Precision Improvement of Overprinting System Based on Improved Laplace Edge Detection Algorithm

Yingbo Wang(✉), Likun Lu, Qingtao Zeng, Rui Zhao, Yang Zhang, and Fucheng You

Beijing Institute of Graphic Communication, Beijing, People's Republic of China

Abstract. This paper proposes a solution to locate the crossline center in overprinting system. Crosslines of 4 different colors are printed at the same coordinate. Because of mechanical error, they do not coincide. The system uses the measured deviation to calibrate. This paper obtains accurate deviation value through three steps. First, the picture collected by a CCD camera will be processed by color segregation algorithm. In this process, RGB data turn into CMYK data, and each color will be on a single picture. After that, a Laplace edge detection algorithm is improved by combining it with 2D Gauss filter. This improved Laplace edge detection algorithm has a better noise suppression effect, which means it is even less likely to judge noise as the edge of a graph. Finally, target searching algorithm based on rhombus matching is used to figure out the center of crossline. The average absolute error from 2,000 simulations is 3.192 pixel, which shows that the algorithm in this paper has a high accuracy.

Keywords: Laplace edge detection · Overprint · Color segment · Crossline

1 Introduction

Overprinting is an important method of color printing [1]. The basic principle of overprinting is that several different colors of ink are printed to the same paper in sequence [2]. When all the different colors are painted, all the print heads need the same frame of reference. It requires a high location precision, or the final prints will be misplaced.

Today, overprinting system is running at a relatively higher speed, which leads to a lower printing precision. To make sure the printing precision meets the requirements, we must limit the printing speed. At the same time, how to evaluate the quality of the prints effectively is also an important task. If we do not find out and solve the problems of printing precision in time, a waste of time and money will not be avoided resulting from defective prints, especially in industrial production.

The paper aims at solving the above problems. First the color picture is divided into four color channels [3–5]: C, M, Y and K. After that we use improved Laplace edge detection algorithm to locate the center of the crossline. Then the deviations of channel K to channel C, M and Y are calculated. And the deviation is the overprinting error. This paper makes the overprinting error more accurate. As a result, the whole overprinting system can work more accurately. And the printing quality can also be improved.

Y. Weng et al. (Eds.): TridentCom 2020, LNICST 380, pp. 17–26, 2021.
https://doi.org/10.1007/978-3-030-77428-8_2

2 Traditional Crossline Detection Methods

Crossline detection is important in many industries. To make every color print head work accurately, we often need to print a crossline of each color and locate the crossline center by a photo token by a high-speed CCD camera [6], to realize the calibration of overprinting system. If we can get a clear photo of the crossline, the method is pretty good.

However, the photos usually are of a low signal-to-noise ratio and contrast. There are often some limitations in traditional crossline detection methods, such as Hough transform [7], template matching [8], morphological corrosion [9], linear fitting and so on.

The result using linear fitting will be of a large deviation if the edge of a crossline image is irregular [10, 11]. And the result using morphological corrosion will not find the crossline center precisely if the contrast of the photo is not high enough.

3 Error Detection Based on Overlay Quasi-identifier

3.1 Overprinting Quality Detection Method

Traditionally, the overprinting error detection process is realized by examining the overlay quasi-identifier artificially. Figure 1 shows three kinds of different overlay quasi-identifiers.

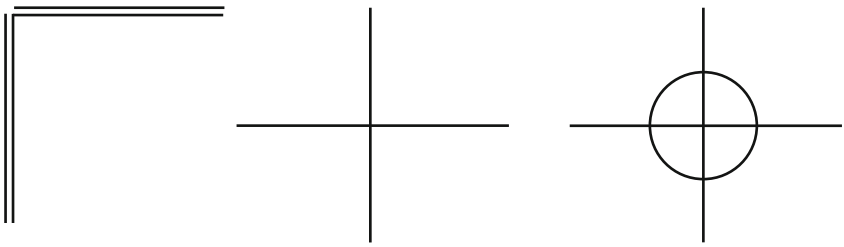

Fig. 1. Common overprint overlay quasi-identifiers

If the deviation is larger than the line width, usually 1 mm, we support that the overprinting system is not working well. Also, the prints are off specification.

The crossline detection processing framework is shown as Fig. 2. First, a picture of the print is collected by an imaging sensor, for example, a CCD camera. And then four crosslines of C, M, Y and K purity colors can be processed by image segmentation.

Fig. 2. Crossline center locating processing framework

After that we locate the center of each crossline by image edge detection algorithm. We can judge whether the overprinting system is working exactly or not.

If the overprinting system is not working well, a calibration can be implemented.

3.2 Color Segmentation Method

A significant step of the detection using computer vision is how to get the data of channel C, M, Y and K. According to the relation of RGB and CMYK, there is not a one-to-one correspondence. If we transform RGB to CMYK, there must be a replacement value for the black color. The replacement value can be figured out by C, M and Y channel. In fact, RGB color range is 0–255, and CMYK color range is 0–100. So, we can get the transform relation as:

$$\begin{bmatrix} c \\ m \\ y \end{bmatrix} = \begin{bmatrix} G_{\max} \\ G_{\max} \\ G_{\max} \end{bmatrix} - \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{1}$$

$$K = \min(c, m, y) \tag{2}$$

$$C = c - K \tag{3}$$

$$M = m - K \tag{4}$$

$$Y = y - K \tag{5}$$

In the expressions, $G_{\max}$ is the most value of a color, and usually it is 255.

In actual situations, four different color crosslines may be in several kinds of states. The four crosslines can be separated from each other. And it can be that two crosslines are overlapped and another two is separated. Or three crosslines are overlapping. All in all, the overlapping relation is complicated.

Some scholars propose an approach to solve the problem. First, we need to judge which overlapping relation the crosslines are in. And for a certain relation, we use a different threshold to divide the picture. As we learned, there are 15 different color overlap, so we need 14 threshold values. This method requires a high color precision and how to judge which color relation the crosslines belonging to is difficult.

To solve this problem, we a method mentioned in reference. Figure 3 shows the flow of this method.

4 Improved Laplacian Edge Detection Algorithm

In the previous section, we discussed how to make overlapping crosslines separated from each other. In this section, we will solve the problem of how to find the crossline center by an improved Laplacian edge detection algorithm.

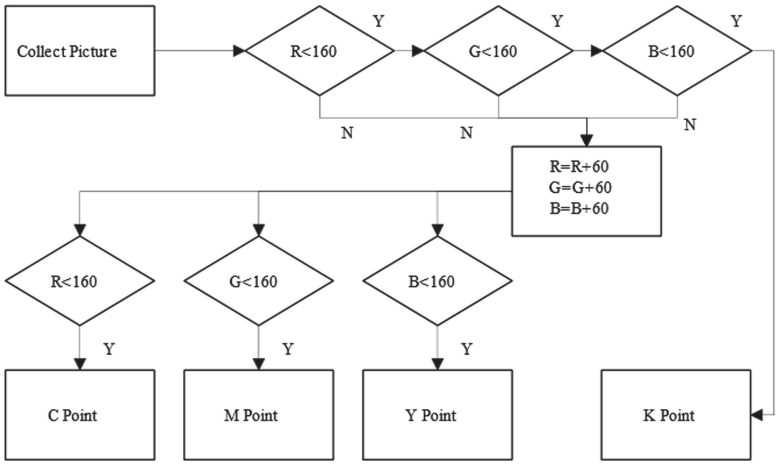

Fig. 3. color segmentation method flow chart

4.1 The Disadvantage of Laplacian Operator

The 2D-Laplace transform is based on 2D-gradient. And the definition is as follow.

$$\begin{aligned}\nabla^2 f(x,y) &= \frac{\partial^2 f(x,y)}{\partial^2 x} + \frac{\partial^2 f(x,y)}{\partial^2 x} \\ &\approx \frac{\partial\left[f(x+1,y) - f(x,y)\right]}{\partial x} + \frac{\partial\left[f(x,y+1) - f(x,y)\right]}{\partial y} \\ &\approx f(x+1,y) - f(x,y) - \left[f(x,y) - f(x-1,y)\right] \\ &+ f(x,y+1) - f(x,y) - \left[f(x,y) - f(x,y-1)\right] \\ &\approx f(x+1,y) + f(x-1,y) + f(x,y+1) + f(x,y-1) - 4f(x,y) \end{aligned} \tag{6}$$

2D-Laplace transform is usually expressed by a convolution of a matrix and a Laplace core. And we can get the following equations.

$$I = \begin{pmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{pmatrix} \tag{7}$$

$$I_- = \begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix} \tag{8}$$

And the 2D-Laplace transform can be described as follow.

$$\begin{cases} \nabla^2 f(x,y) = P * I \\ \nabla^2 f(x,y) = P * I_- \end{cases} \tag{9}$$

In the above equations, * means convolution, and I is Laplace core.

These equations show that if we want to figure out the convolution result of picture matrix P and Laplace core I or I_{-}, actually we will figure out the difference value between the value of a certain pixel and the neighboring horizontal and vertical pixels. The convolution result is four times this difference.

Traditional Laplace edge detection uses the location of zero point or the convolution result to find out the edge of figures. This method may take noise point as an edge with high probability. It is because that Laplace operator is not of the feature of smoothing the picture, which is available for Sobel operator or Prewitt operator.

4.2 Improvement of Laplace Operator Using 2D-Gaussian Function

In this section we combine traditional Laplace edge detection with Gaussian function. 2D-Gaussian function is defined as:

$$\text{Gauss}(x, y, \sigma) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{x^2 + y^2}{2\sigma^2}\right) \tag{10}$$

Improved Laplacian edge detection based on Gaussian smoothing filter is as follow:

Step 1: Figure out Gaussian matrix $G_{H\times W}$ according to Eq. (10).

$$G_{H\times W} = \left[\text{Gauss}(x, y, \sigma)\right]_{0\le x\le H-1, 0\le y\le W-1, x, y\in N} \tag{11}$$

Step 2: Figure out the sum of Gaussian matrix.

$$sum = \sum G_{H\times W} \tag{12}$$

Step 3: Normalize. It means that the Gaussian matrix is divide by the sum. Then we get the Gaussian convolution operator K.

$$K = \frac{G_{H\times W}}{sum} \tag{13}$$

Step 4: The image matrix P is convolved with the Gaussian convolution operator K. After the Gaussian smoothing filter, the result will be convolved with Laplace core I or I_{-}.

$$\begin{cases} R_{\text{conv}} = P * K * I \\ R_{\text{conv}} = P * K * I_{-} \end{cases} \tag{14}$$

4.3 Algorithm Optimization

The time complexity of the algorithm in Sect. 3.2 is high, because it requires 2D convolution. We can optimize the algorithm by turning the 2D convolution into 1D convolutions. The process is as follows.

Step 1: Take the Laplace transform of 2D-Gaussian function.

$$\begin{aligned}\nabla^2\big[\mathrm{Gauss}(x,y,\sigma)\big] &= \frac{\nabla^2\big[\mathrm{Gauss}(x,y,\sigma)\big]}{\partial^2 x} + \frac{\nabla^2\big[\mathrm{Gauss}(x,y,\sigma)\big]}{\partial^2 y} \\ &= \frac{1}{2\pi\sigma^2}\left[\begin{array}{l}\dfrac{\partial\left(-\frac{x}{\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right)}{\partial x} \\ +\dfrac{\partial\left(-\frac{y}{\sigma^2}\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right)\right)}{\partial y}\end{array}\right] \\ &= \frac{1}{2\pi\sigma^4}\left(\frac{x^2}{\sigma^2}-1\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \\ &+ \frac{1}{2\pi\sigma^4}\left(\frac{y^2}{\sigma^2}-1\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \\ &= \frac{1}{2\pi\sigma^4}\left(\frac{x^2+y^2}{2\sigma^2}-2\right)\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \end{aligned} \tag{15}$$

Step 2: Construct an operator K^I, whose window size is $H \times W$, and the standard deviation is σ

$$\begin{aligned}K^I = \nabla^2\left(\mathrm{Gauss}\left(x-\frac{H-1}{2}, y-\frac{W-1}{2}, \sigma\right)\right) \\ 0 \le x \le H, 0 \le y \le W\end{aligned} \tag{16}$$

Step 3: Convolve picture matrix P and improved Laplace operator K^I,then we can get $R_{\mathrm{conv}} = P * K^I$. Use R_{conv} to resolve Eq. (15).

$$\begin{aligned}\nabla^2(\mathrm{Gauss}(x,y,\sigma)) &= \frac{1}{2\pi\sigma^4}\left(\frac{x^2+y^2}{2\sigma^2}-2\right)\cdot\exp\left(-\frac{x^2+y^2}{2\sigma^2}\right) \\ &= \left(\frac{x^2+y^2}{\sigma^2}-2\right)\cdot\frac{\mathrm{Gauss}(x,\sigma)\cdot\mathrm{Gauss}(y,\sigma)}{\sigma^2}\end{aligned} \tag{17}$$

$$\mathrm{Gauss}(x,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{x^2}{2\sigma^2}\right) \tag{18}$$

$$\mathrm{Gauss}(y,\sigma) = \frac{1}{\sigma\sqrt{2\pi}}\exp\left(-\frac{y^2}{2\sigma^2}\right) \tag{19}$$

Step 4: Binarization process. And the edge detection result can be worked out.

$$edge(x,y) = \begin{cases}255, & R_{\mathrm{conv}} > 0 \\ 0, & R_{\mathrm{conv}} \le 0\end{cases} \tag{20}$$

Or:

$$edge(x, y) = \begin{cases} 255, & R_{\text{conv}} < 0 \\ 0, & R_{\text{conv}} \geq 0 \end{cases} \tag{21}$$

In step 3, the 2D-covnlution is turned into 1D-convlution, which leads to a lower time complexity. In step 4, not like Prewitt operator or Sobel operator, we do not take the absolute value of R_{conv}. Instead, Eq. (20) or (21) is used to judge the probability of if a pixel is the edge or not.

5 Determine the Center Coordinates of the Crosslines

In Sects. 3.1, 3.2 and 3.3, we get the edge of the crosslines. In this section the center coordinates will be determined by a target searching algorithm.

5.1 Target Searching Algorithm Based on Rhombus Matching

First, we define $S_{i,j}$ as the pixel value of each color crosslines. $S_{i,j} = 1$ when a pixel belongs to a certain color (C, M, Y or K), or $S_{i,j} = 0$. Figure 4 shows the rhombus matching template.

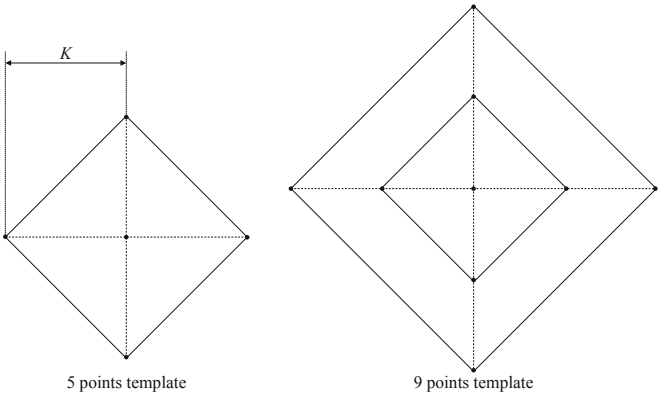

Fig. 4. rhombus matching template

In Fig. 4, K is related to the size of crosslines. It is the horizontal or vertical distance between neighbor points. And in the following searching process, step-size is L. Take $K = 9$ for example.

Step 1: If a pixel point (m, n) is on the crossline, which means that it satisfies $S_{m,n} = 1$, then 3×3 template is used. And go to step 2. If not, go to step 4.

Step 2: If there are 5 points on the crossline, we use 5×5 template. And go to step 3. If not, go to step 4.

Step 3: If there 9 points on the crossline, this point (m, n) is the center of the crossline. If not, go to step 4.

Step 4: Update the pixel point with the following equations. Do step 1–4 again, until it meets the reequipments.

$$(m, n)' = (m + L, n) \tag{22}$$

$$(m, n)' = (m, n + L) \tag{23}$$

5.2 Additional Cases for Practical Application

The line width is usually more than one pixel. As a result, the center point may be more than one, for a certain color. Then we just need to figure out the average of these center points as the center of crosslines.

Define x and y as the center point (x, y), we can get:

$$x = \frac{1}{N} \sum_{i=1}^{N} m_i \tag{24}$$

$$y = \frac{1}{N} \sum_{i=1}^{N} n_i \tag{25}$$

In the equations, N is the total number of center points before averaging.

6 Precision Improvement of Overprinting System

Based on the center point of crosslines of different colors, we can figure out the error of the overprinting system. If the location of the four centers is at the same point, the printing system is working well. If it is not the same, we need to calibrate the reference frame of the nozzles. In fact, this process is performed by computer.

7 Simulation and Experimental Results Analysis

In this paper, we use MATLAB to process image data and all the algorithm is implemented. In this section, the results are showed out.

We use MATLAB to process the collected images. For example, Fig. 5(a) shows an actual picture printed by a certain model of overprinting system.

Figure 5(b)–(e) shows C, M, Y and K grayscale images after the color segmentation. From the images we can see that the main feature of each different color channel is extracted reasonably.

After the color segmentation, we judge the center coordinate of every crossline. We use the simulation program to randomly generate some crosslines with known central coordinates. The method in this paper is used to calculate the central coordinates of the

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

Fig. 5. Color segmentation result

crosslines. And then we calculate the absolute error between the real value and the value calculated by the algorithm.

$$\varepsilon = \sqrt{(x - x_0)^2 + (y - y_0)^2} \tag{26}$$

In Eq. (26), x and y mean the value calculated by the algorithm. x_0 and y_0 are the real values. Table 1 shows some results of the simulation experiment. The unit in the table is pixel. All pictures are 256 * 256.

Table 1. Simulation results of improved Laplace algorithm

No.	x	y	x_0	y_0	ε
1	132	106	133.71	102.71	3.706
2	120	144	120.00	143.71	0.286
3	149	113	151.57	109.86	4.061
4	160	111	163.57	111.00	3.571
5	85	141	83.14	140.29	1.990
6	166	122	170.14	126.29	5.961
7	163	154	165.57	156.71	3.739
8	87	132	83.14	132.86	3.951
9	167	125	171.29	125.43	4.307
10	165	133	163.29	137.14	4.484

The average absolute error from 2,000 simulations is 3.192 pixel. This shows that the algorithm in this paper has a high positioning accuracy.

8 Summary and Prospect

This paper improved the edge detection algorithm in overprinting calibration. According to the results in experiment, the algorithm in this paper is of a higher precision. Using this method can improve the quality of color printing.

However, compared with traditional methods, this algorithm requires more matrix operations. This is not conducive to real-time monitoring at printing time. One possible

solution is to make use of GPU or FPGA to speed up the process. And this is a further study issue.

Acknowledgements. This publication is based on work supported in part by major special project of science and technology of Guangdong Province, No. 190826175545233 and Beijing science and technology innovation service capability construction project, No. PXM2016_014223_000025, and BIGC Project (Ec202007). Any opinions, findings and conclusions or recommendations expressed in this publication are those of the authors and do not necessarily reflect the views of the funding agency.

References

1. Jia, W.: A novel image registration control system based on improved composite metric entropy function in color printing. Multimed. Tools Appl. **79**, 9225–9236 (2020)
2. Seipel, S., Yu, J., Viková, M., et al.: Color performance, durability and handle of inkjet-printed and UV-cured photochromic textiles for multi-colored applications. Fibers Polym. **20**, 1424–1435 (2019)
3. Chen, X., Wang, Q., Lee, Y.: Real-time demosaicking method based on mixed color channel correlation. J. Real-Time Image Proc. **16**, 61–69 (2019)
4. Darwish, S.M., Al-Khafaji, L.D.S.: Dual watermarking for color images: a new image copyright protection model based on the fusion of successive and segmented watermarking. Multimed. Tools Appl. **79**, 6503–6530 (2020)
5. Hinojosa, S., Oliva, D., Cuevas, E., et al.: Reducing overlapped pixels: a multi-objective color thresholding approach. Soft. Comput. **24**, 6787–6807 (2020)
6. Baldi, A.: Digital image correlation and color cameras. Exp. Mech. **58**, 315–333 (2018)
7. Yang, C., Collins, J.: Improvement of honey bee tracking on 2D video with hough transform and Kalman filter. J. Sig. Process. Syst. **90**, 1639–1650 (2018)
8. Chen, F., Ye, X., Yin, S., et al.: Automated vision positioning system for dicing semiconductor chips using improved template matching method. Int. J. Adv. Manuf. Technol. **100**, 2669–2678 (2019)
9. Cao, X., Gao, S., Chen, L., et al.: Ship recognition method combined with image segmentation and deep learning feature extraction in video surveillance. Multimed. Tools Appl. **79**, 9177–9192 (2020)
10. Bi, Q., Huang, J., Lu, Y., et al.: A general, fast and robust B-spline fitting scheme for micro-line tool path under chord error constraint. Sci. China Technol. Sci. **62**, 321–332 (2019)
11. Maity, R.: Regression analysis and curve fitting. Statistical Methods in Hydrology and Hydroclimatology. STCEE, pp. 229–257. Springer, Singapore (2018). https://doi.org/10.1007/978-981-10-8779-0_7