https://doi.org/10.2352/ISSN.2470-1173.2020.15.COLOR-352
© 2020, Society for Imaging Science and Technology

Developing an inkjet printer III: Multibit CMY halftones to hardware-ready bits*

***Sige Hu*[1], *Daulet Kenzhebalin*[1], *Bakedu Choi*[1], *George Chiu*[1], *Zillion Lin*[2], *Davi He*[2], *Jan Allebach*[1]**
[1] ***Purdue University, West Lafayette, IN 47906, U.S.A***
[2] ***Sunvalley Tek, Shenzhen, China***

Abstract

Nowadays, inkjet printers are widely used all around the world. But how do they transfer the digital image to a map that can control nozzle firing? In this paper, we briefly illustrate that part of the printing pipeline that starts from a halftone image and end with Hardware Ready Bits (HRBs). We also describe the implementation of the multi-pass printing method with a designed print mask. HRBs are used to read an input halftone CMY image and output a binary map of each color to decide whether or not to eject the corresponding color drop at each pixel position. In general, for an inkjet printer, each row of the image corresponds to one specific nozzle in each swath so that each swath will be the height of the printhead [1]. To avoid visible white streaks due to clogged or burned out color nozzles, the method called multi-pass printing is implemented. Subsequently, the print mask is introduced so that we can decide during which pass each pixel should be printed.

Introduction

This paper mainly consists of two sections. The first section is dedicated to introduce HRBs. We use three subsections to demonstrate how HRBs are related to inkjet printing. First, we introduce our color ink cartridge since different number of colors, different color arrangements and different numbers of nozzles for each color could all affect our generation of HRBs. Then we present how our inkjet driver communicates with our cartridge nozzles. Different nozzles are controlled by different combinations of power lines and address lines. Hence, for each swath, we provide a PA sequence for each column that continues to the next column while the cartridge is moving horizontally. In this subsection, we explain how to fire a drop once we know the information of which nozzle needs to eject a drop. But how can we obtain this information from the input halftone image? The last subsection introduces the term printer-addressable pixel or said sequence map to represent which nozzle should eject a drop at each swath [2].

The second section mainly focuses on improving print quality. In this section, we introduce multi-pass printing, which is used to reduce the effect of the missing nozzles that could cause visible white stripes [1]. At this point, we introduce the concept of the print mask which is required by multi-pass printing to decide on which pass each pixel needs to be fired [3]. In this paper, we use the direct binary search algorithm to generate the print mask [4, 5]. Finally, we deploy multi-drop printing to enhance the color saturation of our prints [6].

*Research supported by Sunvalley Tek, Shenzhen, CHINA.

Hardware Ready Bits (HRBs)

Color ink cartridge

For our specific nail-printer, we use the Lexmark † color ink cartridge L26, which is the tricolor cartridge shown in Fig. 1. The red circle shown in the Fig. 1 cartridge bottom view indicates the nozzle location. If we magnify that region by a electron microscope, we can observe the nozzle distribution of our cartridge L26.

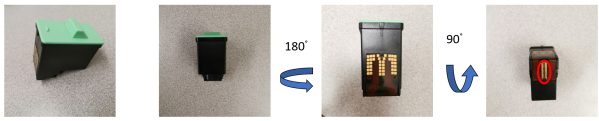


Figure 1. *Four different views of the color ink cartridge L26*

As Fig. 2 shows [7], this cartridge has three colors cyan, magenta, and yellow. Each color has 64 nozzles which consists of 32 even nozzles and 32 odd nozzles that correspond to the even and odd rows for our prints. The even and odd nozzles are arranged in two adjacent columns, as shown in the insert in Fig. 2. Within each column, the vertical separation between nozzles is 1/300 in. By offsetting the two columns relative to each other in the vertical direction by 1/600 in, we effectively achieve 600 dpi (dots per inch) resolution in the vertical direction. By arranging the nozzles in two separate columns, we increase the physical strength of the nozzle plate, since adjacent nozzle holes are further apart. L26 has 192 nozzles in total from H1 to H192, where H1 to H64 are cyan nozzles, H65 to H128 are magenta nozzles, and H129-192 are yellow nozzles. Given the information in Fig. 2, the separation between each color is $1/20$ in. By a simple calculation $\frac{1/20\,in}{1/600\,in}$, we figure out that the distance from H64 to H65 or H128 to H129 is 30 times the consecutive even and odd nozzle distance, which we consider to be the unit pixel height $(1/600\,in)$.

To mitigate the impact of the missing nozzles either being clogged or burned out, for each colorant of our cartridge, we decide to only use 48 nozzles by ignoring the top and bottom eight nozzles. For instance, for the cyan nozzles, we only utilize nozzles H9 to H56 so that malfunctioning of H1 to H8 or H57 to H64 will not affect our prints. However, referring to the abstract, the swath height, which is directly determined by the utilized nozzles is decreased because of this restriction [1]. It will result in a larger number of swaths to print an area of a fixed vertical height, which

† Lexmark International, Inc., Lexmark, KY.

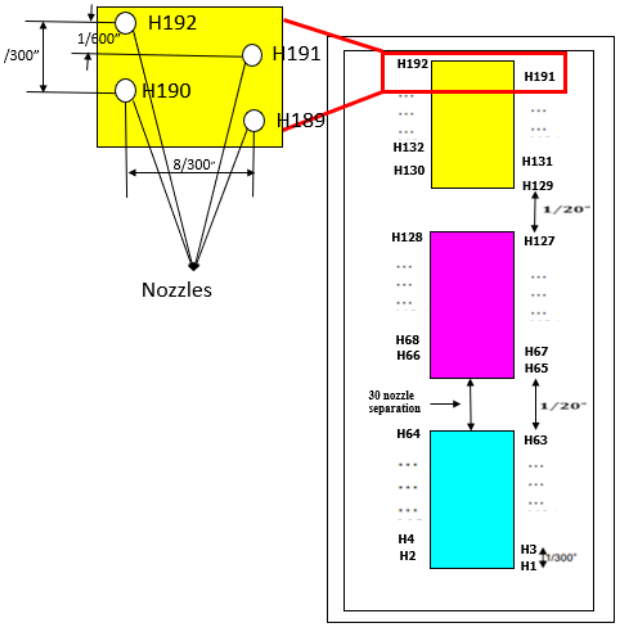


Figure 2. *Color ink cartridge L26 nozzles layout.*

also means spending more time for printing, but which results in better quality prints.

Nozzle and inkjet driver communication

Having introducing our color ink cartridge, the reader might now be curious to know how does our printer control these nozzles to eject ink drops. Looking back to the Fig. 1 cartridge back view, we see that there is a goat horn shaped golden pad, which also exists in the printer cartridge holder as shown in Fig. 3. Hence, this goat horn shaped pad is the channel for the cartridge and inkjet driver to communicate and transfer data.

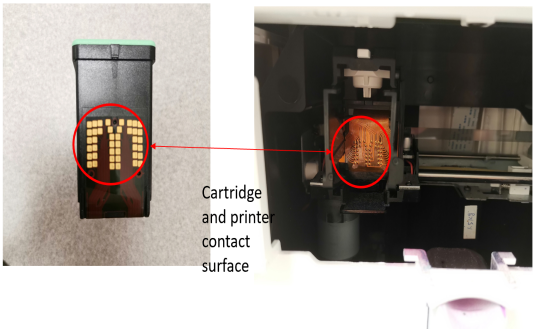


Figure 3. *Goat horn shaped golden pad on cartridge back (left) and in cartridge holder (right).*

But what information can this golden pad provide and how is it used to determine exactly which nozzle will be fired? This leads us to two new terms that are called the *power line* and the *address line* [7]. We have thirteen address lines from A1 to A13 which are sequentially related in a circuit so that only one address line is allowed to be set to high at a time. However, the sixteen power lines, which are designated as P1-P16, are distributed in parallel, so that we are allowed to set several power lines to high at the same time. Fig. 4 shows how the power line and address line correspond to the pins located in the goat horn shaped pad. And Fig. 5 specifies how the power lines and address lines are used to control each nozzle.

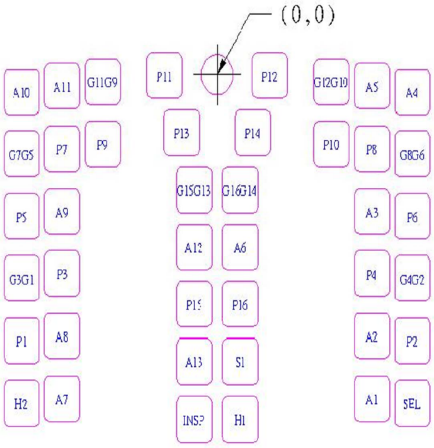


Figure 4. *Power lines and Address lines corresponding to each pin location.*

The first row of Fig. 5 [7], which is shown as light yellow, identifies the power lines P1 to P16. Note that this table separates even and odd power lines because we fire drops of even and odd nozzles in two different phases of each pass. Specifically, odd-numbered nozzles are fired as the printer cartridge passes over the nail from left to right (Phase 1); and even-numbered nozzles are fired as the printer cartridge passes over the nail from right to left (Phase 2). This is known as *bidirectional printing*. Between the two phases for each pass, the cartridge is not advanced in the vertical direction. After the completion of Phase 2 of each pass, we are ready to print Phase 1 of the next pass; and the cartridge moves down by a fraction of its height according to how many passes are being used for the print mode. Multi-pass printing is discussed in further detail later in this paper. The reason to print odd-numbered nozzles and even-numbered nozzles in two different phases is to allow more time between the printing of pixels that are vertically adjacent to each other. This reduces drop coalescence, which would decrease print quality.

The even-numbered power lines are used to control all the even-numbered nozzles and the odd-numbered power lines are used to control all the odd-numbered nozzles. The first column of Fig. 5, which is shown as dark green, identifies the address lines A1 to A13. Each power line and each address line is combined to control one specific nozzle. For instance, the odd cyan nozzle H19 is controlled by P1 and A2. According to the properties of the power lines and address lines mentioned before, we can simultaneously fire all nozzles that are controlled by A1 with different power lines, such as the cyan nozzles H25, H51, H18, H44, the magenta nozzles H77, H103, H70, H96, H122, and the yellow nozzles H129, H155, H181, H148, H174. But due to our bi-directional 2-phase printing mode, we will not fire even-numbered

and odd-numbered nozzles at the same time. Also, because of this property, we cannot simultaneously fire all nozzles that are controlled by P1 with different address line combinations, such as H19, H13, and H7.

To store the PA sequence of this table, we use a thirteen element one dimension array $pa_{out}[\cdot]$. Each element of the array represents a combination of one address line and all the power lines. For example, $pa_{out}[1]$ is used to represent the combination of address line A1 with the power lines P1 to P16. But how can we store the information of sixteen power lines in one element? We use the shift operator " << " which bit-wise shifts data to the left. To be more specific, given data "0010", "0010" << 1 results in "0100", where all the digits in the original data are shifted to the left by one bit. Similarly, "0010" << 2 results in "1000" where all the digit in the original data are shifted to the left by two bits. If we want to fire a given nozzle, the nozzle information will be "1"; otherwise it is "0". Hence, to fire a drop for nozzle H25 and not fire a drop for H45, $H[25] =$ "1" and $H[45] =$ "0". For different power lines, we bit-wise shift them differently, for power line P1, we bit-wise shift the given data by zero bits (using $<< 0$) ; for power line P2, we bit-wise shift the data by one bit (using $<< 1$); shift two bits for P3 (using $<< 2$); shift three bits for P4 (using $<< 3$); and so on.

With this basic introduction, we can now take a look at the table shown in Fig. 5. We use address line A1 which corresponds to the first row of the table as an example. With our bi-directional print mode, we print with the even-numbered and odd-numbered nozzles in two separate phases thus resulting in two different PA sequences for even-numbered and odd-numbered nozzles, respectively. If we want to fire drops for all odd-numbered nozzles controlled by address line A1: [H25], [H51], [H77], H[103], H[129], H[155], and H[181] are all "1". Then, $pa_{out}[1] = [H25] << 2 + [H51] << 4 + [H77] << 6 + [H103] << 8 + [H129] << 10 + [H155] << 12 + [H181] << 14 =$ "100"+"10000"+"1000000"+"100000000"+ "10000000000" + "1000000000000" + "100000000000000" = "101010101010100". So we can see that the first bit is 0; and all the other odd-numbered bits such as third, fifth, and seventh bits are 1 which means only power line P1 is set to low, while the other odd-numbered power lines are high. Similarly, if we want to fire all even-numbered nozzles controlled by address line A1: [H18], [H44], [H70], H[96], H[122], H[148], and H[174] are all set to "1". $pa_{out}[1] = [H18] << 1 + [H44] << 3 + [H70] << 5 + [H96] << 7 + [H122] << 9 + [H148] << 11 + [H174] << 13 =$ "10" + "1000" + "100000" + "10000000" + "1000000000" + "100000000000" + "10000000000000" = "10101010101010". So all even-numbered bits are "1", except bit 16.

Input halftone image and output printer-addressable pixels

The array of printer-addressable pixels is defined by a binary sequence map used to tell at a certain pixel location whether the corresponding nozzle fires an ink drop. For a three-tuple input halftone image, the sequence map is also a three tuple that we denote as the C-sequence, M-sequence, and Y-sequence. The width of the sequence map is always equal to the width of input halftone image, which is 600 pixels for our printer, while the height is given by Eq. 1. But how can we get the number of swaths mentioned in Eq. 1? For single-pass printing, the number of swaths is given by Eq. 2. The expression is slightly different for multipass printing which will be discussed later. The derivation of Eq. 2 will also be provided when we discuss multipass printing. Therefore, for our printer, with image height 900 pixels, the number of swaths for one pass is 17 ($ceil((900+94+94)/64) = 17$).

$$\text{sequence map height} = 64 \times \text{number of swaths} \quad (1)$$

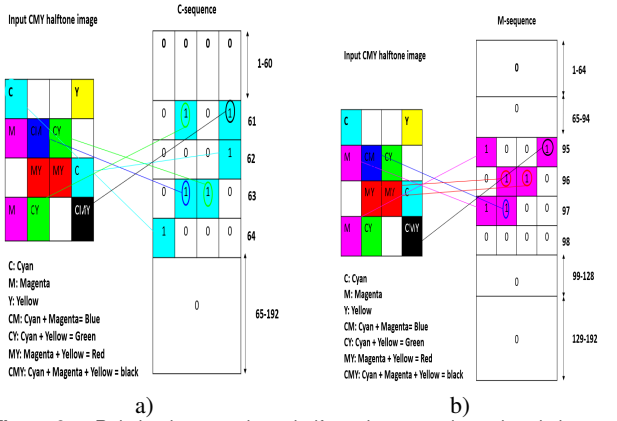
$$\text{number of swaths} = ceil((\text{image height} + 94 + 94)/64) \quad (2)$$

We use Fig. 6 as an example to illustrate how we generate the printer-addressable pixel map from an input halftone image. At the bottom of the image, we briefly describe what each color in our input sample image represents. This sample input image has size 4×4. We need three swaths to complete this print. Hence, we have a three-tuple sequence map with width 4 and height 192 ($3(ceil((4+94+94)/64)) \times 64$). Within each swath, 64 rows of our sequence map correspond to 64 nozzles of each color. More specifically, Row 1 to Row 64 of the C-sequence map describe the firing sequence of the cyan nozzles H1 to H64 during first swath. Similarly, Row 1 to Row 64 of the M-sequence map describe the firing sequence of the magenta nozzles H65 to H128 during the first swath. The same is true for the Y-sequence with respect to the yellow nozzles.

The cyan nozzles are always the first color nozzles to eject ink drops for our nail printer. Therefore, once we start printing, the printhead moves down 64 rows from its starting position so that cyan nozzle H64 is now aligned with the first row of the image region of the finger nail; and the second row is aligned with cyan nozzle H63. Because the sample input in Fig. 6 only has four rows, cyan nozzles H1 to H60 are just aligned with empty space, which results in all 0s in the sequence map from Row 1 to Row 60. By looking at the first row of the given input sample image in Fig. 6, only the first column needs to fire a cyan drop. Thus, only the first column of Row 64 of the C-sequence map has value 1; and all the other columns have value 0. Next, by considering the second row of the input image, which is equivalent to considering Row 63 of the C-sequence map, the second and third columns are required to fire cyan drops, which means that the second and third columns of the C-sequence Row 63 have value 1 and the other columns have value 0. We can analyze the third and fourth rows

	P1	P3	P5	P7	P9	P11	P13	P15	P2	P4	P6	P8	P10	P12	P14	P16
A1		25	51	77	103	129	155	181	18	44	70	96	122	148	174	
A2	19	45	71	97	123	149	175		12	38	64	90	116	142	168	
A3	13	39	65	91	117	143	169		6	32	58	84	110	136	162	188
A4	7	33	59	85	111	137	163	189		26	52	78	104	130	156	182
A5	1	27	53	79	105	131	157	183	20	46	72	98	124	150	176	
A6	21	47	73	99	125	151	177		14	40	66	92	118	144	170	
A7	15	41	67	93	119	145	171		8	34	60	86	112	138	164	190
A8	9	35	61	87	113	139	165	191	2	28	54	80	106	132	158	184
A9	3	29	55	81	107	133	159	185	22	48	74	100	126	152	178	
A10	23	49	75	101	127	153	179		16	42	68	94	120	146	172	
A11	17	43	69	95	121	147	173		10	36	62	88	114	140	166	192
A12	11	37	63	89	115	141	167		4	30	56	82	108	134	160	186
A13	5	31	57	83	109	135	161	187	24	50	76	102	128	154	180	

Figure 5. *Nozzle table showing assignment of each nozzle to a power line and address line.*



a) b)

Figure 6. *Relation between input halftone image and nozzle printing sequence maps: a) Input halftone image and output C-sequence. b) Input halftone image and output M-sequence. Note that the sequence maps are inverted vertically with respect to the input halftone image due to the fact that the printhead moves from top to bottom.*

of the input image in exactly the same manner as we did with the first two rows. After we complete the first swath, all required cyan drops have been fired, which means that we fire no more cyan drops during the second and third swaths. Thus, Row 65 to Row 192 of our C-sequence contain 0s. Meanwhile, during the first swath, the magenta and yellow nozzles are not yet aligned with the target image region on the finger nail. Thus, Row 1 to Row 64 of our M-sequence and Y-sequence maps have value 0.

Once the cartridge finishes the first swath, it moves down vertically by 64 rows; and the magenta nozzles are now aligned with the target image region on the finger nail. However, we need to keep in mind that the first row of the image region is not aligned with the 64^{th} magenta nozzle due to the 30 rows separation between the different sets of color nozzles. Instead, during the second swath, the first row of the image region is now aligned with the 34^{th} (64-30) magenta nozzle which corresponds to Row 98 $((swath\ number - 1) \times 64 + 34)$. Here, the *swath number* is the index of the current swath, where the swaths are numbered $1, \ldots 3$. The second row is now aligned with the 33^{rd} magenta nozzle, which corresponds to M-sequence map Row 97. Each column of the magenta sequence map is analysed in same way as the cyan sequence map. Finally, the yellow sequence map is generated by following the same procedure as for the cyan and magenta maps so we will not describe this process in more details.

Synchronization of cartridge motion with printing process

The whole printing pipeline starts by receiving the HRBs sequence map, which forces the cartridge to go to the default starting position. The cartridge then begins to move from left to right with acceleration. Once the cartridge arrives at the printing zone, where the first column of the input image will be printed on the finger nail, the cartridge switches to constant speed. At this point, we get the sequence of nozzles needed to eject drops for all even nozzles at this column. We then obtain the PA sequence from the given HRBs sequence for firing. After firing the whole column of even nozzles at this swath, the cartridge moves to the next column until we leave the printing zone area. Once the cartridge leaves the printing zone, where the last column of the input image will be printed on the finger nail, it starts to decelerate until it hits the destination position. The cartridge subsequently changes direction, and moves from right to left with acceleration. The procedure is the same as when it moves from left to right; but this time we only fire odd nozzles. After the cartridge hits the end at the left side, we have accomplished one swath of printing. If this swath is not the last swath, the cartridge moves down in the vertical direction so that it can print the next swath. The cartridge keep repeating the same process for each swath until it reaches the last swath. The whole printing procedure is then finished. Fig. 7 shows this whole process in a flow chart diagram.

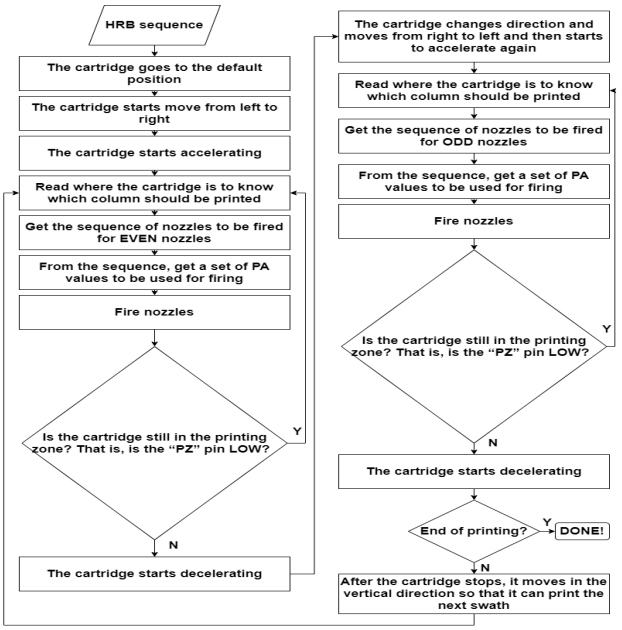


Figure 7. *Flow chart showing synchronization of cartridge motion with printing process*

Multi-pass Printing

Before digging into multi-pass printing, we want to first briefly review the more basic concept which is called the single-pass printing mode [1]. In our case, if we use a single-pass printing mode, each swath height is then equivalent to the number of nozzles we use for each color, which is 48. Therefore, after finishing the firing process of each swath, the cartridge would vertically move down 48 rows. More generally speaking, for a normal inkjet printer that moves the media vertically instead of the cartridge, between swaths, we would advance the media vertically by the swath height [1].

For multi-pass printing, we divide the target image region into N_{pass} horizontal zones, where N_{pass} is the number of passes to be used during printing. Each zone has height equal to $H_{zone} = H_{swath}/N_{pass}$, where H_{swath} is the total number of nozzles to be used to print each color. In our case $H_{swath} = 48$. Then after printing each swath, we only vertically advance the cartridge by H_{zone} pixels. In our case for 4-pass printing, $H_{zone} = 12$. More generally, any given pixel in the target image can be printed during any one of the N_{pass} passes by one of N_{pass} different nozzles.

The *print mask*, to be discussed in more detail later, determines on which pass any given pixel in the image is to be printed, and therefore, for each color which nozzle will be used to print that pixel.

Fig. 8 shows our 4-pass nozzle distribution for each color. The top and bottom 8 nozzles are the nozzles that not being used in order to improve image quality, as discussed earlier. The middle 48 nozzles, which is the H_{swath} of our cartridge, are divided into four sections, each corresponding to one of the four passes. Each section includes $H_{zone} = 12$ nozzles. The bottom most 12 nozzles are called first pass, the 12 nozzles above them are then called second pass, and so on. Thus, for an image of fixed size, four-pass printing will take 4 times longer than single-pass printing.

Figure 8. *Separation of nozzles into zones for 4-pass printing for each color.*

Then why should we use multi-pass printing when it definitely costs us more time than single-pass printing? The answer is that it can dramatically improve print quality. For example, let's consider that we decide to print a pure cyan solid-fill test page; and our cyan nozzle H56 is burned out. With single-pass printing, we would then observe a sequence of 1-pixel height horizontal white stripes in between every 47 rows of well-printed cyan. However, if we use 4-pass printing, those rows that would have originally been printed by nozzle H56 can now be printed by four different cyan nozzles according to the print mask. This can dramatically reduce the effect of the missing nozzle H56.

Four-pass input modification and output illustration

In order to use the four-pass printing mode, we need to also modify our input in several steps. The first step is called top zero padding where we add a specific number of rows of zeros on the top of our input halftone image. This step is not required for single-pass printing since the cartridge can directly move to the first 48 rows to eject drops. But for four-pass printing, we can only align the first pass nozzles to the corresponding first 12 rows of our target printing area. Hence, we need to imagine that there exist 36 rows of white space for our second, third, and fourth pass nozzles. Those 36 rows of white space mean 36 rows of zeros, since we do not need to fire any drops for those rows because they do not actually exist.

The second step is called color shifting. It is due to color separation. Let's consider the distance between the first cyan nozzle and the first magenta nozzle. It consists of two parts, the one is the distance of between the first and last cyan nozzles; and the other is the separation between the last cyan and the first magenta nozzle. The distance between the first cyan nozzle H1 and the last cyan nozzle H64 is 64. And the distance between the first magenta nozzle H65 and the last cyan nozzle H64 is 30, which was mentioned when we introduced our color cartridge. Hence, the distance between the first magenta nozzle H65 and the first cyan nozzle H1 is equal to $64+30=94$. We can think of this as the first magenta nozzle experiencing 94 rows of empty white space after the first cyan nozzle starts printing. Similarly, the distance between the first yellow nozzle H129 and the first magenta nozzle H65 is also 94. This results in the first yellow nozzle being $94+94=188$ apart from the first cyan nozzle. We shift our magenta and yellow color tuples according to these separations.

The last step is bottom zero padding. It first needs to satisfy the requirement that our modified input height should be divisible by 12. Then we add 36 rows at the end following the same reasoning used to determine the top zero padding. Equation 3 shows how we calculate the modified input height.

$$\begin{aligned} \text{height} &= \text{ceil}((36+94+94+900)/12)*12+36 \\ &= 1164 \end{aligned} \tag{3}$$

According to the modified input height, we can easily figure out our HRB output height according to Eqs. 4 and 5

$$\begin{aligned} \text{swath number} &= 1164(\text{modified input height})/12 \\ &= 97 \end{aligned} \tag{4}$$

$$\text{HRBs output height} = 97 \times 64 = 6208 \tag{5}$$

To provide an illustration of the modified input, Fig. 9 shows an example if we want to print a pure black page with input size 900 $\times$ 600 by representing the input, each modified color tuple, and the final combined input after modification.

Print mask

The print mask is used for multi-pass printing to decide during which pass each pixel should be printed [3]. It is a specific map to control the nozzles for firing drops. Unlike single-pass printing, which uses one nozzle to print a whole horizontal line, multi-pass printing needs to assign each pixel of a given row to different pass numbers which for a given color, indirectly specifies which nozzle is used to print each pixel in this row. For four-pass printing, each element in this masks is an integer between 1 and 4, specifying on which pass the pixel corresponding to this mask will be printed. It has the same size as the modified input with height 1164 and width 600. Different spatial arrangements of the numbers 1-4 can generate different print masks. We create three different masks which we call 1) the general mask, 2) the random permutation mask, and 3) the DBS mask. We will only briefly introduce the first two print masks since the DBS mask, which we are currently using, outperforms the other two.

The general mask is basically constructed by the sequence of 1-4 in order and then shifting this sequence by one in the

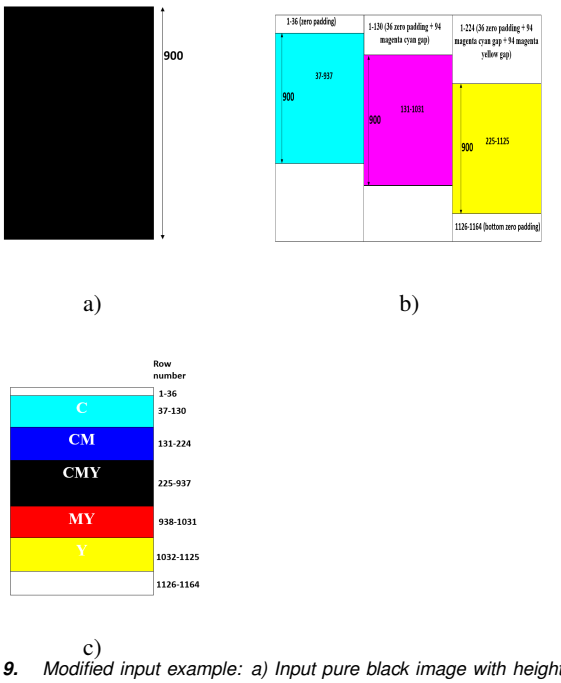


Figure 9. *Modified input example: a) Input pure black image with height 900; b) Modified input for each color tuple; c) modified input after combining the three tuples.*

next row. More specifically, the first row of the general mask is the sequence 12341234...1234, the second row is the sequence 23412341...2341, the third row is the sequence 34123412...3412, and so on. However, the print results with the general mask display some structured artifacts. So, we come up with another print mask, which is called random permutation mask. We randomly assign the same number of digits 1-4 to each print mask position. By using random permutation mask, the structured artifacts which occurred with the general mask disappear. But the prints made with the random permutation mask are too noisy. Fig.10 show an 16×16 example of those two masks. Each mask consists of four different colors. The corresponding four pass number is indicated by the color bar on the right. Even with this mask example, we can clearly see the structured artifacts of the general mask and the noisiness of the random permutation mask.

Therefore, we propose a third method for mask generation which called DBS mask since it used the DBS (Direct Binary Search) framework [4, 5]. For the DBS mask, we first generate a full size of random permutation mask and separate each pass map so that we have four masks each related to one pass. Then, we scan through each pixels in raster order for each mask and consider to swap the center not white pixel with neighboring white pixels. If the pixel at the corresponding mask satisfied the requirement for swapping, we swap the pixel while other masks remain unchanged. We can then calculate ΔE and accept the swaps that has the lowest ΔE, or otherwise don't swap any pixels. Since each pixel can only be assigned to one of the four pass, the swap will only involved with two masks and the other two would remain unchanged. Hence, ΔE are the sum of ΔE in each changed mask. This DBS mask notably decrease the noisy level of the original random permutation mask. To represent quality difference with

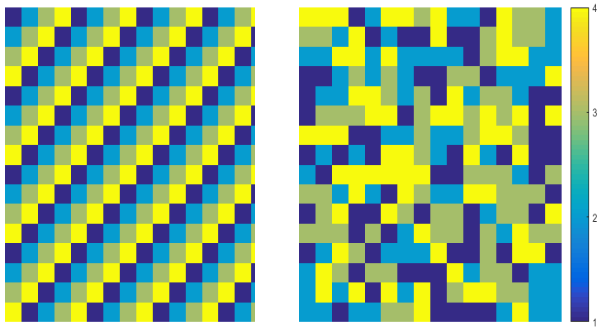


Figure 10. *Size 16×16 mask examples: a) general mask. b) random permutation mask.*

different print masks, we use Fig. 11 to compare the results with different print masks implemented. The top image shows the difference between the general mask and the random permutation mask. The bottom image compares the results with the DBS mask and the random permutation mask.

Two drop printing

Multi-drop printing, similar to multi-pass printing, also intends to improve print quality [6]. But differ than multi-pass printing, which is used to mitigate the missing nozzle effects. Multi-drop printing dedicates to enhance the image color saturation, or in other word, to make the color of prints to be more vivid. In the contrast of normal one drop printing, which fires at most one drop at each pixel, two drop printing allows to fire at most two drops at each pixel. Fortunately, using together with multi-pass printing, multi-drop printing will not increase the print time one more time within certain limit drops given by how many pass we using. For example, in this paper, we use four-pass printing, which enable us to use up to four-drop printing that allows to fire four drops at each pixel without increasing print time. However, by comparing several print results from one-drop printing to four-drop printing, we found out that two drop printing works the best for our nail printer, which enhances color saturation while not results in too dark prints.

But how does two drop printing work? Stopping the cartridge immediately and firing the second drop is probably not a good idea. First, we don't have a driver could achieve this function. Second, even with an idealized motor driver that satisfies the requirement, it will cost us more time to finish printing. That is why multi-pass printing benefits our multi-drop printing without using more time. Recall that for multi-pass printing, we designed the print mask that each pixel of the input is printed by a specific nozzle by given the different pass number. What if we give a pixel two different pass numbers, which results in firing two drops by two different nozzles in two different passes at the given pixel. Two different print masks are used to assign two different pass numbers. We just need to make sure that the second print mask is totally different than the first print mask otherwise it will result in the case that one nozzle need to eject two drops at exact same location without stopping the cartridge. We currently design our second print mask based on the information of the first print mask

with some simple mathematical equation. As Eq. 6 showed, if the pass number was 1 at a given pixel, print mask2 switches it to 4. By Eq. 6, we can certify that two print masks would not generate same pass numbers at the same pixel.

$$\text{mask2} = 5 - \text{mask1} \tag{6}$$

To understand our two drop printing better, we briefly go through the whole printing pipeline for two drop printing. Given input multi-bit CMY halftones and output PA sequence information. First, we generate a DBS print mask which is called mask1 and then obtain our mask2 by Eq. 6. Note that we now need two input halftone images because of two drop printing. For the halftone images, using cyan as an example, if we don't need to eject any cyan drops at a pixel, both halftone images cyan tuple would be zero at that pixel. If we want to fire one cyan drop at a pixel, halftone image1 cyan tuple would be one while halftone image2 cyan tuple will be zero. If we need to fire two cyan drops at a pixel, both halftone images cyan tuple will be one at that pixel. Therefore, we input two halftone images and each of them corresponding to one print mask that we generated in previous step.

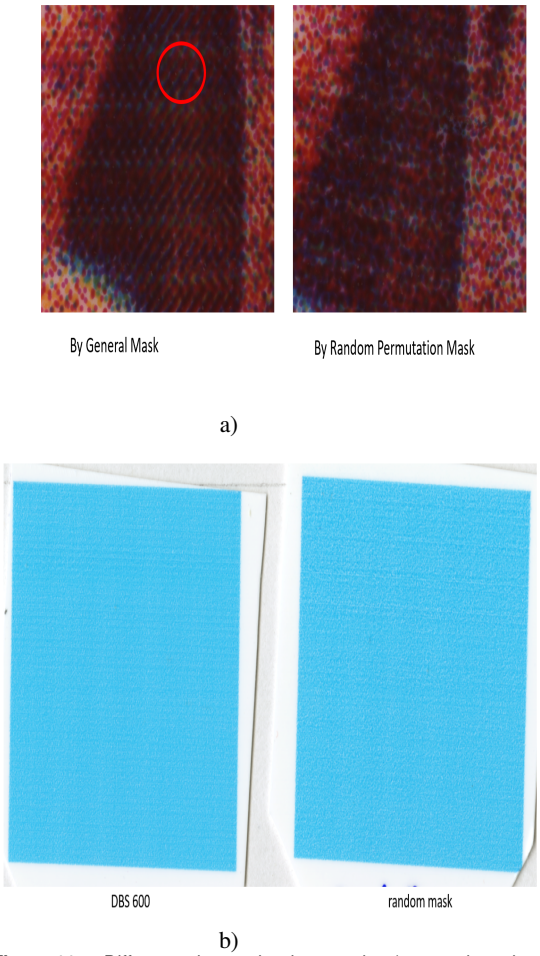


Figure 11. *Different print mask prints result: a) general mask vs random permutation mask. b) random permutation mask vs DBS mask.*

According to these two separate inputs, we produce two HRBs sequence maps of printer-addressable pixel with same size and we merge two sequence maps into one by adding them pixel-wise, which means we add the two numbers in the same pixel together from two sequence maps. Finally, we obtain our PA (power line and address line) sequence based on the merged HRBs sequence map to control nozzles firing.

Conclusion

In this paper, we introduced the procedure that how we generate HRBs from an input halftone image and we showed the whole printing pipeline in the last subsection of HRBs, which reader might obtain some basic cognition for how does an inkjet printer work. We also represent that how did the DBS print mask outperformed the other two print masks. Our experiments successfully verified that multi-pass and multi-drop printing dramatically improve our printing quality. Thus, instead of using convenient single-pass printing, we consider that spend more time for multi-pass printing for much better quality prints.

References

[1] Peter A. Torpey, Multipass printing in an ink-jet device, Proc. SPIE 3018, Color Imaging: Device-Independent Color, Color Hard Copy, and Graphic Arts II, (4 April 1997).

[2] Anthony D. Parkhurst, Ramchandran Padmanabhan, Steven D. Mueller, and Kirt A. Winter, Connectivity of the HP DeskJet 1200C Printer, Hewlett-Packard Journal, (February 1994).

[3] Jonathan Yen, Mats Carlsson, Joan Manel Garcia, and Hugh Nguyen, Constraint Solving for Inkjet Print Mask Design, J. Imaging. Sci. and Technol., 44, 391-397 (2000)

[4] Boley, J. William, Jan P. Allebach, and George T-C. Chiu. "Direct Binary Search for Print Mask Design in Inkjet Printing." NIP Digital Fabrication Conference. Vol. 2011. No. 2. Society for Imaging Science and Technology, 2011.

[5] Lieberman, David J., 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 9.11 (2000): 1950-1963.

[6] Askeland, Ronald A., and Paul E. Hunter. Hybrid multi-drop/multi-pass printing system. U.S. Patent No. 6,193,347. 27 Feb. 2001.

[7] Documents by SunValleytek International, Inc.

Author Biography

Sige Hu *received his BS in Electrical and Computer Engineering from Pennsylvania State University (2017) and is currently pursuing PhD in Electrical and Computer Engineering at Purdue University. His research is mainly focus on image processing.*

Daulet Kenzhebalin *received his BS in computer engineering from Purdue University (2015). Currently he is pursuing a PhD in electrical engineering at Purdue University. His primary area of research has been image processing and machine learning.*

Baekdu Choi *received his B.Sc. in electrical and computer engineering from Seoul National University, Seoul, South Korea in 2017 and is currently working on a Ph.D. in electrical and computer engineering at Purdue University, West Lafayette, IN, USA. His research mainly focuses on digital image processing, digital halftoning and color management for inkjet printers.*

George T. Chiu *is a Professor in the School of Mechanical Engineering with courtesy appointments in the School of Electrical and Com-*

puter Engineering and the Department of Psychological Sciences at Purdue University. He also serves as the Assistant Dean for Global Engineering Programs and Partnership for the College of Engineering. Dr. Chiu received the B.S. degree in Mechanical Engineering from the National Taiwan University in 1985 and the M.S. and Ph.D. degrees in Mechanical Engineering from the University of California at Berkeley, in 1990 and 1994, respectively. From September 2011 to June 2014, he served as the Program Director for the Control Systems Program at the National Science Foundation. His current research interests are mechatronics and dynamic systems and control with applications to digital printing and imaging systems, digital fabrications and functional printing, human motor control, motion and vibration perception and control. He received the 2012 NSF Director's Collaboration Award and the 2010 IEEE Transactions on Control System Technology Outstanding Paper Award. He served as the Editor-in-Chief for the IEEE/ASME Transactions on Mechatronics from 2017-19 and as the Editor for the Journal of Imaging Science and Technology from 2012-14. Dr. Chiu served on the Executive Committee of the ASME Dynamic Systems and Control Division (DSCD) from 2007 to 2014 and as the Chair of the Division from 2012-13. He is a Fellow of ASME and a Fellow of the Society for Imaging Science and Technology (IST).

***Peng (Davi) He** is a software manager in Sunvalleytek International Inc, Shenzhen, Guangdong, China. He received his B.S in Communication Engineering from Hunan University of Arts and Science, Changde, Hunan, China in 2012.*

***Zillion (Zhenqing) Lin** is a technical director in Sunvalleytek International Inc, Shenzhen, Guangdong, China. He received his BS in Microelectronics Engineering from University of Electronic Science and Technology of China, Chengdu, Sichuan, China in 2008 and MS in Biomedical Engineering from Shenzhen University, Shenzhen, Guangdong, China. His current research interest including artificial intelligence and robot.*

***Jan P. Allebach** is Hewlett-Packard Distinguished Professor of Electrical and Computer Engineering at Purdue University. Allebach is a Fellow of the IEEE, the National Academy of Inventors, the Society for Imaging Science and Technology (IS&T), and SPIE. He was named Electronic Imaging Scientist of the Year by IS&T and SPIE, and was named Honorary Member of IS&T, the highest award that IS&T bestows. He has received the IEEE Daniel E. Noble Award and the IS&T/OSA Edwin Land Medal, and is a member of the National Academy of Engineering*

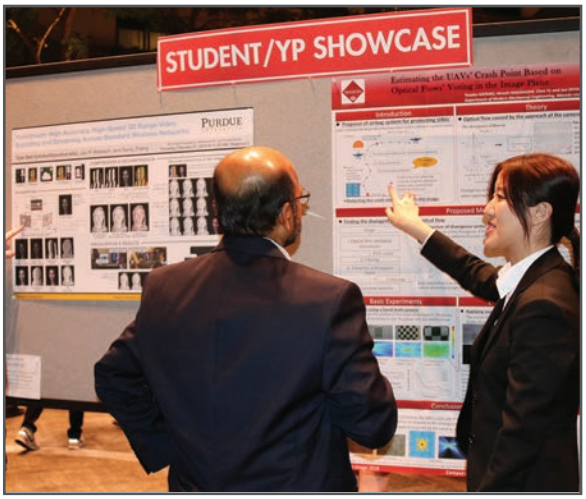
JOIN US AT THE NEXT EI!

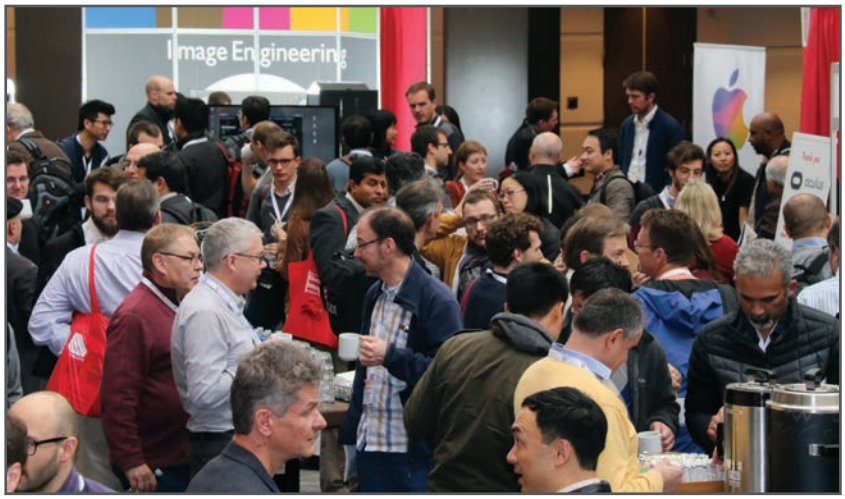
IS&T International Symposium on

Electronic Imaging

SCIENCE AND TECHNOLOGY

Imaging across applications . . . Where industry and academia meet!





• SHORT COURSES • EXHIBITS • DEMONSTRATION SESSION • PLENARY TALKS •
• INTERACTIVE PAPER SESSION • SPECIAL EVENTS • TECHNICAL SESSIONS •

www.electronicimaging.org

IS&T
imaging.org