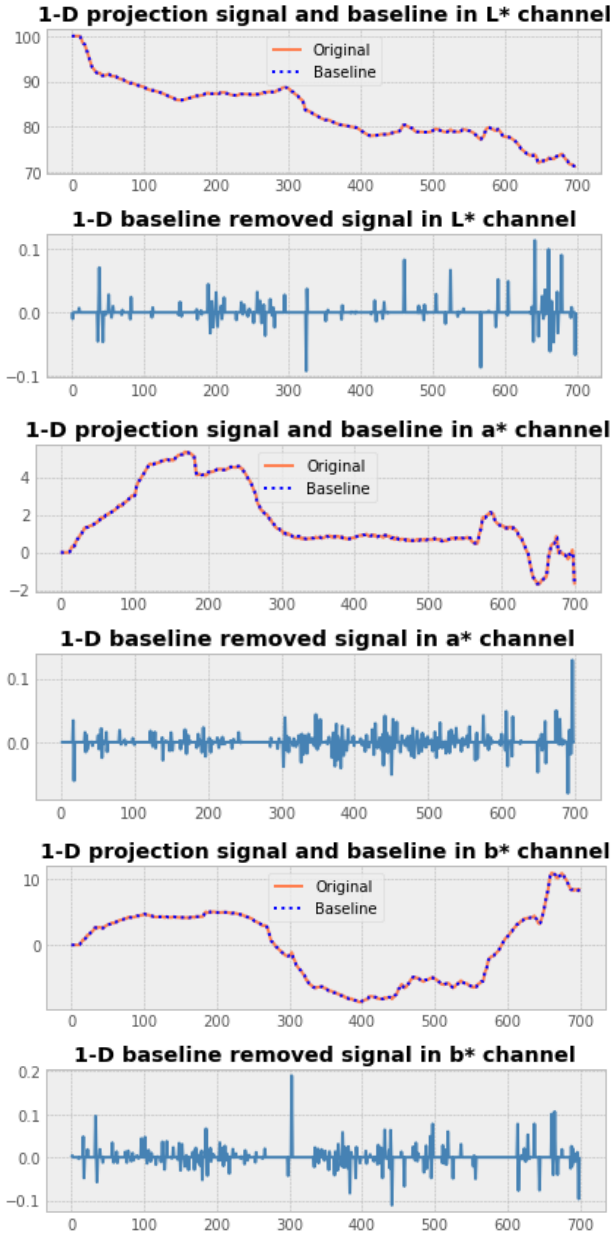


Figure 4.3. 1-D projection signals and baseline-removed signals. The units of the horizontal axes are pixels at 300 dpi.

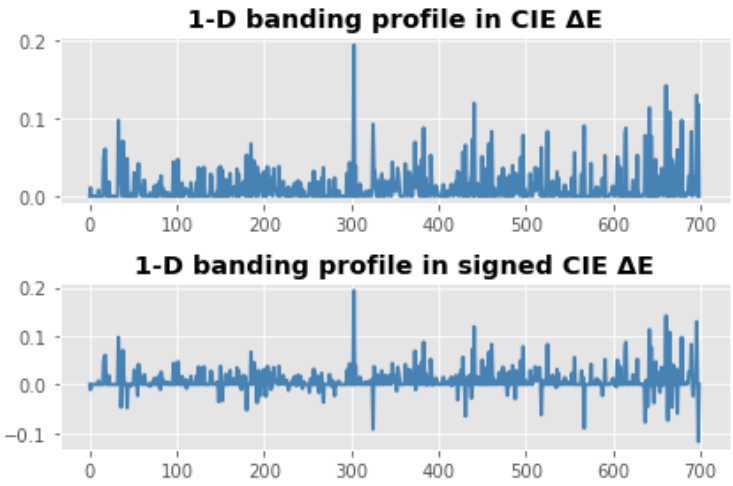


Figure 4.4. 1-D banding profile in CIE ΔE and signed CIE ΔE units. The units of the horizontal axes are pixels at 300 dpi.

Since the L^* channel indicates the lightness information, for the signal of L^*, if it is higher than the baseline, the banding in this position is a light banding; otherwise, it is a dark banding. $CIE\ \Delta\, E$ can indicate the color difference in the CIE $L^*a^*b^*$ color space, but because it is non-negative, the light/dark information about the banding is lost. We calculate the signed $CIE\ \Delta\, E$ by multiplying $CIE\ \Delta\, E$ by the sign of the L^* channel baseline-removed projection, which makes the color difference also include lightness information. Figure 4.3 shows the 1-D projection signals and baseline-removed signals in the L^*, a^*, and b^* channels, respectively. Figure 4.4 shows the 1-D banding profile in $CIE\ \Delta\, E$ and signed $CIE\ \Delta\, E$ units.

Once we get the $CIE\ \Delta\, E$, we need to find local maxima in $CIE\ \Delta\, E$. A local maximum is found by comparing adjacent values. Once the local maximum is determined, its center point, height, and width will be calculated, as shown in Figure 4.5. The blue curved lines are signed $\Delta\, E$. The red points signify the center position of each detected banding, which also is the local maximum value of the signed $\Delta\, E$. The green vertical line is the height for each peak, which is obtained by computing a vertical distance between the peak and its lowest side. The two yellow lines represent the total width and the width at half the height of each peak.

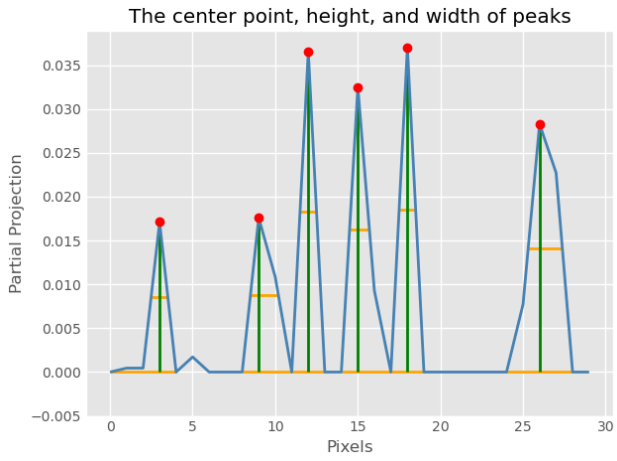


Figure 4.5. The center point, height, and width of peaks. The units of the horizontal axes are pixels at 300 dpi.

Then, we customize some thresholds for height, width, and distance between adjacent bandings according to the properties of the banding defect, and finally select local maxima that met the threshold as bands. We set the threshold for local maxima according to a previous work [44] and our several experiments. The thresholds for height, width and distance are shown in the Table 4.1 below and Figure 4.6 shows the peaks refined by the different thresholds.

Table 4.1. Thresholds for local maxima.

Height	$Mean_{height} + \frac{1}{2}Stddev_{height}$
Width	2 pixels
Inter-band Distance	5 mm (59 pixels at 300 dpi)

Image features may also lead to local maxima in the 1D projection. To distinguish the local maxima caused by banding defects from those caused by image features, we use the master image as a reference. Although we have performed a global image registration between the master image and the test image, we found that for a small ROI, there will still be a slight translation misalignment. Therefore, we use cross-correlation to locally register the test ROI and the master ROI pair, and calculate the signed offset of the test ROI relative to the master ROI in the process direction, which is denoted by o. Then, we extract banding profiles to obtain two sets of local maxima (master set and test set). Obviously, the master

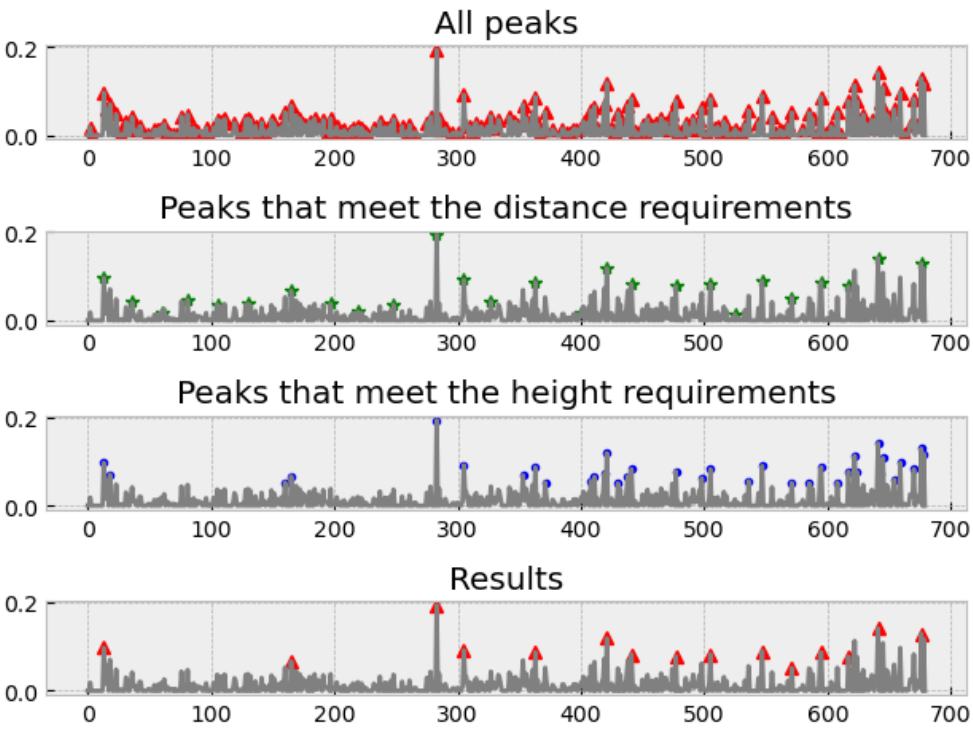


Figure 4.6. Peaks refined by different thresholds.

sets include all the local maxima caused by the image features, so we only select the local maxima in the test set, but not in the master set. For each local maximum position p_{test}[i] in the test ROI, if it is caused by image features, it must have a local maximum on the position p_{master}[j] of the master ROI. The relationship between the two positions is

$$position_{test}[\mathrm{i}] = position_{master}[\mathrm{j}] + o \tag{4.3}$$

where i indicate i-th peak in the test set and j indicate j-th peak in the master set

Through this step, we can obtain the local maximum caused by banding defects, which will not be affected by the image features or content.

Repetitive Interval Calculation

Periodic bandings are an important print defect, which can help us figure out problems that occurred in an internal printer rotating component. To define periodic bands and

estimate their repetitive interval, we use a search strategy to select periodic peaks from the isolated peaks detected in the previous section, estimate an approximate interval, and then optimize it until an accurate result is obtained.

We first compute all intervals between neighboring bands,

$$\text{i}interval_{\text{i}} = Band\ Position_{\text{i}+1} \text{ - } Band\ Position_{\text{i}}$$

and sort them by length. So we obtain a sorted array of lengths [i$interval_1$, i$interval_2$, ..., i$interval_{\text{i}}$]. If there are periodic bandings in the test image, at least three isolated bands are identified as periodic bands. Thus, the length of at least two intervals is the same or similar.

We set an upper bound (i$interval \times 1.1$) and a lower bound (i$interval \times 0.9$) for each interval of a different length, then repeatedly scan all intervals to find the bounded intervals, group them, and update their mean value as the new rough repetitive interval. This step is repeated until there is no change between the new rough interval and the old rough interval. After that, we use a new upper bound (i$interval \times 1.05$) and a new lower bound (i$interval \times 0.95$) of the updated interval, and re-scan all intervals to find all bounded intervals and update the mean value as the new interval. We compute the banding occurrence ratio (BOR) as expressed by the following formula (4.4) and select the interval with the largest BOR as the periodic interval.

$$BOR = \frac{n \times I}{h} \tag{4.4}$$

where n is number of intervals that fall in this bin, h is the height of the ROI and I is average value of the intervals in this bin.

Figure 4.7 shows a test image and its banding detection results. The yellow lines indicate the boundary between the three areas; the blue lines indicate the center lines of each area; the white projection around the blue line is the 1D projection of this area; the short red lines perpendicular to each blue line represent periodic bandings. We also calculated that the repetitive interval of the periodic bands of this image is 315.33 pixels (26.67 mm at 300 dpi).

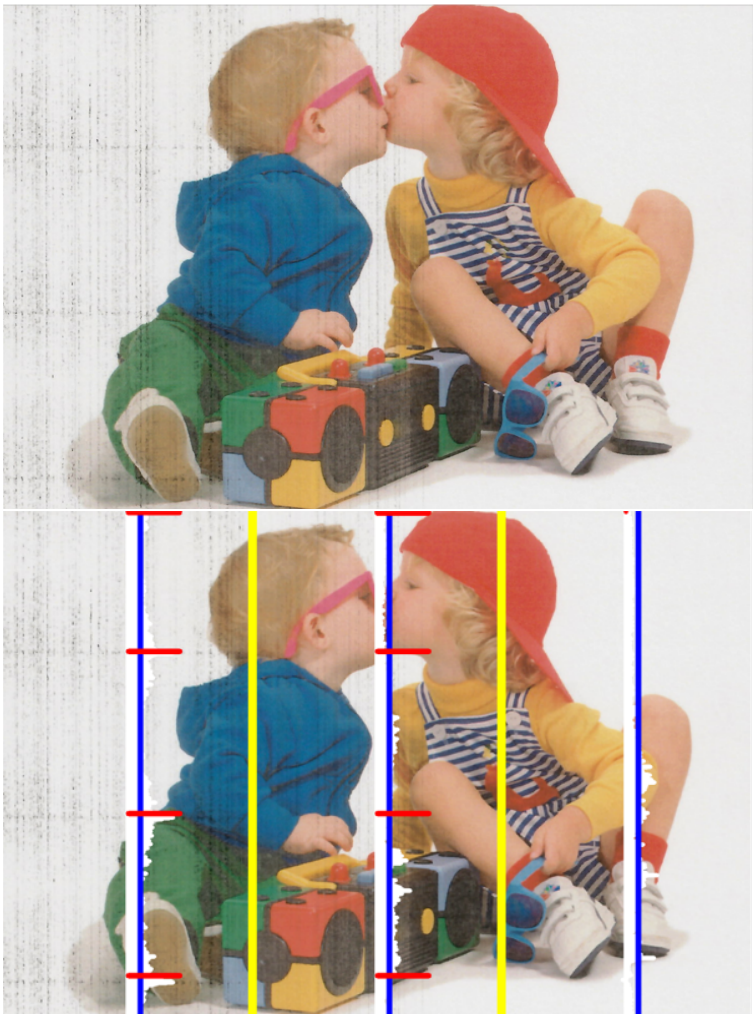


Figure 4.7. A customers' content page and its banding detection results.

4.2.3 Banding Feature Extraction and Quality Ranking Classification

In the previous section, we detected banding in the image or ROI. In this section, we describe the extraction of banding features, which represent the severity of banding. Then, we develop classification models that can predict the print quality based on the severity of banding.

Data preparation and banding feature selection

Our test images are provided by HP Inc. All test images are scanned pages of printed images, with various print defects of varying severity. We select 800 images (each image has a size of 3100 × 2400 pixels and a resolution of 300 dpi), and after the aforementioned pre-processing, they are aligned with their master image and cropped into ROIs. We concentrate on the raster ROIs, which are the regions that contain the texture, figures, and color changes; and we have a total of 956 ROIs from these images.

We set four levels from A to D to represent the print quality. The level A symbolizes almost perfect print quality, and the subsequent grades B, C, and D indicate that the banding defects are becoming more and more obvious and the print quality is getting worse and worse. Since we would like to use machine learning models to predict print quality based on perceptual assessment, three of our laboratory members labeled the ground truth manually. A set of ground truth samples is shown in Figure 4.8.

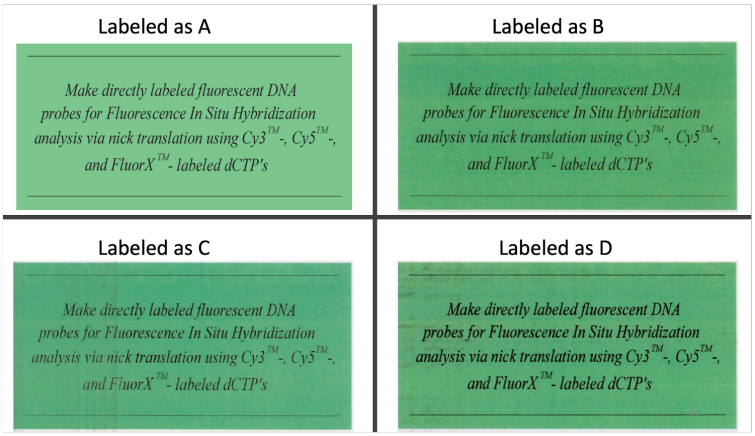


Figure 4.8. A set of ground truth samples. The reader is advised to zoom in to see the banding defects, which appear as horizontal lines.

For the banding defects, our pipeline can automatically detect the total number of bands in each ROI, and the position coordinate, length, width, and prominence of each band. Among them, the prominence shows the "local protrusion" of each band. It is defined as follows: let B denote the current peak, A denotes the peak immediately to the left of B, and

C denotes the peak immediately to the right of B. Let h_{i}, $\mathrm{i} = A, B, C$ denote the height of each of these peaks, then the prominence for peak B is defined as

$$prominence_B = h_B - \max(h_A, h_B, h_C) \tag{4.5}$$

The following Table 4.2 shows the components of the detectable banding feature vector.

Table 4.2. Components of the detectable banding feature vector for an ROI.

1.	The total number of bands
2.	The mean value of signed delta E
3.	The standard deviation of signed delta E
4.	ROI area
5.	The average height
6.	The maximum height
7.	The average width
8.	The maximum width
9.	The average prominence

As detailed above, we have calculated various features, which are sufficient to define the shape and state of the banding. However, for machine learning models, too many input features often lead to model overfitting.

The ANOVA F-statistic [49] is a popular feature selection technique. In [50], researchers found that the traditional ANOVA F-statistic is proper for selecting features for classification modeling problems where the inputs are numeric and the predictor outputs are categorical. For our model, all banding features are numeric, and we would like to obtain four levels (from A to D) of predictors to show the print quality. Therefore, our classification modeling problem contains the proper inputs and outputs to use the ANOVA F-statistic for feature selection.

We use the Python scikit-learn library function f-classif [48] to implement feature selection. This function computes the ANOVA F-statistic of the given inputs and returns the feature importance score based on the F-statistic score and p-value of each feature. First, we randomly split our images into 60% training images, 20% validation images, and 20% test images, and pass them into our banding detection pipeline to compute feature vectors.

The validation image set is used for feature selection and model selection. The test data is used to analyze the performance of the final model.

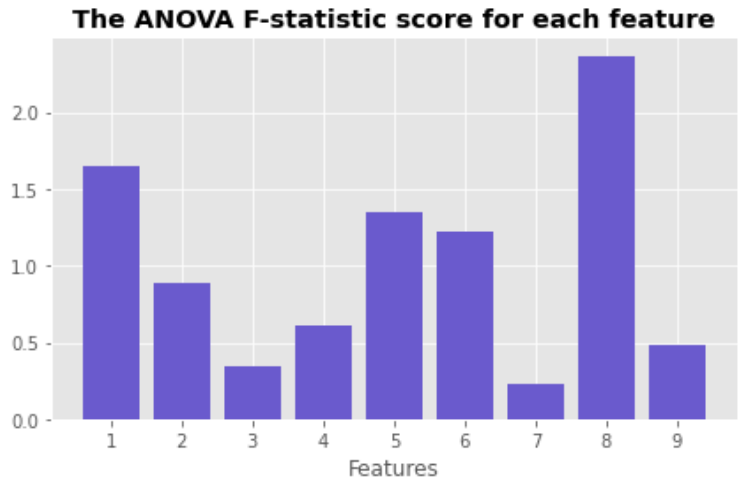


Figure 4.9. The ANOVA F-statistic score for each feature.

Figure 4.9 shows the feature importance score of each input feature. This clearly shows that Feature 1 (number of bands), Feature 5 (the average height), Feature 6 (the maximum height) and Feature 8 (maximum width) are the most important. In addition, Feature 2, Feature 4, and Feature 9 also have slightly higher scores. In order to decide the number of features most suitable for the model, we use logistic regression with an L1 penalty to iteratively estimate the accuracy of choosing 3 to 9 features. For the same validation data, the accuracy of selecting 3 features is 71.43%, the accuracy of selecting 4 features is 76.19%, and the accuracy of selecting 5 features is as high as 80.95%. The accuracy goes back down to about 70% if 6 and subsequent higher numbers of features are selected. Therefore, we chose the top 5 features ranked according to the importance score, which are the total number of bands in an ROI, the mean value of signed delta E, the average banding height, the maximum banding height, and the maximum banding width within a ROI.

Classification models

We use four popular machine learning models to build predictors: logistic regression, support vector machine, KNN and random forest. There are two heuristics to use binary

classification for multi-class classification: One-vs-Rest (aka OvR, One-vs-All or OvA), and One-vs-One (OvO) [51].

OvR means dividing a multi-class classification into multiple binary classification problems. Then, we train a binary classifier for each binary classification problem, and for each data sample, use the most confident model to make the prediction. For our case, the multi-class classification problem can be divided into four binary classifications, as shown below:

Binary classification problem 1: A vs [B,C,D]

Binary classification problem 2: B vs [A,C,D]

Binary classification problem 3: C vs [A,B,D]

Binary classification problem 4: D vs [A,B,C]

OvO also splits the multi-class classification into binary classification problems, but uses a one-to-one strategy. Compared with the OvR, OvO requires more binary models. The formula used to calculate the number of binary models is as follows:

$$\#\ of\ binary\ models = \frac{NumClasses(NumClasses1)}{2} \tag{4.6}$$

So, our problem with four types: A, B, C, and D, can be divided into several binary classifications as follows:

Binary classification problem 1: A vs B

Binary classification problem 2: A vs C

Binary classification problem 3: A vs D

Binary classification problem 4: B vs C

Binary classification problem 5: B vs D

Binary classification problem 6: C vs D

Each binary classifier predicts a class label. When we input the test data into the classifier, we will get the final result based on the majority count of the outputs of all the binary classifiers.

4.3 Results

We mentioned that we use a random splitting tool to randomly split the data sets into 60% training data, 20% validation data, and 20% test data.

For skewed classes, accuracy cannot sufficiently indicate the performance of the model. Two supplementary evaluation scores, balanced accuracy [52], and F1 score [53] are computed based on the test data, to avoid over-performance estimation of unbalanced data sets.

$$Accuracy = \frac{TP + TN}{Total\ samples} \tag{4.7}$$

$$Balanced\ Accuracy = \frac{1}{2}(\frac{TP}{TP + FN} + \frac{TN}{TN + FP}) \tag{4.8}$$

$$F_1 = \frac{TP}{TP + \frac{1}{2}(FP + FN)} = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.9}$$

$$\begin{cases} Precision & = \frac{TP}{TP+FP} \\ Recall & = \frac{TP}{TP+FN} \end{cases} \tag{4.10}$$

where, TP = number of true positives;
TN = number of true negatives;
FP = number of false positives;
FN = number of false negatives.

The following Table 4.3 shows the models we selected and the corresponding accuracy, balanced accuracy, and F1 results. The best decision is made by the random forest. We conclude that random forest is based on ensemble learning and introduces randomness, so that the model can handle outliers and solve the problem of data imbalance, thus it better adapts to our data. In addition, all the prediction results on the test dataset have notable performance, which can help us quickly inference to predict the print quality.

Table 4.3. Score results for various classifiers.

Model	Accuracy	Balanced accuracy	F1
LR(OVR)	0.81	0.75	0.77
SVM (OVO) (RBF)	0.82	0.78	0.80
SVM (OVR) (RBF)	0.82	0.78	0.80
KNN	0.86	0.83	0.85
Random Forest	0.91	0.85	0.90

4.4 Conclusion

In this chapter, we developed a banding processing pipeline and print quality classifier to diagnose printer defects and evaluate print quality. Our pipeline includes banding profile extraction, periodic banding interval estimation, and print quality classification. Our results show that banding defects can be automatically detected, the periodic interval of periodic banding can be estimated, and higher accuracy, balanced accuracy, and F1 score of the quality predictors that we developed can be obtained.

5. MEASURING CMYK COLOR PLANE MISREGISTRATION FROM SCANNED PRINTED CUSTOMER CONTENT IMAGES

5.1 Background and Previous Work Review

Misregistration of the color planes is a significant issue in printing technologies. Due to the fact that individual colorant planes are printed separately and independently, there will be color shifts in the printed image if there is no strict registration between the individual colorant planes. This limitation is particularly common with laser, electrophotographic printing technologies. The problem is illustrated in Figures 5.1 and 5.2. The right image in Figure 5.1 demonstrates that the magenta plane is not properly registered in the horizontal direction, and when compared to the reference image on the left, we can clearly see the impact of this issue on the image's boundaries. Additionally, in Figure 5.2, cyan moves downward in the right image, particularly around the fingers. This is due to the cyan plane being misregistered vertically.

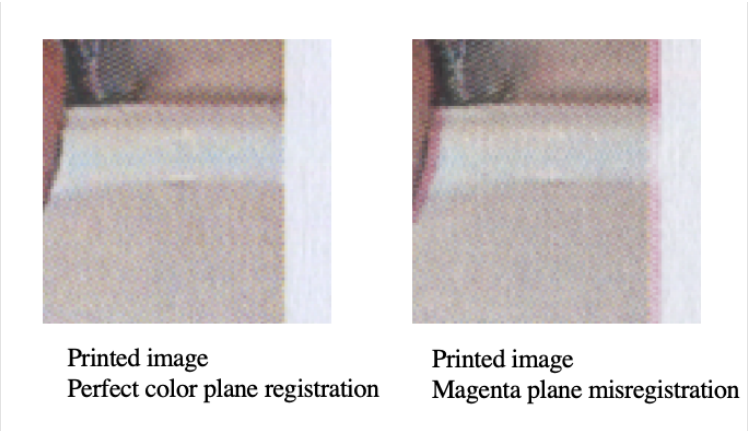


Figure 5.1. Illustration of color plane misregistration: the Magenta plane of the right image moved in the horizontal direction.

The traditional solution to this problem, which is used in offset lithographic printing, where the printing plates are created using extremely high-resolution image-setters, is to

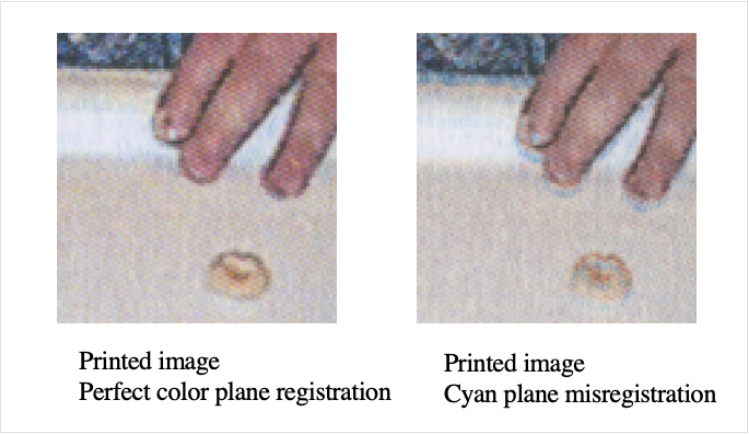


Figure 5.2. Illustration of color plane misregistration: the Cyan plane of the right image moved in the vertical direction.

rotate the screens to the most separated angles possible. However, achieving the desired angles with digital printers, which have a much lower spatial resolution, is difficult.

Oztan, Sharma, and Loce addressed the issue by developing a quantitative method for determining the color difference [54]. When registration errors occur, their approach calculates the change in the fractional area of Neugebauer primaries and then uses it to predict color shifts. They established registration insensitivity conditions by identifying instances in which the fractional coverage of each Neugebauer primary remains constant in the presence of registration errors.

Kim and Chen define this issue in terms of the visual appearance of the halftone microstructure [55]. They concentrated on determining the sensitivity of halftone screens to registration errors and quantifying the effect of registration errors.

While all the previous studies were beneficial to this question, they were primarily focused on the hardware mechanical level, or halftone patterns and images, which cannot be used directly on printouts. While some methods for measuring this problem on printouts do exist, they typically necessitate the creation of a uniform content image or a specific test image. Our contribution is that we will directly measure color plane misregistration using the customer's content images, which means we will have no prior knowledge of image content/texture, halftone images, or print quality, allowing the processing pipeline we designed to be more widely used by users or machines. Furthermore, we not only identify the misregistered

color planes, but also estimate their direction and magnitude, which allows us to detect the problem more thoroughly and accurately.

5.2 Methodology

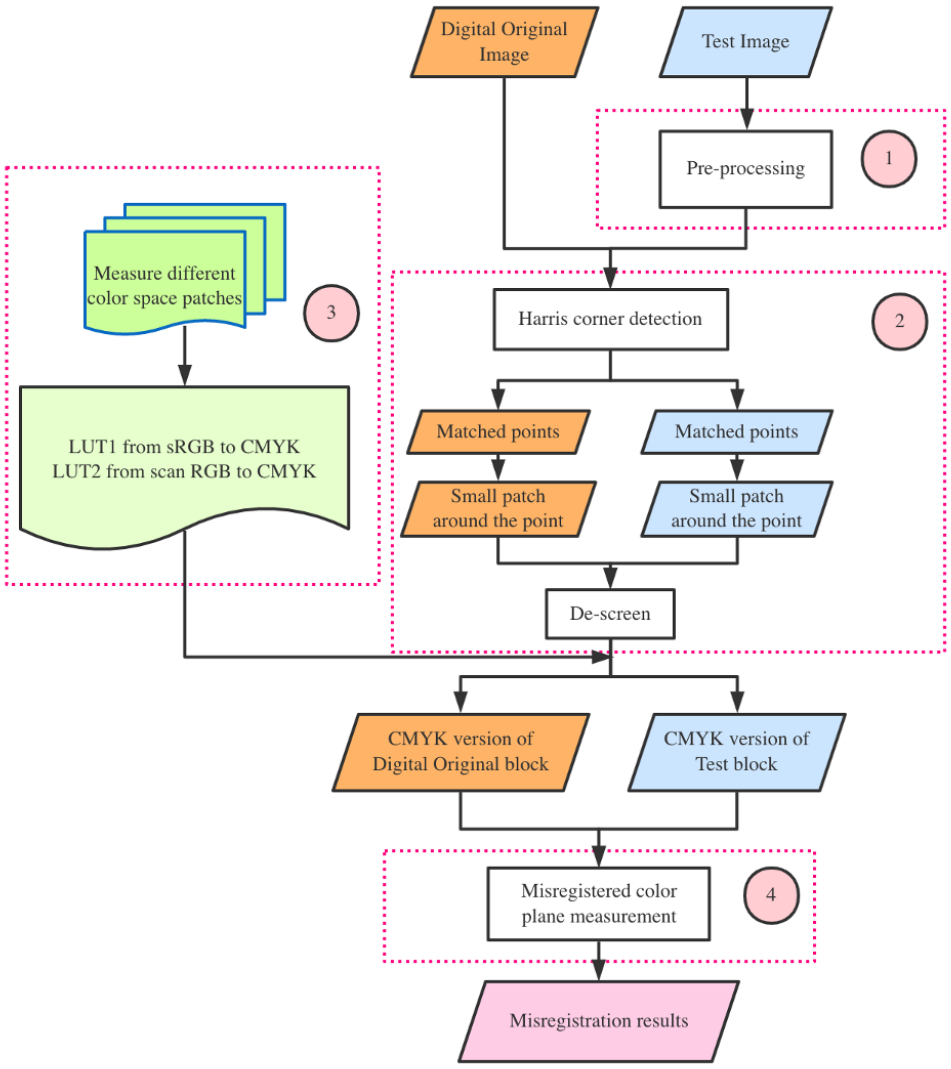


Figure 5.3. Overall pipeline for color plane misregistration analysis.

Our overall framework is depicted in Figure 5.3. Along with the scanned printed image of the customer content page (hereinafter referred to as the scanned image), our input contains

the digital original image of the customer content page (hereinafter referred to as the digital original image). The digital original image is the image that was sent to the printer and is used as a reference to avoid confusion between color plane misregistration and the original texture of the image.

We first conduct preprocessing on the scanned image, which includes image resizing, edge skew correction, image registration, and image slicing. Following that, to ensure that the entire image is measured uniformly, we use keypoint extraction and feature matching [56] on each small region pair (scanned region and digital original region) to identify aligned point pairs, and then use the aligned point pairs as the center to extract the surrounding pixels to form aligned block pairs. We filter out block pairs that are not strictly aligned by computing the SSIM (Structural Similarity Index) [57] and setting a threshold. To remove halftone dots in different orientations (horizontal and vertical), we de-screen these aligned blocks using different 1-dimensional Gaussian filters.

Following that, we need to convert the block pair's color space, which requires enough data to generate LUTs (Look Up Tables). We start color space conversion by creating three color test pattern pages in three distinct color spaces (sRGB, scanned RGB, and CMYK). Each color test pattern pages is composed of over 1000 distinct color blocks, as shown in Figure 5.5. We take three lists of 1000 corresponding color values (RGB values list of RGB patch, RGB values list of scanned RGB patch, and CMYK values list of CMYK patch), then develop two mappings using tetrahedral interpolation [58], and create two LUTs. One LUT is a mapping from sRGB to CMYK for digital original blocks, while the other is a mapping from scanned RGB to CMYK for scanned blocks.

Our block pairs are now in CMYK space as a result of the above LUT mapping. To obtain the final color plane misregistration result, we use two strategies. One strategy is to calculate the MSE(Mean-Square Error) of the CMYK values in the horizontal and vertical directions, while the other strategy is to directly calculate the cross-correlation of the block pairs in the C, M, Y, or K channel.

The following sections will explain the details of each function.

5.2.1 Pre-processing

In general, the scanned image will be smaller in size, and resolution than the digital original image due to the printing and scanning processing. In addition, due to the possibility of paper skew during scanning, the scanned image may have irregular margins.

An example digital original image is shown in Figure 5.4(a), with a size of 6912 $\times$ 4768 pixels at a resolution of 600 dpi in sRGB space. Our initial scanned image, which is 3456 x 2384 pixels at 300 dpi and has uneven margins, is shown in Figure 5.4(b). Prior to image alignment, we trim and correct the skew margins introduced during the scanning process to eliminate the irregular margins, as illustrated in Figure 5.4(c). Then, using bicubic interpolation, the size and resolution of the test image are adjusted so that the scanned image is the same size as the digital original image, as illustrated in Figure 5.4(d).

We assume for our scanned images that the geometric transformation involves only a small skew angle rotation and a small translation along the x-axis and the y-axis, and thus we want to find several matched key points in order to compute the optimal transformation. To save computation, we convert RGB images to grayscale and then resample them at a 1/3 downsampling rate. Following that, we use histogram matching to match the gray values in the scanned image to those in the digital original image. Then we use Harris Corner detection to extract key points from the digital original image and scanned image, respectively. The feature descriptor is built using the 31 $\times$ 31 local areas that surround each key point. We find the best-matched key point pair by minimizing the sum of squared differences (SSD) and computing the ratio SSD. The optimal transformation is then determined by leveraging the Maximum Likelihood Estimation Sample Consensus (MLESAC). We use this transformation to transform the scanned image so that it aligns with the digital original image, as shown in Fig. 5.4(e).

After obtaining the aligned digital original and scanned image pair, we divide each image evenly into 100 small blocks to make the color shift more visible. Each block is a tenth of the original image's width and height. To ensure the alignment's accuracy, we perform block

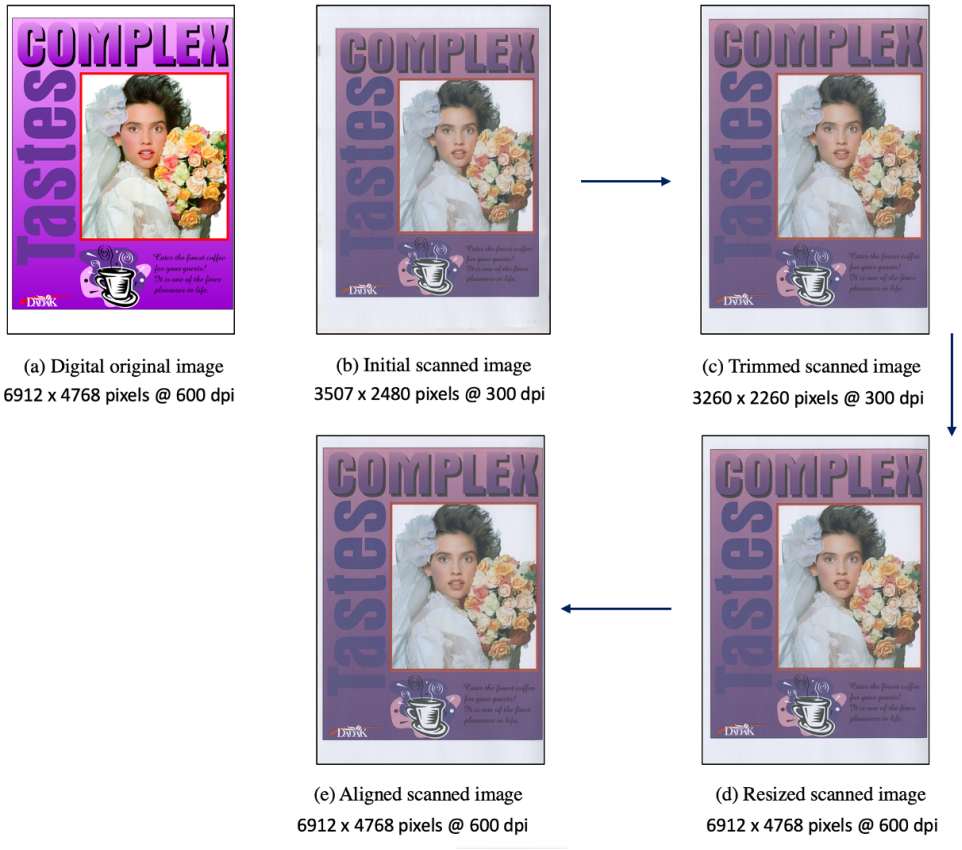


Figure 5.4. Digital original image, original scanned image and pre-processed scanned image.

registration using the aforementioned method to generate the aligned digital original and test block pair.

5.2.2 Color Space Conversion

As mentioned previously, our scanned image is in scanned RGB space, the digital original image is in sRGB space, and the misregistered color plane is in CMYK space. As a result, two LUTs are required to convert the color space between sRGB and CMYK, as well as between scanned RGB and CMYK.

We create a color page with 1000 distinct color blocks that are ordered by the arithmetic difference of the three RGB channels. We obtain this color page in three different color spaces: sRGB, scanned RGB, and CMYK, as shown in Figure 5.5. We measure the start,

end, length, and width of each small color block on the color page and use the average color values of the 0.6 Height × 0.6 Width area in the center of each color block to indicate the color. As a result, three lists containing 1,000 corresponding color values are created. The RGB values for the sRGB and scanned RGB pages are measured and recorded, as are the CMYK values for the CMYK pages.

Then, using tetrahedral interpolation, we create two LUTs from these data: one for converting the digital original image from sRGB to CMYK and another for converting the scanned RGB image to CMYK. As illustrated in Figure 5.6, prior to performing tetrahedral interpolation, we perform trilinear interpolation to augment the data.

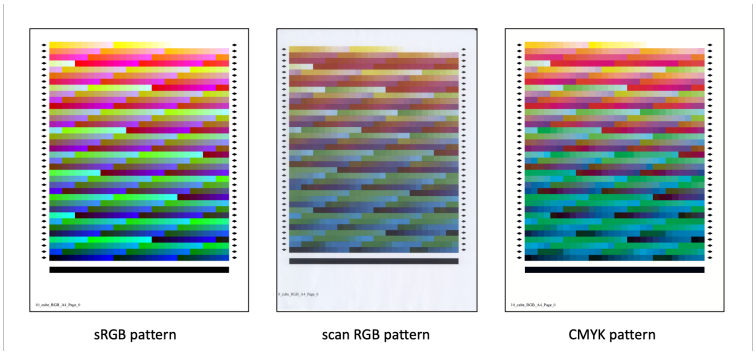


Figure 5.5. The test pages in three different color spaces.

5.2.3 Image De-screening

De-screening is required to remove halftone patterns from the input scanned image. While Gaussian filtering is commonly used to de-screen images, it has the potential to blur image details. We want to avoid weakening any useful details when measuring color plane misregistration, so we divide the problem into two problems in different directions. We de-screen only vertically when measuring horizontal color plane misregistration, and only horizontally when measuring vertical color plane misregistration.

This also decomposes the two-dimensional Gaussian filter into two one-dimensional filters. After conducting several tests, we determined that 15×1 and 1×15 1D Gaussian filters were more suitable for horizontal and vertical descreening, respectively, of our images. Figure 5.7 illustrates the performance of our descreening methods.

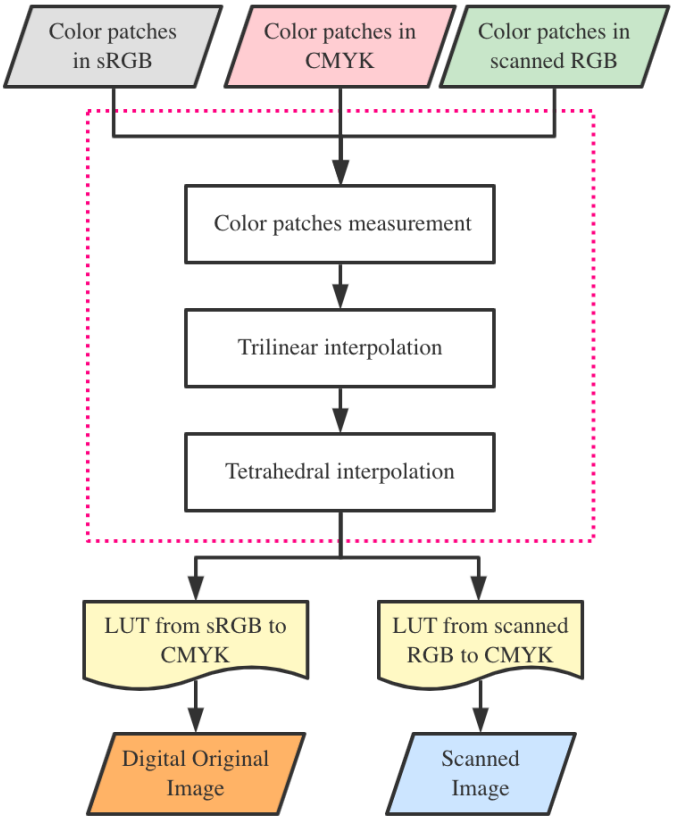


Figure 5.6. Framework for color space conversion.

5.3 Color Misregistration Measurement and Results

To obtain the final color plane misregistration result, we use two strategies. One strategy is to directly calculate the cross-correlation of the block pair in the C, M, Y, or K channels, while the another strategy is to calculate the MSE of the CMYK values in the horizontal and vertical directions.

5.3.1 Measurement using Cross-correlation

The first technique starts with Fourier transformation. We transform the block pairs to the Fourier domain, use element-wise multiplication of the transform of one block with the complex conjugate the of the transform of the other block to compute cross-correlation in this domain, and then perform an inverse FFT to obtain the cross-correlation. Then we

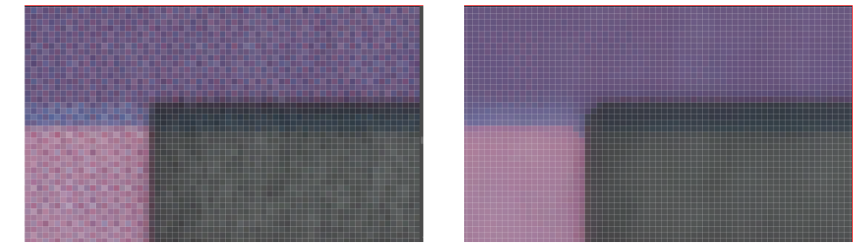


Figure 5.7. Zoom-in detail in the original test image and de-screened test image.

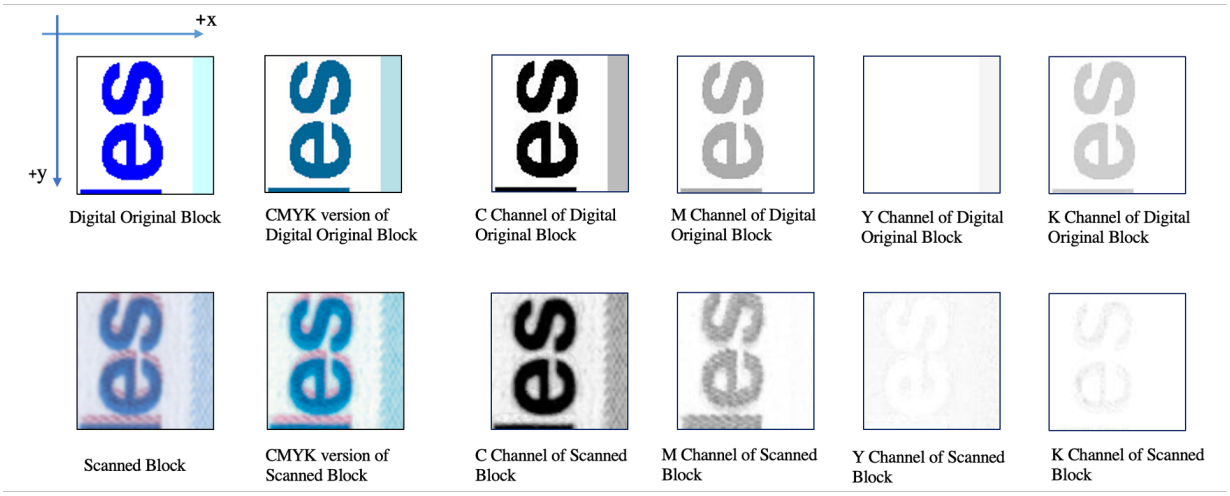


Figure 5.8. CMYK channels of digital original block and scanned block pair.

locate the maximum value of the cross-correlation and apply it to the original block of pixels to get the x and y axis offsets.

Table 5.1. Color plane shift results of Figure 5.8.

The amount of color plane shifts (pixel value)				
	Cyan	Magenta	Yellow	Black
x-axis	-1	0	0	-1
y-axis	0	-6	0	0

If color plane misregistration occurs, it indicates that the CMYK version of the scanned block is offset from its digital original block. As shown in Figure 5.8, from left to right are the digital original block (Row 1) and the scanned block (Row 2), the CMYK color space

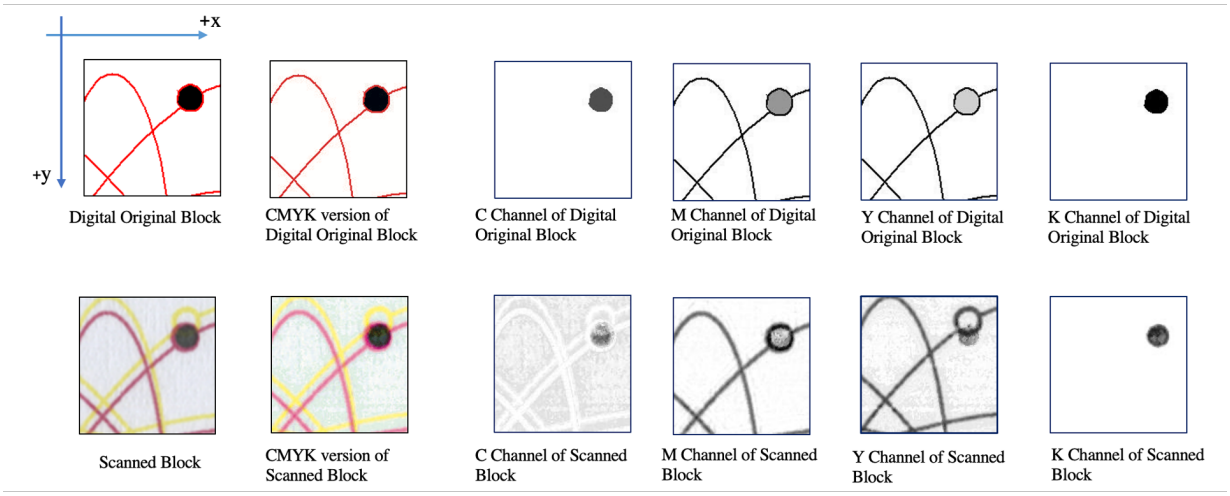


Figure 5.9. CMYK channels of digital original block and scanned block pair.

version of the digital original block and scanned block, and the block in the four channels of C, M, Y, and K, respectively. Table 5.1 shows the offsets of the scanned image relative to the digital original image based on their cross-correlation for each of the CMYK colors. Because it is imperceptible to the human eye when the offsets are less than or equal to 2 pixels, we collect only offsets larger than 2 pixels. From the results Table 5.1, the magenta plane has been shifted by 6 pixels in the -Y direction. This is our color plane misregistration measuring result, which is identical to the visual observation.

Table 5.2. Color plane shift results of Figure 5.9.

The amount of color plane shifts (pixel value)				
	Cyan	Magenta	Yellow	Black
x-axis	2	0	1	1
y-axis	1	1	-16	1

Additional results are shown in Figure 5.9 and Table 5.2, which indicate that the yellow plane has shifted 16 pixels in the -Y direction, which matches the visual observation.

5.3.2 Measurement using MSE

The feature-matched small block pair typically contains key points, implying that it contains edges or texture variations. Thus, for a small block pair with a regular texture

(rectangle or square), rather than performing color space conversion, Fourier transformation, and cross-correlation on the entire image, we can derive a color plane misregistration conclusion by computing color shift along a horizontal line and a vertical line that cross the edges.

We convert the digital raw blocks to grayscale and then convert them to a binary image using Otsu's binarization. Otsu's method derives the optimal global threshold from the image histogram. Following that, we do morphological opening (erode the image and then dilate) in order to eliminate symbols and details from the binary image while retaining the stronger contours. Then, using the Hough transform, we find horizontal and vertical lines. We refer to this open source function to implement Otsu's method and Hough transform [78]. Figure 5.10 shows the edge detection processing.

Figure 5.11 depicts an example of a regular texture (rectangle or square) that meets the MSE calculation requirements. Rather than performing color space conversion, Fourier transformation, and cross-correlation on the entire block pair, we only apply color space conversion to the pixels on the horizontal and vertical lines indicated by the arrows. Then, using the following formula, we compute the individual and sum MSE of CMYK on the lines.

$$\mathrm{i}ndMSE_{\mathrm{i}} = (\mathrm{i}_{digital} - \mathrm{i}_{scanned})^2 \tag{5.1}$$

where i is one of C, M, Y and K.

$$\begin{aligned} sumMSE = (C_{digital} - C_{scanned})^2 + (M_{digital} - M_{scanned})^2 \\ + (Y_{digital} - Y_{scanned})^2 + (K_{digital} - K_{scanned})^2 \end{aligned} \tag{5.2}$$

When color plane misregistration appears, the MSE of CMYK will significantly increase. We use a sample Figure 5.11 to illustrate the results measurement. The horizontal and vertical MSE values for this sample are shown in Figure 5.12. MSE values are scaled in the 0-1 range. We can see that the vertical MSE is always small and has no distinct peak, while the horizontal MSE has a distinct peak near the center, which corresponds to the

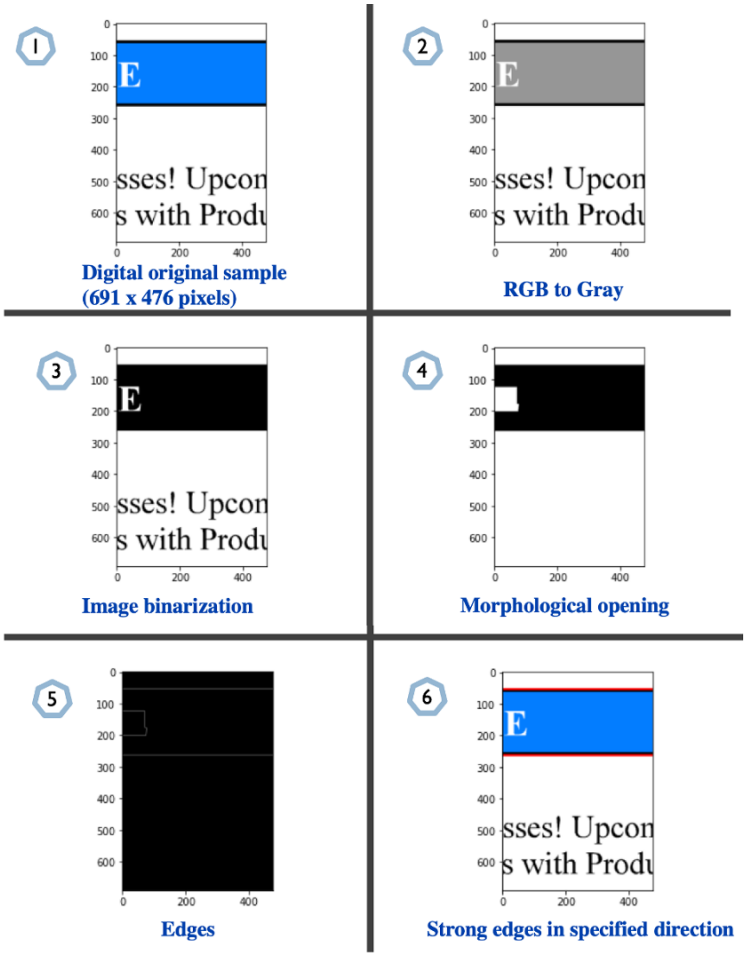


Figure 5.10. Sample results for Hough transform (Note: the edges of the last sub-image are highlighted with 6-pixel-wide red lines; the actual edges are 1 pixel wide lines.

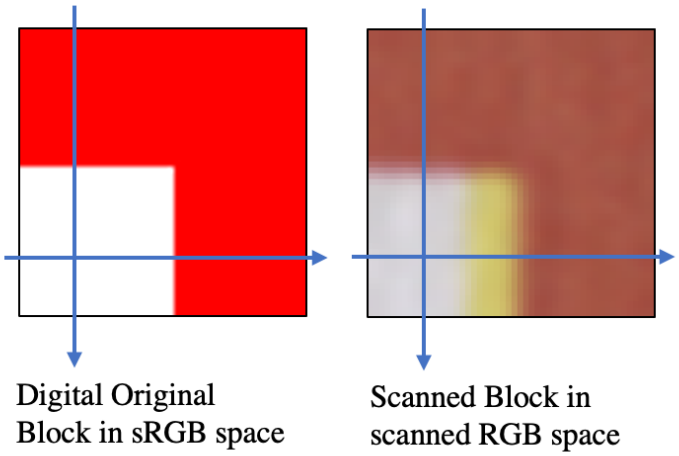


Figure 5.11. A digital original block and scanned block pair.

yellow plane misregistration we see in the scanned block. We can determine the magnitude of the color plane misregistration by calculating the width of this peak, which is 9 pixels for this sample. The misregistered color can be determined by computing the four MSEs and determining the maximum value, which is Yellow, as shown in Table 5.3.

Table 5.3. MSE of each color plane.

MSE			
Cyan	Magenta	Yellow	Black
0.0075	0.019	0.972	0.002

5.4 Conclusion

In this chapter, we designed and developed a processing pipeline for measuring color plane misregistration. The processing pipeline can automatically determine the direction, magnitude, and identity of the misregistered color plane for a scanned printed customer content image. We developed two measurement strategies: one measured using cross-correlation achieves subpixel accuracy and is thus preferred for high-resolution printed images. Although the other strategy, MSE calculation strategy is slightly less accurate than the former, it does

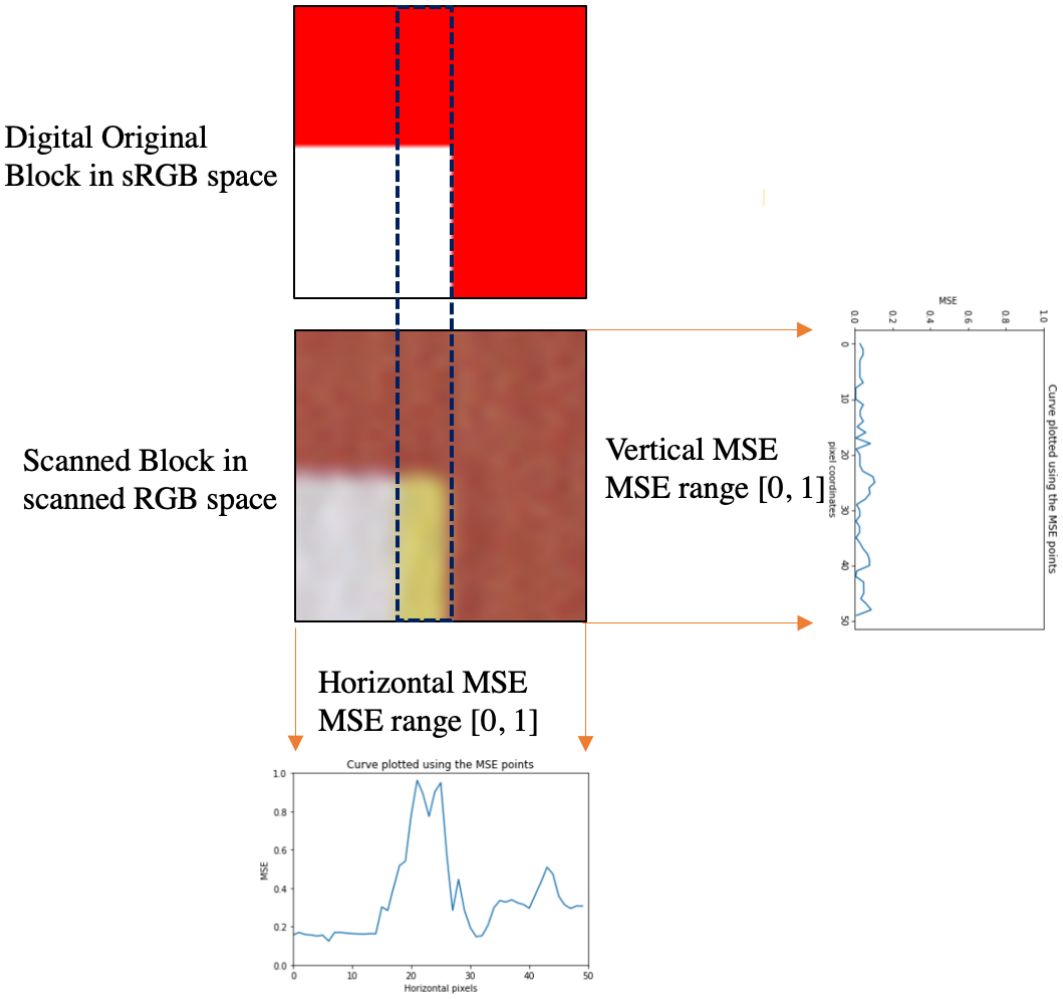


Figure 5.12. MSE plots of the block pair.

not require the entire image's color space conversion, nor the entire image's FFT and IFFT transformation, which reduces the amount of computation. The entire design pipeline can be directly used on the printed output without prior information on image content, allowing it to be widely used for printer troubleshooting.

6. MIX-LOSS TRAINED BIAS-REMOVED BLIND IMAGE DENOISING NETWORK

6.1 Background and Previous Work Review

Due to the limitations of various recording devices, images are sensitive to random noise during acquisition. Noise is signal distortion that impedes image observation and information extraction. Thus, as a fundamental topic of image analysis and processing, image noise suppression aids our understanding of image statistics and processing.

There are numerous noise sources during the imaging process, including photon-shot noise, photon-to-charge conversion noise, analog-to-digital conversion noise, hot pixels, read noise, compression artifacts, and so on. The most frequently concerned noises in the literature are Additive White Gaussian Noise (AWGN), impulse noise (salt-and-pepper noise), Poisson noise, and speckle noise. Impulse noise, speckle noise, and Poisson noise are caused primarily by defective manufacturing, bit errors, and an insufficient photon count. AWGN is most commonly seen in analog circuitry during image acquisition and transmission. In most image denoising tasks, the noise is assumed to be additive white Gaussian noise (AWGN).

Digital images can be viewed as a matrix or a two-dimensional signal containing gray level intensity or color channel values: $(x, y(x))$, where x is the index of a pixel; $y(x)$ is the value indicating the pixel's intensity at a given location in a grayscale image; or indicating a three-value array representing the red, green, and blue channels in a color image. An image AWGN noise model can be thought of in the following manner:

$$y(x) = m(x) + n(x) \qquad x \in Z^2 \tag{6.1}$$

where $m(x)$ represents the original image, $y(x)$ represents the noisy observation and $n(x)$ represents the Gaussian noise with zero mean and standard deviation σ. The denoising problem requires determining a function $f : R^N \rightarrow R^N$, that can accurately estimate the original image $m(x)$. This issue is typically resolved by minimizing the mean squared error :

$f = argmin_g E||m(x) - g(y(x))||^2$. In deep learning, the denoising function g is parameterized by the weights of the network, so the optimization is performed over these parameters.

Image denoising algorithms have made significant strides with the development of imaging technology over the last few decades. Denoising algorithms are classified into the following categories according to their type [63]: Spatial domain filtering, which includes average filter, Weiner filter, bilateral filters, non-local mean and so on. Transform domain thresholding, which makes use of the Fourier transform (FT), fast Fourier transform (FFT), discrete cosine transform (DCT) and discrete wavelet transform (DWT). The most prevalent approach in this category is color block matching 3D collaborative filtering (CBM3D)[64]. Apart from these, additional well-established denoising approaches include random field based methods Markov random field (MRF) and hidden Markov models (HMM) [65]; as well as sparsity based methods such as K-singular value decomposition and convolution sparsity representation (CSR) and the dictionary learning method [63].

Recently, as network architectures became more flexible, deep learning has been widely used to deal with image restoration problems, resulting in significant advances in image denoising. The convolutional neural networks (CNNs) were proposed and quickly gained popularity in image/video processing and low-level computer vision. As a result, CNNs were frequently used for AWGN denoising or feature extraction for removing real-world noise or hybrid noise. For addressing multiple low-level tasks by one model, a denoising CNN (DnCNN) [66] consisting of convolutions, batch normalization (BN), rectified linear unit (ReLU) and residual learning (RL) was proposed to deal with image denoising, super-resolution, and JPEG image deblocking. Taking into account the tradeoff between denoising performance and speed, a color non-local network (CNLNet) [67] efficiently reduced image noise by combining non-local self-similarity (NLSS) and CNN. In terms of blind denoising, a fast and flexible denoising CNN (FFDNet) [68] used different noise levels and the noisy image patch as the input of a denoising network, thus ensuring that the denoiser can adapt to all noise levels; and a convolutional blind denoising network (CBDNet) [69] removed the noise from the given real noisy image by two sub-networks for blind denoising. To address the lack of clean reference images, Generative Adversarial Network (GAN) based methods

(e.g. GCBD) and CNN based unsupervised/weakly supervised methods (e.g. Noise2Noise, DIP, Noise2Self) are denoising methods that do not require clean images [70].

Deep learning-based image denoising has been extensively studied for denoising tasks with many aspects such as blind, unsupervised, universal and perceptually oriented. Our research focuses on a blind, bias-removed, mix loss optimized, and perceptually oriented image denoising task. Therefore, we will discuss related work and progress in these aspects.

6.1.1 Bias-removed Network

The idea of a bias-free CNN [71] inspired us. It points out that CNN feedforward neural networks with ReLUs are piece-wise affine, implying that the net network bias fluctuates wildly outside the training range, leading to the CNN overfitting to the noise levels within the training range and underfitting to the noise levels outside of the training range. The authors suggested that the issue could be ameliorated by removing additive (bias) terms at each stage of the network, resulting a bias-free CNN network (BF-CNN).

Feedforward neural networks with rectified linear units (ReLUs) are piecewise affine since each of these is affine. For a noisy input image y, the function f_1 computed by a denoising neural network with bias may be written:

$$f_1 = W_L R(W_{L-1}...R(W_1 y + b_1) + ...b_{L-1}) + b_L = A_n y + b_n \tag{6.2}$$

where W_n is the weight of the convolutional layers, b_n is the bias (additive constants), R is the activation function ReLU, $A_n \in R^{N \times N}$ is the Jacobian of f_1 evaluated at input y, and b_n represents the net bias. Also, a bias-removed denoising neural network can be derived as follows:

$$f_2 = W_L R(W_{L-1}...R(W_1 y)) = A_n y \tag{6.3}$$

The work [71] demonstrates a viewpoint for several popular deep models, including DnCNN, Recurrent CNN, UNet and DenseNet, that these biased convolutional networks overfit to noise levels within the training range, but underfit to noise levels outside of the

training range, because the net bias fluctuates wildly over the untrained noise level. This issue can be addressed by removing bias (additive terms) from each stage of the network, resulting in a bias-removed CNN. If bias-removed CNN has ReLU activations, the denoising map is locally homogeneous, and thus invariant to scaling. This property behaves as shown in the formula:

$$f_2(\alpha y) = \alpha f_2(y) \tag{6.4}$$

The bias-removed neural network has the scaling invariant property, which means that rescaling the input by a constant value simply rescales the output by the same amount, and this property is intuitively desirable for a denoising method that operates on natural images.

6.1.2 Loss Functions for Image Denoising

Different losses compute the similarity between the estimated image $g(y)$ and the clean image (ground truth) m in different ways. We list several important loss functions for image restoration tasks.

- Mean absolute error (MAE) or L_1 is defined as:

$$Loss_{L_1}(P) = \frac{1}{N} \sum |g(y(p)) - m(p)| \tag{6.5}$$

- Mean squared error (MSE) or L_2 is defined as:

$$Loss_{L_2}(P) = \frac{1}{N} (g(y(p)) - m(p))^2 \tag{6.6}$$

For Formulas (5) and (6), p is the index of the pixel, P is the patch, and N is the number of pixels in the patch; $g(y(p))$ and $m(p)$ are the values of the pixels in the processed patch and the ground truth, respectively.

- The Structural Similarity Index (SSIM) is a perceptual metric. SSIM is based on visible structures in the image. It is a perceptual metric used to quantify the image quality in terms of luminance, contrast, and structural similarity. SSIM is defined as:

$$SSIM(p) = \frac{2\mu_{g(y)}\mu_m + c_1}{\mu_{g(y)}^2 + \mu_m^2 + c_1} \frac{2\sigma_{g(y)m} + c_2}{\sigma_{g(y)}^2 + \sigma_m^2 + c_2} \tag{6.7}$$

where $\mu_{g(y)}$ and μ_m are the averages of $g(y)$ and m, respectively, $\sigma_{g(y)}^2$ and σ_m^2 are the variance of $g(y)$ and m, respectively, and $\sigma_{g(y)m}$ is the covariance of $g(y)$ and m. Also, $c_1 = 1 \times 104$ and $c_2 = 9 \times 104$ are two constants, which are used to stabilize the division with a weak denominator. The SSIM loss is defined as:

$$Loss_{SSIM}(P) = 1 - SSIM(p) \tag{6.8}$$

In the image restoration task, each loss function has its advantages and disadvantages. Although the loss layer is the primary driver of network learning, dealing with image restoration tasks is often accomplished by using L1 (MAE) or L2 (MSE). The loss function is typically set to the L2 for the majority of AWGN denoising tasks. However, L2 has a number of well-documented drawbacks, for instance, L2 has a weak correlation with perceived image quality by human observers [72]; L2 will significantly blur image details; and the use of L2 makes implicit assumptions about the noise, such as that the noise is additive and independent of the image, which is not suitable to real world noise.

The paper [72] focuses on the performance of various loss functions in multiple image restorations and proposes a mix loss function. Inspired by this work, we propose and develop a novel mix loss function that is suitable for our bias-removed denoising networks. We demonstrate that our denoised image quality outperforms those that were trained with conventional loss in terms of perceptual assessment.

6.1.3 Image Quality Assessment (IQA) Metrics

While assessing the perceptual similarity of two images is straightforward for humans, the underlying processes are regarded to be highly complex. The widely used image quality assessment measures, such as PSNR and SSIM, are far too simplistic to account for subtleties in human perception. Therefore, many researchers have studied image quality assessment metrics based on human perceptual similarity.

The work [74] summarizes and compares a variety of models for evaluating the quality of full-reference images. They tested 11 IQA models, which include objective, subjective, conventional, and deep learning based models. They proved that the Learned Perceptual Image Patch Similarity model (LPIPS) [73] score, which is computed automatically by a pretrained network, is positively correlated with human subjective evaluation in the image denoising task. Thus, we use PSNR, SSIM, and LPIPS to comprehensively evaluate the denoised image quality in our work.

6.2 Goal and Overall Frame

In this work, we focus on the CNN-based AWGN noise blind denoising task. We study the networks with and without bias, then train a bias-removed network and compare its denoising performance to that of biased networks across various noise levels to demonstrate its superiority on the denoising task. We investigate the impact of loss functions on the performance of the CNN-based image restoration network and design a mix loss that combines MSE and SSIM for our bias-removed network. We train a denoiser and compare its performance to that of traditional MSE loss functions in the AWGN denoising task to demonstrate the advantage of mix loss. Additionally, we utilize not only the conventional image quality assessment metrics SSIM and PSNR, but also the cutting-edge perceptual image quality assessment (IQA) metric LPIPS to evaluate our denoising results and comparison results, showing that our results perform well on both objective and subjective IQA metrics. We have made the following contributions:

- First, we develop a bias-removed image denoising network, demonstrating that it is capable of handling the AWGN denoising over a wide range of noise levels while training

over a very narrow range of noise levels. The bias-removed network achieves impressive denoising results on trained noise levels, and outperforms the state-of-the-art bias network on untrained noise levels.

- Second, we design a mix loss that combines MSE and SSIM, and use it as the loss function in our bias-removed network to further improve denoising performance. Our network is fully capable of denoising AWGN while retaining visually pleasing image details, so it outperforms the deep denoiser using a conventional L2 loss function in terms of perceptual image quality assessment.

- Finally, we study objective and subjective image quality assessment, and comprehensively evaluate our denoising results and comparison results using objective and subjective image quality assessment metrics.

6.3 Methodology

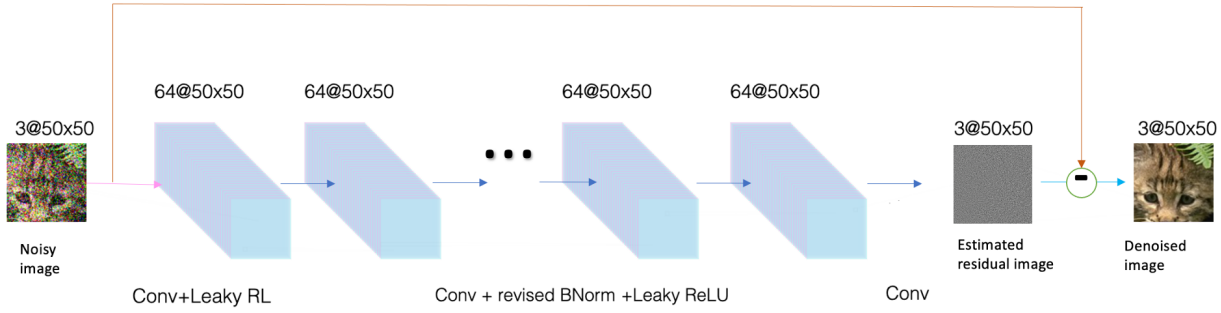


Figure 6.1. The network architecture of our denoiser. We adopt 17 convolutional layers for the main stem of the network, each consisting of 3×3 filters and 64 channels, revised batch normalization according to bias-removal, and a leaky ReLU activation function.

We propose a network designed for AWGN image denoising. Our network removes bias at each layer to achieve the benefits of bias-removed networks. Additionally, it implements a mix loss function to boost performance. AWGN noisy image and clean image pairs serve as inputs. The overall architecture is shown in Figure 6.1.

6.3.1 Network Architectures

We developed our network architecture based on the benchmarking network, Denoising CNN (DnCNN) [66]. DnCNN is composed of 17 or 20 convolutional layers, depending on whether the task is non-blind or blind. Each layer of DnCNN comprises 3×3 filters and 64 channels, batch normalization, and a ReLU activation function. It has a skip connection from the initial layer to the final layer.

We set the depth of our AWGN denoising network to 17 layers. This depth is used by DnCNN to train a non-blind denoiser, which has a relatively shallower depth and thus requires a smaller dataset. We also use this depth to train a non-blind denoiser, but our non-blind denoiser will have the ability to perform blind denoising. Our network includes the following attributes:

- Bias removed. We removed all additive constants at each layer of the network and made corresponding changes to the batch normalization function.
- Mix loss optimized. We design and implement a mixed function of L2 and SSIM, as indicated by the following formula.

$$mixLoss = \alpha Loss_{L_2} + (1 - \alpha) Loss_{SSIM} \tag{6.9}$$

  where $Loss_{L_2}$ represents the MSE function of pixel-by-pixel comparison, $Loss_{SSIM}$ represents SSIM Loss function we mentioned before, and α is the loss weight set to 0.6. We chose the value for this parameter so that the contribution of the two losses would be roughly balanced. Also, based on a published reference [72], the results were not significantly sensitive to small variations of α.

- Leaky ReLU used. We adapt the activation function to a leaky ReLU instead of the ReLU.

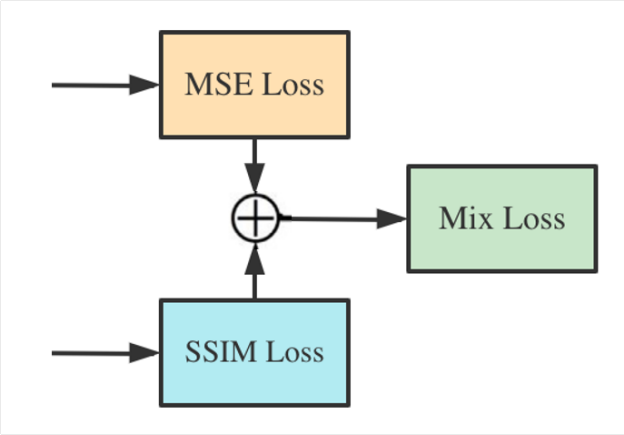


Figure 6.2. Mix Loss.

Thus, the network includes the first layer of convolutional layer and leaky ReLU, and then uses convolutional layer, revised batch normalization and leaky ReLU for layers $2 \sim (L-1)$. For the last layer, the convolutional layer is used to reconstruct the estimated residual image.

The denoiser trained from the above-mentioned network has a dual application: 1). It can be used to train blind denoisers directly, and the blind denoiser can produce cutting-edge denoising results for AWGN noisy images. 2). It can be used to train non-blind denoisers on limited noise ranges and smaller datasets, while the non-blind denoiser still maintains the ability of a blind denoiser to remove noise at any level, including levels beyond the training range. This attribute will be shown in detail in the subsequent results section.

6.3.2 Dataset

The training datasets are divided into two categories: grayscale images and color images. We used well-known datasets: The Berkeley Segmentation Dataset [75] includes the datasets BSD400, composed of 400 grayscale images; and BSD500, composed of 500 color images. The test datasets we used include grayscale image test datasets Set12 and Set68, and color image test datasets Set5 and Set14. All images are in png format.

We applied the following data augmentation and data pre-processing steps: 1). Using cubic interpolation, resize each image by different scales (1, 0.9, 0.8, and 0.7). 2). Flip or

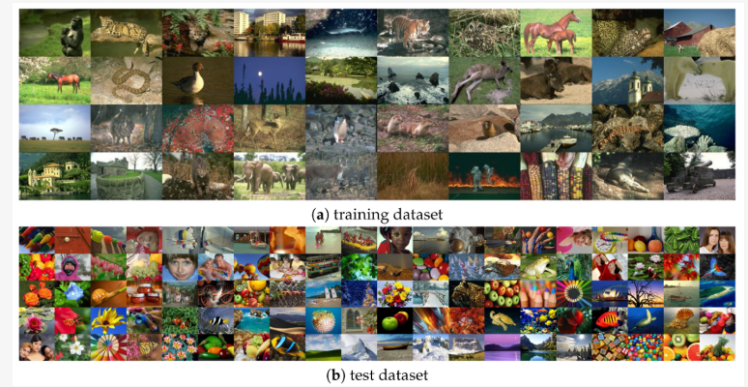


Figure 6.3. The Berkeley Segmentation Dataset.

rotate each image at random. 3). Divide each image into small patches of 50×50 pixels. 4). Divide training images into a training set and a validation set at random. 5). Save training, validation, and test sets in an h5 (Hierarchical Data Format 5) file.

6.3.3 Training

After the data augmentation, we input a total of 655,900 small color patches with a size of 50×50 pixels. The loss function is mix loss combined by MSE and SSIM. We initialize the weights using the Kaiming weight initialization method [76] and use the Adam Optimizer [77]. For Adam, the learning rate $= 1e^{-3}$, $\beta_1 = 0.9$, $\beta_2 = 0.999$, $\epsilon = 1e^{-8}$. We train on Nvidia Titan Xp graphics cards, check the Loss and PSNR per batch, and validate in the end of each epoch. We train a grayscale image denoiser on noise levels $\sigma \in [0, 30]$ using the grayscale dataset and a color image denoiser on noise levels $\sigma \in [0, 50]$ using the color dataset.

6.4 Results and Evaluation

In our work, we use PSNR, SSIM, and LPIPS to assess image quality. Numerous researchers have found that PSNR, which is derived from MSE, frequently contradicts human perceptual assessment. SSIM is a basic perceptually motivated metric, but it still represents

only a limited perceptual assessment. However, PSNR and SSIM remain the most mainstream IQA metrics in the image processing area. For PSNR and SSIM, higher scores indicate greater similarity between the two images (denoised image and ground truth). Additionally, we use LPIPS to assess the image quality. LPIPS, as previously stated, is a cutting-edge deep-learning-based perceptual assessment metric that quantifies image similarity in units of JND, and correlates positively with human perceptual assessment. For LPIPS, a lower score indicates a greater similarity.

The performance of the denoiser based on grayscale images, and the advantages of the bias-removed network is illustrated in Figures 6.4 and 6.5. The non-blind denoiser is trained at a given range of noise $\sigma \in [0, 30]$, and compared to a non-blind DnCNN pretrained in the same range. The two models are compared at four different levels of noise: $\sigma = 15$, $\sigma = 30$, $\sigma = 50$ and $\sigma = 80$. $\sigma = 15$ and $\sigma = 30$ are two test noise levels that fall within the training range, while $\sigma = 50$ and $\sigma = 80$ are two significantly higher noise levels that exceed the training range. For $\sigma = 15$ and $\sigma = 30$, as illustrated in Figure 6.4, the denoising performance of the two models is comparable because both models are trained on this noise interval. Our results are slightly lower on PSNR but remain competitive on SSIM and better on LPIPS. $\sigma = 50$ and $\sigma = 80$, as illustrated in Figure 6.5, are noise levels higher than the training range for both models. While the non-blind DnCNN is nearly incapable of removing noise at these levels, our model is still capable of doing so effectively. While our denoising performance is insufficient at $\sigma = 80$ noise level, as a non-blind denoiser with narrow training range and shallow network depth, its blind denoising capability is still impressive.

The performance of the denoiser based on color images is illustrated in Figures 6.6, 6.7 and 6.8, and Table 6.1. Our color image denoiser is trained at a given range of noise $\sigma \in [0, 50]$, and compared to a non-blind DnCNN are pretrained in the same range. We compared the two at $\sigma = 15$, $\sigma = 25$ and $\sigma = 50$ noise level, thus, they can all be regarded as blind denoisers. Table 6.1 shows the quantitative results for the test dataset Set14.

Our results, even though they are a little lower on PSNR, are competitive on SSIM and impressive on LPIPS. At various noise levels, our denoiser achieves the best LPIPS scores. Considering that LPIPS metrics have a higher correlation with human perceptual assessment, our results outperform DnCNN in subjective assessment. Some qualitative results on images

are shown in Figures 6.6, 6.7 and 6.8. We can see that our results preserve details while denoising. This is the benefit of using a relatively perceptual-aware SSIM in the loss function, which mitigates the over-smoothing property of MSE loss.

Table 6.1. Overall quantitative results compare with non-blind DnCNN on the same testset Set14. Our proposed method obtain better LPIPS values, and also perform competitive in PSNR and SSIM scores. The best values are highlighted in bold.

Dataset	Noise Level	IQA Scores	DnCNN	Ours
Set 14	$\sigma = 15$	PSNR ↑	**31.22**	31.05
		SSIM ↑	**0.9013**	0.9003
		LPIPS ↓	0.0217	**0.0187**
	$\sigma = 25$	PSNR ↑	**28.51**	28.32
		SSIM ↑	**0.8212**	0.8209
		LPIPS ↓	0.1312	**0.0956**
	$\sigma = 50$	PSNR ↑	**24.67**	24.33
		SSIM ↑	**0.7264**	0.7249
		LPIPS ↓	0.1763	**0.1378**

6.5 Conclusion

The modern CNN-based image denoising networks were investigated in this chapter, and a CNN-based image denoising network that combines blind, bias-removed, and mix-loss was proposed and implemented. We train and evaluate our denoising results using PSNR, SSIM, and the perceptual metric LPIPS, and demonstrate that our results achieve impressive performance evaluated with both objective and subjective IQA metrics.

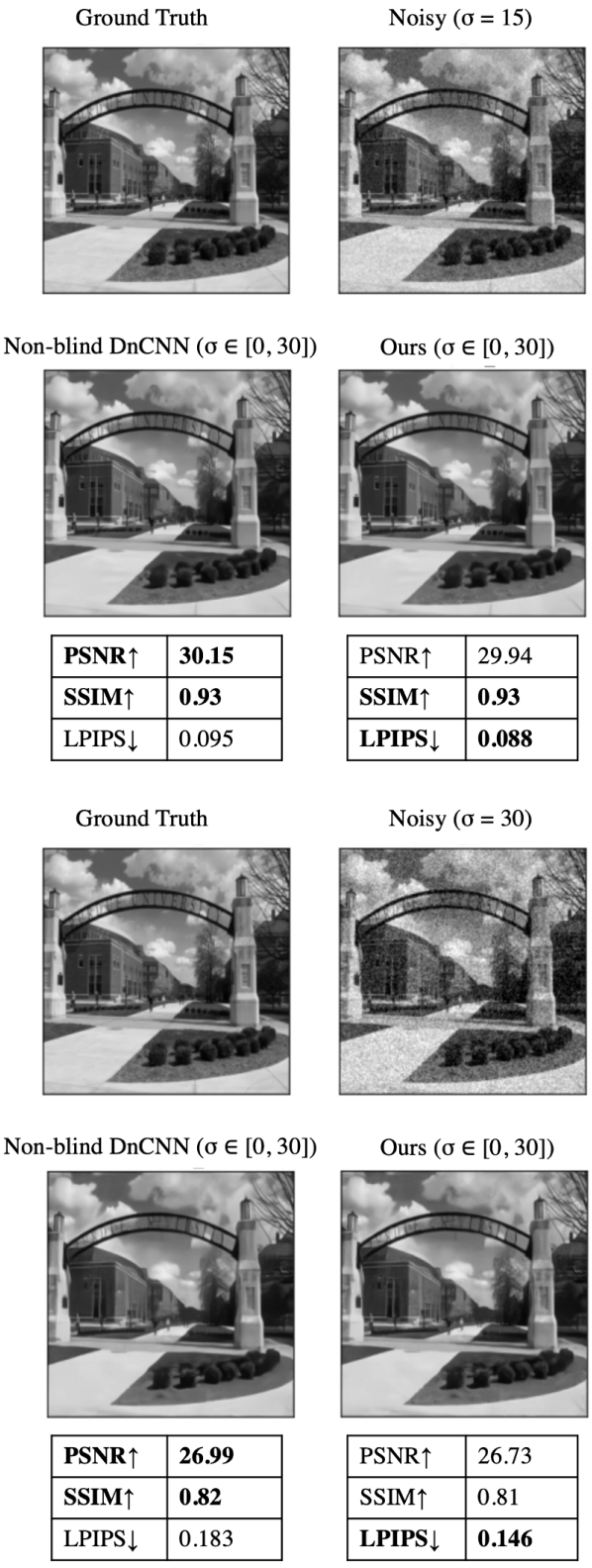


Figure 6.4. Visualized results of denoising for $\sigma = 15$ and 30. The best values are highlighted in bold.

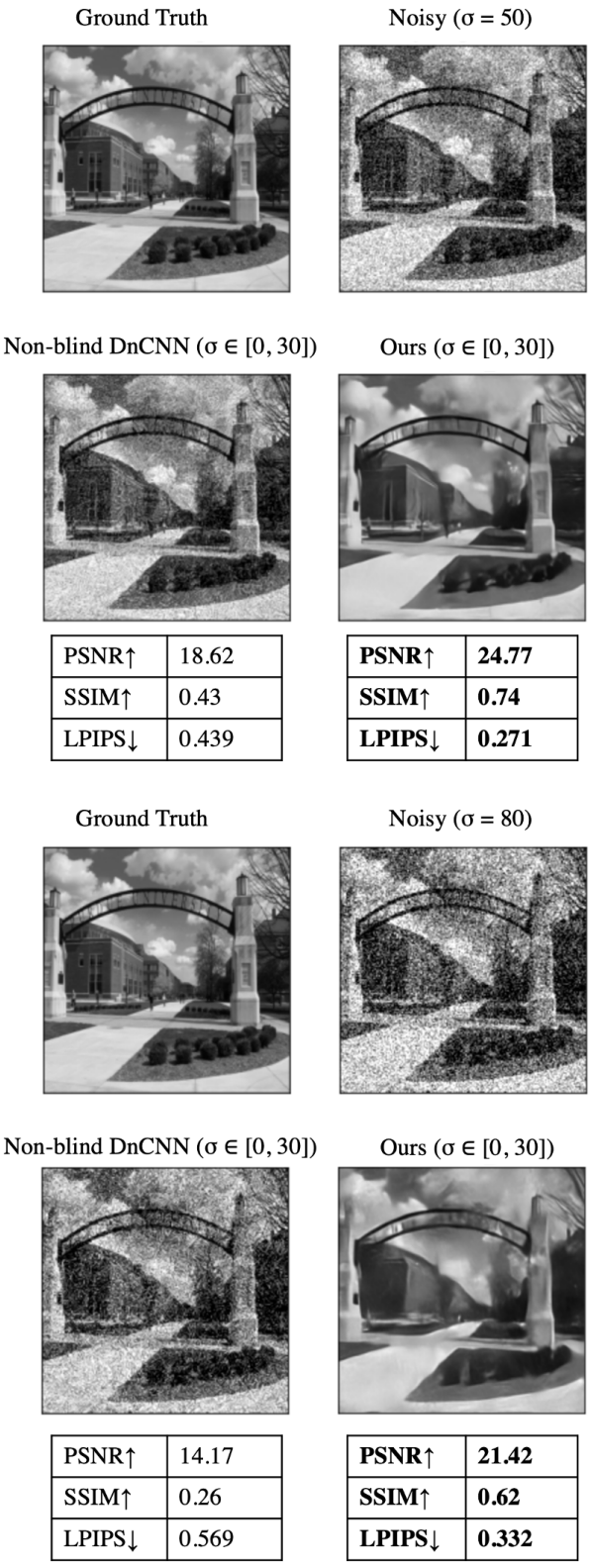


Figure 6.5. Visualized results of denoising for $\sigma = 50$ and 80. The best values are highlighted in bold.

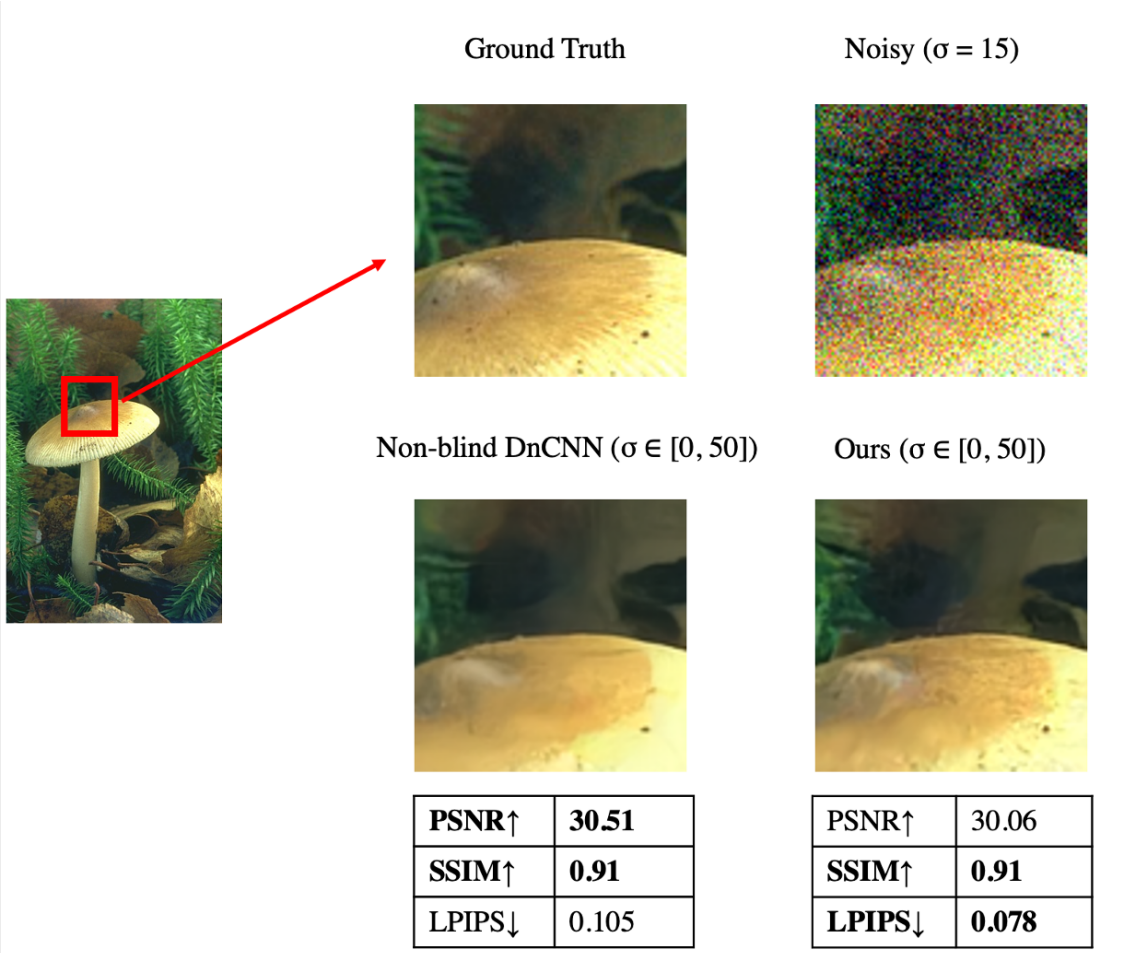


Figure 6.6. Visualized results of denoising for $\sigma = 15$. The best values are highlighted in bold.

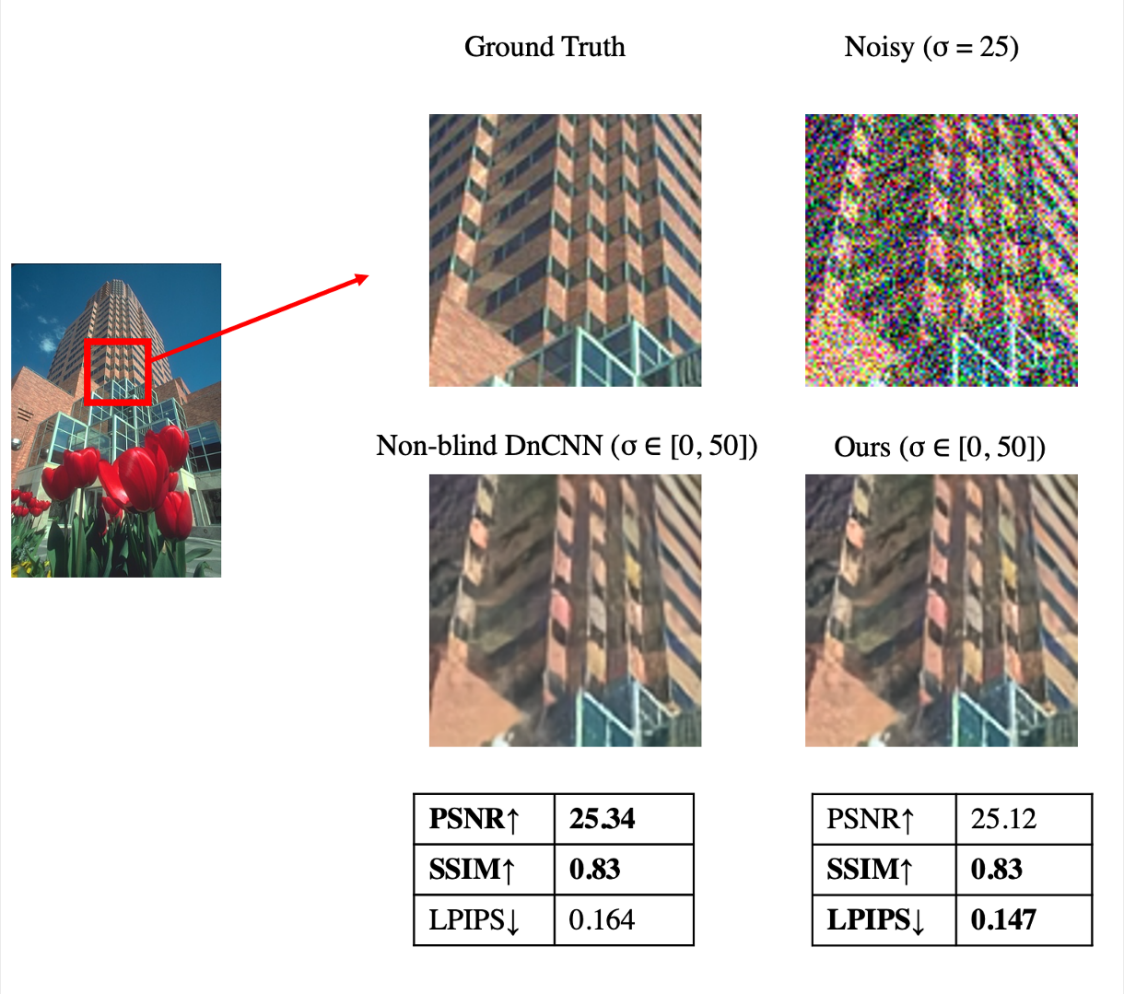


Figure 6.7. Visualized results of denoising for $\sigma = 25$. The best values are highlighted in bold.

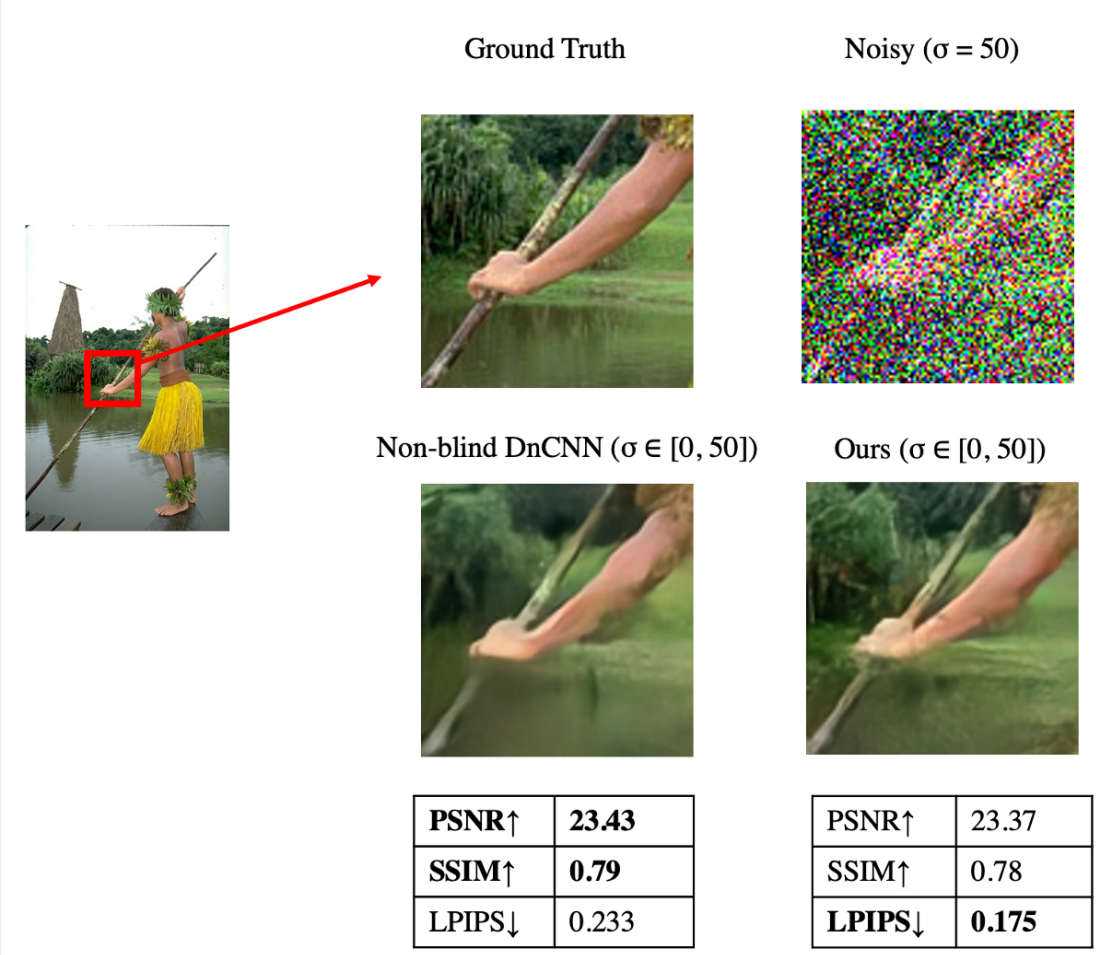


Figure 6.8. Visualized results of denoising for $\sigma = 50$. The best values are highlighted in bold.

7. SUMMARY AND CONTRIBUTIONS

In thesis, we study several problems related to digital image processing and machine learning. Our research is concentrated on the following areas: 1). Color digital halftones. 2). Print image macro-uniformity prediction. 3). Print image defect detection and quality analysis. 4). Image Denoising.

In Chapter 2, we designed and developed a joint FM screen set for halftone color images. We investigated the benefits and drawbacks of the most popular color halftoning algorithms and proposed a method that combines screening and DBS. We designed joint screen sets for Cyan, Magenta, and Yellow colorants, which can be used to halftone color images directly by pixel comparison. We showed our results have better quality than regular screening halftoning, and are simpler to use than the color DBS halftoning.

In Chapter 3, we delved in the macro-uniformity of printed images, one of the most critical factors affecting the overall visual perception of prints. While quantitative analysis of a single artifact is possible, due to the nonlinear nature of the human eye's response, it is difficult to assess the overall perceived quality when multiple artifacts are present simultaneously. We conducted psychophysical experiments to pool perceptual assessments of our printed test images, using the International Council for Information Technology Standards (INCITS) W1.1-designed macro-consistency quality scale as an experimental reference. Then, we computed artifact features that characterize the severity of artifacts in our test sample, and we used these features to train a predictive model. On the basis of human judgment, the predictor can automatically calculate the macro-uniformity score in JND units. Our results showed that the predictor is accurate and that the predicted score corresponds to the subjective visual score (ground-truth). Thus, our predictor can help researchers gain a better understanding of the perceptual macro-uniformity of print quality without conducting subjective experiments.

In Chapters 4 and 5, we detected and measured two important artifacts in printed images: banding and color plane misregistration. These two artifacts are considered to be among the most serious and common defects affecting the overall image quality in the printing

industry. There has been some research on it before, but most has focused on measuring them on uniform color pages or particular test images. We proposed measurement and detection methods that directly operate effectively on customer content pages. Therefore, it is more convenient and can be more widely used than the previous methods. Also, we built artifact classification and quality predictors using machine learning methods.

In Chapter 6, we studied modern deep convolutional neural networks for image denoising. Deep learning-based image denoising has been extensively studied with respect to many properties such as blind, unsupervised, comprehensive and perceptually oriented. Our research focused on the blind, bias-removed, mix loss optimized, and perceptually oriented image denoising task. We proposed a network designed for AWGN image denoising. Our network removed bias at each layer to achieve the benefits of bias-removed networks; additionally, it implemented a mix loss function to boost performance. We trained and evaluated our denoising results using PSNR, SSIM, and the perceptual metric LPIPS, and demonstrated that our results achieve impressive performance on both objective and subjective IQA assessments.

To summarize, the major contributions of the work are listed below:

- Image halftoning
  - We proposed a halftoning method that combines image screening and DBS.
  - We designed joint screen sets for Cyan, Magenta, and Yellow colorants to do color halftoning.
  - Our designed screen set is convenient to store, simple to use, and suitable for generating various color halftone images.
- Image macro-uniformity
  - We conducted human psychophysical experiments to collect the perceptual assessments of image macro-uniformity.
  - We developed two machine learning based predictors for predicting the overall macro-uniformity as judged by humans. We confirmed the efficacy of the predictors using 6-fold cross-validation.

  - We demonstrated that the macro-uniformity image quality ruler method proposed by the INCITS W1.1 macro-uniformity team is a robust, accurate and repeatable method to evaluate perceptual image quality.

- Printed image quality
  - We detected and measured two important artifacts in printed images: banding and color plane misregistration.
  - We developed machine learning based print image quality predictors. We use different models and several evaluation scores to confirm the efficacy of our predictors.

- Image denoising
  - We proposed and built a CNN-based image denoising network that combines blind, bias-removed, and mix-loss.
  - We trained a AWGN blind denoiser and evaluated our denoising results using PSNR, SSIM, and the perceptual metric LPIPS, and demonstrated that our results achieve impressive performance evaluated with both objective and subjective IQA metrics.

Bibliography

[1] Jan P. Allebach, "Selected papers on digital halftoning", *SPIE Milestone Series*, 1999.

[2] Qian Lin and Jan P. Allebach, "Color FM screen design using DBS algorithm", *Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts*, 1998.

[3] Jan P. Allebach and Qian Lin, "FM screen design using DBS algorithm", *IEEE International Conference on Image Processing*, 1996.

[4] Guo-Yau Lin and Jan. P. Allebach, "Generating stochastic dispersed and periodic clustered textures using a composite hybrid screen", *IEEE Transactions on Image Processing*, 2006.

[5] Je H. Lee and Jan P. Allebach, "Colorant-based direct binary search halftoning", *Journal of Electronic Imaging*, 2002.

[6] Sang H. Kim and J. P. Allebach, "Impact of HVS models on model-based halftoning", *IEEE Transactions on Image Processing*, 2002.

[7] Jan P. Allebach, "DBS: retrospective and future directions", *Color Imaging: Device Independent Color, Color Hardcopy, and Graphic Arts*, 2001.

[8] Jan P. Allebach and R. N. Stradling, "Computer-aided design of dither signals for binary display of images", *Applied Optics*, 1979.

[9] Robert W. Floyd and Louis S. Steinberg, "An adaptive algorithm for spatial grayscale", *Proceedings of the Society of Information Display*, 1976.

[10] David J. Lieberman and Jan P. Allebach, "A dual interpretation for direct binary search and its implications for tone reproduction and texture quality", *IEEE Transactions on Image Processing*, 2000.

[11] Peter Morovic, Jan Morovic, Jay Gondek and Robert Ulichney, " Direct pattern control halftoning of neugebauer primaries", *IEEE Transactions on Image Processing*, 2017.

[12] Peter Morovic, Jan Morovic, Jay Gondek and Robert Ulichney, " PARAWACS: color halftoning with a single selector matrix", *Color and Imaging Conference*, 2016.

[13] Thomas J. Flohr, Bernd W. Kolpatzik, Raja Balasubramanian, David A. Carrara, Charles A. Bouman and Jan P. Allebach, "Model-based color image quantization", *IS&T/SPIE International Symposium on Electronic Imaging Science and Technology*, 1993.

[14] Brian A. Wandell, *Foundations Of Vision*, Sinauer Associates, 1995.

[15] Je H. Lee and Jan P. Allebach, "CMYK halftoning algorithm based on direct binary search", *Color and Imaging Conference*, 2001.

[16] Risto Näsänen, "Visibility of halftone dot textures", *IEEE Transactions on Systems, Man, and Cybernetics*, 1984.

[17] Brian Keelan, *Handbook of Image Quality: Characterization and Prediction*, CRC Press, 2002.

[18] Zhou Wang, A.C. Bovik et al, "Image quality assessment: from error visibility to structural similarity", *IEEE Transactions on Image Processing*, 2004.

[19] René D. Rasmussen, Frans Gaykema, Yee S. Ng, Kevin D. Donohue, William C. Kress, and Susan Zoltner, "W1.1 macro uniformity", *Proceedings of SPIE*, 2009.

[20] René D. Rasmussen, William C. Kress, Yee S. Ng, Marguerite Doyle, Kevin D. Donohue, Kate Johnson, and Susan Zoltner, "INCITS W1. 1 macro-uniformity", *Image Quality and System Performance*, 2003.

[21] Brian W. Keelan and Hitoshi Urabe, "ISO 20462: a psychophysical image quality measurement standard", *Image Quality and System Performance*, 2003.

[22] René D. Rasmussen, *W1.1 macro-uniformity software 0.2.0.*

[23] Jim Grice and Jan P. Allebach, "The print quality toolkit: An integrated print quality assessment tool", *Journal of Imaging Science and Technology*, 1999.

[24] Weibao Wang, Gary Overall, Travis Riggs, Rebecca Silveston-Keith, Julie Whitney, George Chiu, and Jan P. Allebach, " Figure of merit for macrouniformity based on image quality ruler evaluation and machine learning framework", *Image Quality and System Performance*, 2013.

[25] Weibao Wang, "A Study on image quality evaluation in image capture and production process", Ph.D. Dissertation, Purdue University, West Lafayette, IN, *ProQuest Dissertations and Theses*, 2016.

[26] James Mannos and David Sakrison, "The effects of a visual fidelity criterion of the encoding of images", *IEEE Transactions on Information Theory*, 1974.

[27] https://www.esat.kuleuven.be/sista/lssvmlab/

[28] https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVR.html

[29] Ron Koha, "A study of cross-validation and bootstrap for accuracy estimation and model selection", *Appears in the International Joint Conference on Artificial Intelligence*, 1995.

[30] Xing Liu, Gary Overall, Travis Riggs, Rebecca Silveston-Keith, Julie Whitney, George Chiu, Jan P. Allebach, "Wavelet-based figure of merit for macrouniformity", *Image Quality and System Performance X*, 2013.

[31] https://www.mathworks.com/help/stats/understanding-support-vector-machine-regression.html.

[32] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2007.

[33] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, 1995.

[34] Robert Chung and Matthew Rees, "A survey of digital and offset print quality issues", *Rochester: Printing Industry Center at RIT*, 2007.

[35] D. René Rasmussen, Edul N. Dalal and Kristen M. Hoffman, "Measurement of macrouniformity: streaks, bands, mottle and chromatic variations", *IS&T PICS Conference Proceedings*, 2001.

[36] Peter D. Burns, Jonathan B. Phillips and Don Williams, "Adapting the ISO 20462 softcopy ruler method for online image quality studies" *IS&T Electronic Imaging*, 2013.

[37] Elena A. Fedorovskaya and Huib De Ridder, "Subjective matters: from image quality to image psychology", *IS&T Electronic Imaging*, 2013.

[38] Yi Yang, Utpal Sarkar, Isabel Borrell and Jan P. Allebach, "Inkjet quality ruler experiments and print uniformity predictor", *IS&T Electronic Imaging*, 2020.

[39] Pei-Ju Chiang, Nitin Khanna, Aravind K. Mikkilineni, Maria V. Ortiz Segovia, Sungjoo Suh, Jan P. Allebach, George T.-C. Chiu and Edward J. Delp, "Printer and scanner forensics," *IEEE Signal Processing Magazine*, 2009.

[40] Ahmed H. Eid; Mohamed N. Ahmed, Brian E. Cooper and Edward E. Rippetoe, "Characterization of electrophotographic print artifacts: Banding, jitter, and ghosting," *IEEE Transactions on Image Processing*, 2011.

[41] Cheng-Lun Chen, George T.-C. Chiu, Jan P. Allebach, "Banding reduction in electrophotographic process using human contrast sensitivity function shaped photoreceptor velocity control", *Journal of Imaging Science and Technology*, 2003.

[42] Jia Zhang, Stephen Astling, Renee Jessome, Eric Maggard , Terry Nelson, Mark Shaw and Jan P. Allebach, "Assessment of presence of isolated periodic and aperiodic bands in laser electrophotographic printer output", *IS&T Electronic Imaging*, 2013.

[43] Jia Zhang and Jan P. Allebach, "Estimation of repetitive interval of periodic bands in laser electrophotographic printer output", *IS&T Electronic Imaging*, 2015.

[44] Jia Zhang, An investigation of print quality defects: psychophysical evaluation of content masking, development of web-based troubleshooting tools, and analysis of sharp roller bands, *Ph.D. Dissertation, Purdue University, West Lafayette, IN*, 2016.

[45] Jie Chen J, Li-hui Zou, Juan Zhang and Li-hua Dou, "The comparison and application of corner detection algorithms", *Journal of Multimedia*, 2009.

[46] Philip HS. Torr and Andrew Zisserman, "MLESAC: A new robust estimator with application to estimating image geometry", *Computer Vision and Image Understanding*, 2000.

[47] Konstantinos G. Derpanis, *Overview of the RANSAC algorithm*, Image Rochester NY, 2010.

[48] scikit-learn, https://scikit-learn.org/stable/

[49] Henry Scheffe, *The Analysis of Variance*, Wiley Classics Library, NY, 1999.

[50] Max Kuhn, and Kjell Johnson, *Feature Engineering and Selection: "A Practical Approach for Predictive Models*, CRC Press, FL, 2019.

[51] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar, *Foundations of Machine Learning*, MIT Press, Cambridge, 2018.

[52] Digna R. Velez, et al, "A balanced accuracy function for epistasis modeling in imbalanced datasets using multifactor dimensionality reduction", *Genetic Epidemiology: the Official Publication of the International Genetic Epidemiology Society*, 2013.

[53] David M.W. Powers, "Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation", *arXiv preprint arXiv:2010.16061*, 2020.

[54] Basak Oztan, Gaurav Sharma, et al., "Misregistration sensitivity in clustered-dot color halftones." *Journal of Electronic Imaging*, 2008.

[55] Jin-Young Kim, Yung-Yao Chen, et al.,"Design of color screen sets for robustness to color plane misregistration", *IEEE International Conference on Image Processing*, 2011.

[56] Runzhe Zhang, Yi Yang, et al., "A comprehensive system for analyzing the presence of print quality defects", *IS&T Electronic Imaging*, 2020.

[57] Alain Hore AND Djemel Ziou. "Image quality metrics: PSNR vs. SSIM", *20th International Conference on Pattern Recognition*, 2010.

[58] James Kasson, Wil Plouffe, et al., "Tetrahedral interpolation technique for color space conversion", *Device-Independent Color Imaging and Imaging Systems Integration*, 1993.

[59] Jae-Chern Yoo, Tae Hee Han, "Fast normalized cross-correlation". *Circuits, Systems and Signal Processing*, 2009.

[60] Manuel Guizar-Sicairos, Samuel Thurman, et al., "Efficient subpixel image registration algorithms'. *Optics Letters*, 2008.

[61] Dana H. Ballard, "Generalizing the Hough transform to detect arbitrary shapes", *Pattern Recognition*, 1981.

[62] scikit-image, https://scikit-image.org/

[63] Goyal B, Dogra A, Agrawal S, et al., "Image denoising review: From classical to state-of-the-art approaches", *Information Fusion*, 2020.

[64] Dabov K, Foi A, Katkovnik V, et al., "Image denoising by sparse 3-D transform-domain collaborative filtering", *IEEE Transactions on Image Processing*, 2007.

[65] Mardia K V, "Multi-dimensional multivariate Gaussian Markov random fields with application to image processing", *Journal of Multivariate Analysis*, 1988.

[66] Zhang K, Zuo W, Chen Y, et al., "Beyond a gaussian denoiser: Residual learning of deep cnn for image denoising", *IEEE Transactions on Image Processing*, 2017.

[67] Lefkimmiatis S, "Non-local color image denoising with convolutional neural networks", *IEEE Conference on Computer Vision and Pattern Recognition*, 2017.

[68] Zhang K, Zuo W, Zhang L, "FFDnet: Toward a fast and flexible solution for cnn-based image denoising", *IEEE Transactions on Image Processing*, 2018.

[69] Guo S, Yan Z, Zhang K, et al., "Toward convolutional blind denoising of real photographs", *IEEE Conference on Computer Vision and Pattern Recognition*, 2019.

[70] Lehtinen J, Munkberg J, Hasselgren J, et al., "Noise2Noise: Learning image restoration without clean data", *International Conference on Machine Learning*, 2018

[71] Mohan S, Kadkhodaie Z, Simoncelli E P, et al., "Robust And Interpretable Blind Image Denoising Via Bias-Free Convolutional Neural Networks", *International Conference on Learning Representations*, 2019.

[72] Zhao H, Gallo O, Frosio I, et al., "Loss functions for image restoration with neural networks", *IEEE Transactions on Computational Imaging*, 2016.

[73] Zhang R, Isola P, Efros A A, et al., "The unreasonable effectiveness of deep features as a perceptual metric", *IEEE Conference on Computer Vision and Pattern Recognition*, 2018.

[74] Ding K, Ma K, Wang S, et al., "Comparison of full-reference image quality models for optimization of image processing systems", *International Journal of Computer Vision*, 2021.

[75] Arbelaez P, Fowlkes C, Martin D, *The Berkeley Segmentation Dataset and Benchmark*, Berkeley University, 2007.

[76] He K, Zhang X, Ren S, et al., "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification", *IEEE International Conference on Computer Vision*, 2015.

[77] Kingma D P, Ba J, "Adam: A method for stochastic optimization", *International Conference on Learning Representations*, 2014.

[78] OpenCV, https://opencv.org/

[79] Rawashdeh, Nathir A, et al.,"Characterization of printer banding in regions of complex image content", *IEEE SoutheastCon*, 2007.

VITA

Yi Yang received her B.S in Geomatics Engineering from Wuhan University and M.S in Geomatics Engineering from Chinese Academy of Sciences. She is a Ph.D. candidate in Electrical and Computer Engineering at Purdue University in West Lafayette. Her primary area of research has been digital image processing, computer vision and machine learning.