Dynamic Print Mode Control for Inkjet Printing

Mustafa Kamasak, Anand V. Deshpande, Kok Leong Thoon, Charles A. Bouman, George T.-C. Chiu, and Jan P. Allebach
Purdue University, W. Lafayette, IN 47906, USA

Abstract

We introduce a novel printing scheme that allows arbitrarily shaped regions in a document to be printed using different modes. Our implementation £rst segments the document into two basic modes: one pass and four pass. The one pass mode results in greater print speed but has lower quality, whereas the four pass mode results in greater quality but lower print speed. Typically, the one pass mode is more suitable for text and graphics, while the four pass mode is more suitable for continuous tone pictures.

We demonstrate our method using a Lexmark Z52 print engine controlled by a Texas Instruments C6211 DSP. The resulting dynamic print mode control system can be used to continuously trade-off print speed for print quality, or to optimize print quality at a desired print speed.

1. Introduction

Typically, ink jet printers use a variety of different print modes to control the quality and speed of printing. In one pass printing, each horizontal motion of the print head is used to print a full swath of the image. This print mode is fastest, but it can result in print quality defects, particularly at swath boundaries. One pass printing is also not robust to defects or nonuniformities in the print nozzles. Alternatively, in multi-pass printing, each pixel is covered by more than one pass of the print head, and the order that dots are £red is determined by a print mask [1]. Multi-pass printing is much slower, but it produces much higher print quality, particularly for pictures and continuous tone graphics.

A limitation of typical printers is that they must print the entire document using a single print mode. This means that if a small region of the document requires the quality of multi-pass printing, then the entire document must be printed using multi-pass printing; thereby substantially reducing print speed.

In this paper, we introduce a novel printing system which we call dynamic print mode control. This system allows arbitrary regions of the document to be printed using different modes. So a small region containing a photo can be printed using four pass bi-directional printing, while the remaining portions of the document can be printed using one pass bi-directional printing. The print mode of each pixel is individually determined by £rst segmenting the document into one pass and four pass pixel types. For our purposes, we use a very simple segmentation method based on the point-wise gray level of each pixel.

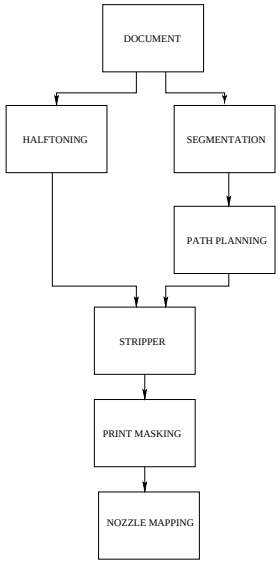
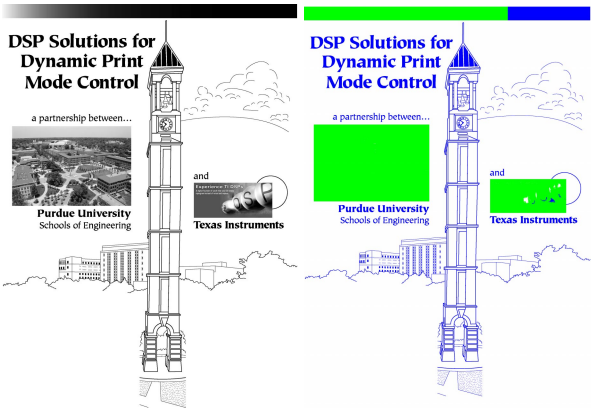


Figure 1: *Overview of dynamic print mode system.*

Dynamic print mode control allows for a continuous trade-off between the print-speed and print-quality of the document. It also allows the quality to be optimized at a particular print speed by selecting the multi-pass print modes for the portions of the document that are most sensitive to printing artifacts.

2. The Dynamic Print Mode System

Figure 1 shows an overview of the architecture we used for implementing the dynamic print mode system. We used a layered architecture to both simplify implementation, and allow for portability to print mechanisms other than the Lexmark Z52 platform used in this work.



(a) (b)

Figure 2: *a) The 600 dpi 8 bit raster test image used in the experiments. b) Segmentation of test image into no print (white), 1 pass (blue), and four pass (green) regions.*

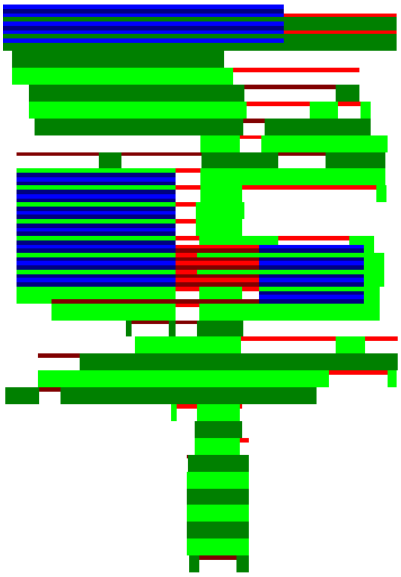


Figure 3: *A diagram illustrating the path chosen to allow printing of each pixel with the desired mode. The color key is given by the following: Light Green ⇔ 1 pass left to right (1PLR); Dark Green ⇔ 1 pass right to left (1PRL); Light Blue ⇔ 4 pass left to right (1PLR); Dark Blue ⇔ 4 pass right to left (4PRL); Light Red⇔ no print left to right (NPLR); Dark Red⇔ no print right to left (NPRL).*

The document to be printed is assumed available as input in 8 bit raster form. For these experiments, we used Adobe Illustrator to render the original postscript document as a 600 dpi 8 bit raster with no anti-aliasing. The document is then processed along two paths. The £rst path performs binary halftoning. The second path segments the image into regions corresponding to different print modes, and then plans the path that the print head will take as it moves across the page. We refer to each each pass of the print head for a £xed paper position as a print swath.

The £nal three operations use the segmentation, path plan, and halftone as input. The ”stripper” operation extracts the binary image strips corresponding to the path of the inkjet head as it moves across a swath. The print mask then determines exactly which pixels are printed and which are not on each pass of the print head. We will see that the print mask is critical in multipass printing since it can be used to minimize print defects. Finally, the nozzle mapping organizes the output data so that it is time sequenced to suit the speci£c nozzle format of the printer.

2.1. Segmentation

The function of the segmentation step is to label each pixel with its required print mode. For this work, we use three print modes.

- **No print (NP)** - These are pixels which are not printed i.e. white space.
- **1 Pass (1P)** - These are pixels that are printed in one pass print mode.
- **4 Pass (4P)** - These are pixels that are printed in four pass print mode.

Figure 2 shows the test image and its segmentation into these three modes. We use a very simple segmentation algorithm computed directly from the value of each 8 bit pixel.

gray level 0 ⇔ No print (NP)

gray level 255 ⇔ 1 pass (1P)

other gray levels ⇔ 4 pass (4P)

Note that the segmentation can have regions of arbitrary shape. So, for example, a small region of four pass pixels can exist inside a region of one pass pixels. This is very important since it allows the mode of each pixel to be chosen in a manner which is best, independently of other regions in the document.

2.2. Path Planning

Once the document is segmented, the path of the print head must be planned so that each pixel can be printed using the desired mode. More speci£cally, each 1P pixel must be covered by at least one pass of the print head, and each 4P pixel must be covered by exactly four passes of the print head. We present a relatively simple method for path planning, but more complex methods could be used to further minimize head motions.

The basic head motions are itemized below.

- 1 pass left to right (1PLR)
- 1 pass right to left (1PRL)
- 4 pass left to right (4PLR)
- 4 pass right to left (4PRL)
- No print left to right (NPLR)
- No print right to left (NPRL)
- Paper advance by $N/4$ (PA)

The paper advance (PA) operation moves the paper forward by $N/4$ where $N = 192$ is the height of the print head in pixels for our system. We refer to the first four operations (i.e. 1PLR, 1PRL, 4PLR, 4PRL) as printing head motions, since the remaining operations do not cause ink to be fired from the print head.

Depending on the initial position of the head, the following initial set of head motions is chosen.

```
If(HeadPosition == left){                    (1)
 NewMotions =
  (1PLR, PA, 4PRL, PA, 4PLR, PA, 4PRL, PA)
}
If(HeadPosition == right){
 NewMotions =
  (1PRL, PA, 4PLR, PA, 4PRL, PA, 4PLR, PA)
}
```

Note that there are four PA operations in the sequence, so at the end of the sequence the paper has advance a full print head height. Next the 1PLR,1PRL,4PLR, and 4PRL head motions are processed to determine the horizontal starting and ending position of the head for each motion.

- For 1PLR or 1PRL

```
If(number of 1P or 4P pixels in swath == 0)
   empty = true; otherwise empty = false;
If(empty == false) {
   Start = first position containing 1P or 4P pixel.
   Stop = last position containing 1P or 4P pixel.
}
```

- For 4PLR or 4PRL

```
If(number of 4P pixels in swath == 0)
   empty = true; otherwise empty = false;
If(empty == false) {
   Start = first position containing 4P pixel.
   Stop = last position containing 4P pixel.
}
```

Note that one pass operations consider both 1P and 4P pixels, whereas four pass operations only consider 4P pixels. Since any head motion that is empty results in no printing, it is deleted from the NewMotions sequence.

After deletions of empty head motions, the print head may finish in either the left or right position. For example, the following sequence might result after deletions of empty head motions, leaving the head in the right position.

$$(1PLR, PA, 4PRL, PA, 4PLR, PA, PA) \qquad (2)$$

Depending on the final state of the head, we set the HeadPostion variable to either $left$ or $right$. Then the new head motions are concatenated to the existing head motions

$$\text{HeadMotions} \leftarrow \text{cat}\left(\text{HeadMotions}, \text{NewHeadMotions}\right),$$

and the process is then repeated starting with (1) until the end of the page is reached.

This algorithm produces a list "HeadMotion" which contains a sequence of print head movements that are not necessarily connected horizontally. In other words, the horizontal stop position of one head movement may not correspond to the horizontal start position of the subsequent head movement. To fix this, we insert NPLR and NPRL commands as necessary to connect the head motions. We note that during the NPLR and NPRL motions the head moves substantially faster to reduce printing times.

Finally, the path plan is optimized inserting by NPLR and NPRL operations in white regions of the document. So for example, a command such as 1PLR may contain a white space in which the print head can be accelerated. In this case, the command might be replaced with the sequence 1PLR, NPLR, 1PLR, depending on the size of the white space and the dynamics of the print mechanism.

Figure 3 shows the result of applying this path planning algorithm to the test page. For the purposes of illustration, the 1 pass motions are printed as color stripes that are $N = 192$ pixels high, and 4 pass motions are printed as color stripes that are $N/4 = 48$ pixels high. However, the color corresponding to later motions overwrites earlier motions; so some green strips are less than 192 rows high.

2.3. Halftoning and Stripping

We use Floyd and Steinberg error diffusion to convert the 300 dpi 8 bit raster document into a 300 dpi binary raster document [2]. Once the halftoning is complete, the halftone, path plan, and segmentation are sent to the "stripper". In our implementation, the stripper extracts the halftone and segmentation data required for each printing head motion. The extracted data is stored in data structures which are individually processed by the print masking function. This

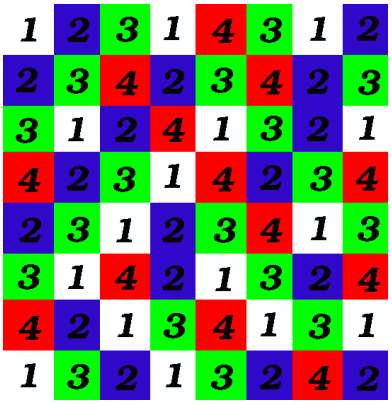


Figure 4: *The* 8×8 *print mask used in the experiments.*

approach simpli£es implementation, but increases computation and memory requirements.

2.4. Print Masking

In four pass printing, each pixel can be printed in any one of four possible passes of the print head. The speci£c pass used to print the pixel is determined by the print mask. Let i be the index of the print nozzle, where $i = 0$ indicates the top row, and $i = 191$ indicates the bottom row of the nozzle. When four pass printing is used, the pass number of nozzle i is given by

$$PixelPassNumber = (191 - i)\%48\ .$$

Note that the bottom 1/4 of the nozzles are always used to print the £rst pass, and the top 1/4 are always used to print the last pass.

A dot is £red when the pixel's pass number is equal to the entry in the tiled print mask. We used an 8×8 print mask, as shown in Fig. 4. We designed our print mask to minimize the number of adjacent pixels that are £red sequentially [1]. So the test for £ring a 4 pass pixel is

$$Mask[x\%8][y\%8] == PixelPassNumber$$

where x and y are the absolute position of the printed pixel on the page.

Figure 5 shows pseudo-code specifying the logic used to £re dots in the dynamic print mode system. The decision to £re a dot is based on both the mode of the print head motion (i.e. 1PLR, 1PRL, 4PLR, or 4PRL) and the class of the pixel (i.e. 1P and 4P). In particular, one pass printing motions are used to print both one pass and four pass pixels, whereas four pass printing motions are only used to print four pass pixels. This is the key factor which allows regions with differing print modes to have arbitrary shape.

```
For each printed strip {
  if(Head Motion == 1PLR or 1PRL) {
    for each pixel in strip {
      test1 = (segmentation[x][y] == 1P);
      test2 = (Mask[x%8][y%8]==PixelPassNumber);
      if(test1 or test2)
        fire dot at position [x,y] on page;
      else
        do not fire dot;
    }
  if(Head Motion == 4PLR or 4PRL) {
    for each pixel in strip {
      test1 = (segmentation[x][y] == 4P);
      test2 = (Mask[x%8][y%8]==PixelPassNumber);
      if(test1 and test2)
        fire dot at position [x,y] on page;
      else
        do not fire dot;
    }
  }
}
```

Figure 5: *Pseudo-code illustrating the logic used to £re dots in the dynamic print mode system. Note that one pass printing motions are used to print both one pass and four pass pixels, whereas four pass printing motions are only used to print four pass pixels.*

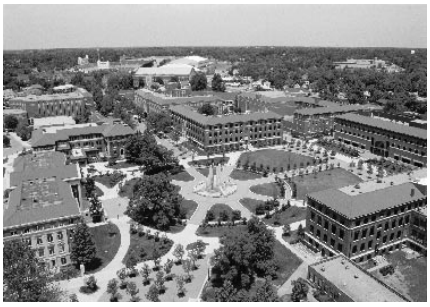
2.5. Nozzle Mapping

The last function is nozzle mapping. This function reorganizes the data in a form suitable for sequential input to the print head.

2.6. Experimental Results

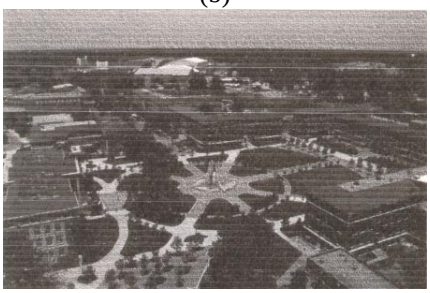
This dynamic print mode control system was implemented on a Lexmark Z-52 printer mechanism. A Texas Instruments Tiger II DSP board based on the C6211 VLIW DSP was used to control carriage motion, paper motion, and nozzle £ring in real-time. The system was successfully demonstrated at the Comdex 2000 exhibition. While the system was developed for full bi-directional printing, we limit our examples to left-right printing in order to eliminate artifacts due to horizontal misregistration.

Figures 6 and 7 contain results scanned from actual printer output. Figures 6 shows results of printing an image in both one pass and four pass print modes. Note that the image quality is substantially improved by using the four pass print mode since the print mask covers most of the banding artifacts generated by variations in nozzle performance. Figure 7 shows portions of the printed output page. Region a), which was printed with one pass mode, shows some defects due to nonuniformities in nozzle output. However, this region is printed with much greater speed. Region b), which was printed with four pass mode, has no visible banding artifacts because they are hidden by the print mask. However, this region is printed at approximately 1/4 the speed of the one pass regions.



(a)

(b)



(c)

Figure 6: *a) A 600 dpi original image. The same image rendered using b) four-pass printing, and c) one-pass printing.*

(a)

(b)

Figure 7: *Portions of printed output page. a) A region printed with one pass mode shows some defects due to miss-£ring nozzles. However, this region is printed with greater speed. b) A region printed with four pass mode shows no banding defects because defects are hidden by the print mask. However, this region is printed at approximately 1/4 the speed of the one pass mode.*

3. Conclusions

We introduced a novel printing scheme that allows arbitrarily shaped regions in a document to be printed using different modes. The one pass mode results in greater print speed but has lower quality; whereas the four pass mode results in greater quality but lower print speed. Typically, the one pass mode is more suitable for text and graphics, and the two pass mode is more suitable for pictures. More sophisticated segmentation algorithms could be used to produce a more optimal trade-off between print quality and print speed.

Acknowledgement

We would like the thank Texas Instruments for their support and Jim Bearss for his creative input.

References

[1] J. Yen, M. Carlsson, M. Chang, J. Garcia, and H. Nguyen, “Constraint solving for inkjet print mask design,” *Journal of Electronic Imaging*, vol. 44, no. 5, pp. 391–397, Sept./Oct. 2000.

[2] R. W. Floyd and L. Steinberg, “An adaptive algorithm for spatial greyscale,” *Journal of the Society for Information Display*, vol. 17, no. 2, pp. 75–77, 1976.