Segmentation and Automatic Descreening of Scanned Documents

Alejandro Jaimes[a], Frederick Mintzer[b], A. Ravishankar Rao[b] and Gerhard Thompson[b]

[a]Columbia University
Department of Electrical Engineering
New York, NY 10027

[b]IBM T.J. Watson Research Center
Yorktown Heights, NY 10598

ABSTRACT

One of the major challenges in scanning and printing documents in a digital library is the preservation of the quality of the documents and in particular of the images they contain. When photographs are offset-printed, the process of screening usually takes place. During screening, a continuous tone image is converted into a bi-level image by applying a screen to replace each colour in the original image. When high-resolution scanning of screened images is performed, it is very common in the digital version of the document to observe the screen patterns used during the original printing. In addition, when printing the digital document, moiré effects tend to appear because printing requires halftoning. In order to automatically suppress these moiré patterns, it is necessary to detect the image areas of the document and remove the screen pattern present in those areas. In this paper, we present efficient and robust techniques to segment a grayscale document into halftone image areas, detect the presence and frequency of screen patterns in halftone areas and suppress their detected screens. We present novel techniques to perform fast segmentation based on *α-crossings*, detection of screen frequencies using a *Fast Accumulator Function* and suppression of detected screens by low-pass filtering.

Keywords: descreening, dehalftoning, page segmentation, document image analysis, printing on-demand, digital library.

1. INTRODUCTION

The advent of digital libraries and advances in scanning and printing technologies have opened up new possibilities for reproduction of printed materials. In one application, for example, IBM printing equipment and software are being used for an on-demand book printing service. Books that are out of print are being scanned and stored in a digital library for on-demand printing. When a book is ordered, in any quantity, it is printed directly from the digital library.

One of the major quality challenges encountered in this application is the suppression of moiré patterns or artifacts that appear in screened pictures when the book is printed from the library. When pictures in the original books are offset printed, a screening process takes place. When those screened pictures are scanned and printed from the digital library, moiré effects are very likely to appear. These effects can be eliminated if the screen in the scanned picture is removed.

In this paper, we address image processing issues in creating a solution for this on-demand-printing problem. Once an image is generated by scanning, the following steps are taken: segmentation of the image into text and screened picture areas; image skew angle detection; detection and determination of screen frequencies in screened picture areas; removal of screened picture screens (descreening).

We present novel algorithms, which are robust and computationally efficient for this processing, placing special emphasis on the effective detection and removal of screens in picture areas. Effective descreening is a crucial step in removing the moiré effects that appear in the pictures when they are printed.

[a] Alejandro Jaimes is with the Image and Advanced TV Laboratory & Electrical Engineering Department, Columbia University, 1312 SW Mudd Mailcode 4712 Box No. F8 New York, NY 10027. E-mail: ajaimes@ee.columbia.edu Web: http://www.ee.columbia.edu/~ajaimes
[b] Frederick Mintzer, Ravishankar Rao and Gerhard Thompson are with IBM TJ Watson Research Center E-mail: {mintzer, ravirao, thompson}@research.ibm.com

For segmentation, we use a simple technique based on *α-crossing* density, which is similar to zero-crossings except that a threshold α is used instead of zero. The *α-crossing* density is a measure of spatial frequency and yields different values for picture[1] and text areas. Picture areas typically exhibit high spatial frequency and text areas exhibit low frequency. The segmentation results in a set of bounding boxes where areas inside correspond to pictures and areas outside correspond to text or background. To accurately compute screen frequencies, it is necessary to compensate for any rotation that occurs during scanning- we assume the orientation or skew angle of the scanned page has been corrected. Once bounding boxes are obtained, screen patterns in the picture areas are detected by applying a *Fast Accumulator Function* in both the horizontal and vertical directions. Since the function acts as an accumulator in each dimension, this yields two one-dimensional vectors for which dominant frequencies are found by analysis of the corresponding Fourier power spectra. Removal of the screen is then performed by applying a smoothing filter chosen according to the detected frequencies in the horizontal and vertical directions. Once the pictures are descreened, they can be enhanced to improve their contrast and sharpness.

We present a robust approach for the detection of the halftone frequency and the removal of screens produced by different halftone patterns. The use of these image-processing techniques together with operator review provide an acceptable solution to the on-demand-printing problem described above.

In the next sections, we describe segmentation, detection of screen frequency, and removal of the screen in image areas.

1.1 Related work

Kang [Kang] and Ulichney [Ulichney] provide good overviews of different halftoning methods. Previous work in document image segmentation is described in [Jain, Pavlidis]. Detection of screen frequencies, has been done by [Liu96], but in most cases direct analysis of the Fourier spectra is required. This is a computationally expensive operation. The problems of descreening and inverse halftoning have been studied in the literature. Wong [Wong95] describes a method to reconstruct error-diffused images. Xiong *et al* [Xiong97] present a technique to perform inverse halftoning using wavelet transforms. Both techniques focus on error-diffused halftones. Rao *et al* [Rao98] present a technique to descreen bi-level images. The present paper is an extension of that work to grayscale images.

2. SEGMENTATION INTO TEXT AND HALFTONED AREAS

2.1. *α-Crossings*

In order to reduce the moiré effects that appear in the picture areas of the document, it is necessary to first segment the document into picture and non-picture areas. To perform that segmentation, we make use of the difference in variations of pixel intensity values between picture and text areas. Inspecting intensity values in an area of the document along one dimension yields a signal such as the ones illustrated in figure 1. Selecting a value of α, we can count the number of times the signal crosses this value and use this as the base to perform the segmentation, given that these variations differ for picture and non-picture areas, as will be shown later. A more detailed definition of α-crossings is given below, but first we show how the image is subsampled for segmentation and how to determine values for α.

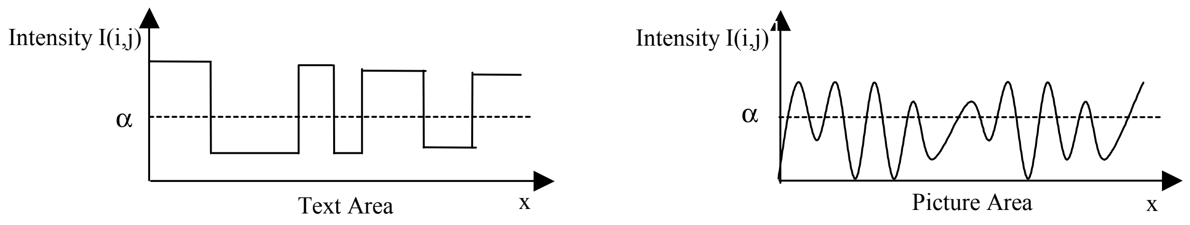


Figure 1: Example of pixel intensity values along a row of text and picture areas. Variation in the number of *α-crossings* is apparent, as is the shape of the curves- text intensities have less variation.

[1] To avoid confusion, throughout the paper, we use the word *picture* to refer to a halftone area and *image* to refer to the entire document.

Segmentation of the document into text and picture areas basically implies labeling each pixel using a pre-defined set of labels. We use the labels $l_1=1$ for pixels that belong to picture areas and $l_2=0$ for pixels that do not belong to picture areas (including text and background). Given that it is not quite necessary to differentiate at the pixel level (no picture is that small) we can assign labels to groups of pixels instead. We define a rectangular window W of size n x m, which is used to inspect a group of pixels from the document and give them a common label. Decisions as to whether the pixels in the window correspond to a picture area or not are taken by examining the number of *α-crossings* (see figures 1 & 2) in the x and y directions that occur in W.

We perform two 1-dimensional operations in which intensity values I(i,j) of neighboring pixels (pixel location in the image given by i and j indices on the horizontal and vertical direction respectively) are examined in a scan-line fashion inside the window W (figure 2).

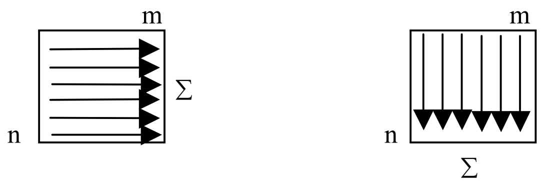


Figure 2: Horizontal and vertical scans within an n x m window W, to count the number of *α-crossings*. Scan direction (up to down, left to right) is not relevant, but orientation is. The total number of *α-crossings* in the window corresponds to the sum of *α-crossings* in the horizontal and vertical directions.

We define the binary functions $Z_x(i,j,\alpha)$ and $Z_y(i,j,\alpha)$ which have a value of one if an *α-crossing* exists and zero if there is no *α-crossing*. More specifically:

$$Z_x(i,j,\alpha) = \begin{cases} 1 & \text{if } I(i,j) > \alpha \text{ and } I(i+1,j) <= \alpha \text{ or} \\ & \text{if } I(i,j) < \alpha \text{ and } I(i+1,j) >= \alpha \\ 0 & \text{Otherwise} \end{cases} \qquad Z_y(i,j,\alpha) = \begin{cases} 1 & \text{if } I(i,j) > \alpha \text{ and } I(i,j+1) <= \alpha \text{ or} \\ & \text{if } I(i,j) < \alpha \text{ and } I(i,j+1) >= \alpha \\ 0 & \text{Otherwise} \end{cases} \qquad (1)$$

The number of *α-crossings* per unit area (window) depends on the variation in intensity values and on the choice of the α value. It is expected that the number of *α-crossings* will be different for picture and text areas. As can be seen clearly in figure 1, the choice of a α is very important since that value will affect the measures given by $Z_x(i,j,\alpha)$ and $Z_y(i,j,\alpha)$. Experimentally, we determined α=128 as a good value given that most text is printed in black on a white background and the range of intensity values for grayscale images is 0-255. This parameter, however, should not be fixed since the measures given by $Z_x(i,j,\alpha)$ and $Z_y(i,j,\alpha)$ would depend on the range of intensity values in the image. Assume, for instance, that we have an image with values between 0 and 255. If we scale those values to be between 0 and 100 instead, the image will be darker, but the content will not change; in this case, we would obtain very different $Z_x(i,j,\alpha)$ and $Z_y(i,j,\alpha)$ values for the same document image.

In order to select this parameter dynamically and alleviate this problem, we can compute the average intensity value for the pixels in the window W and use that average as the α in the Z_x and Z_y functions. Instead, to make the process more efficient computationally, we choose to compute α using only the pixel intensity values in the main diagonal of the area covered by the window (figure 3). In the case where m=n, $\alpha=(Tr(W)/n)$; that is, α is the average value of the trace of the square matrix represented by the window W.

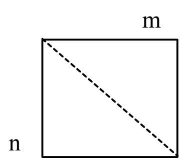


Figure 3: Sampling window of size n x m used to compute *α-crossing* density. Only pixel values along the main diagonal are used to compute α, as defined by (2).

Therefore the value of α is given by:

$$c = \min(m-1, n-1)$$
$$\alpha = (1/c) * \left(\sum_{i=0}^{c} I(i, i)\right) \qquad (2)$$

The value of α is adaptive and depends on the local values. Using the definitions in (1) and (2), we compute the *α-crossing* density which corresponds to the sum of the number of *α-crossings* in the horizontal and vertical directions in the area of the window W:

$$Z_d(k,l,\alpha) = \sum_{k=0}^{m-1} \sum_{l=0}^{n-1} (Z_x(i,j,\alpha) + Z_y(i,j,\alpha)) \qquad (3)$$

where k and l are the indices in the image along the x and y directions on which the window W is centered and the indices i and j correspond to the local x and y coordinates in the window W.

The window W, which is used to compute the α-crossing density, serves as a sampling function with horizontal period 1/h and vertical period 1/v (figure 4).

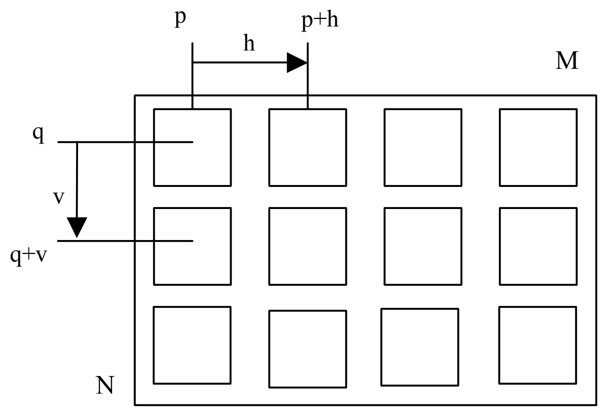


Figure 4: Sampling pattern for computing the *α-crossing* density in a document image of size N x M. In this particular example, since the sample frequencies h and v are larger than m and n respectively, there is no overlap in the computation of the densities. Indices p and q correspond to coordinates in the image, where the window W is centered.

The choice of the sampling frequency will have an effect on the accuracy of the segmentation into text/picture areas. Even though these parameters are set by the user, we found a reasonable setting for a varied set of images to be as follows: h = v = 15 and a window size of n = m =25.

2.2 Density histograms

Once picture and text area α-crossing densities have been found, we compute the corresponding density histograms as defined by the discrete function (4). The purpose of the histograms, as explained below, is to select a suitable threshold to determine which areas contain pictures or text.

$$h(b_k) = s_k/t \qquad (4)$$

Where b_k = kth bin of the histogram
s_k = number of samples with density k
$t = (n(m-1) + m(n-1))$, total number of samples taken
$k=0,\ldots,t$

Note that the maximum number of α-crossings in a window of size n x m is n(m-1)+m(n-1). An examination of the histogram plots for picture and text areas (figure 5) shows the differences in density that appear. It is clear that text areas rarely show densities above 50%.

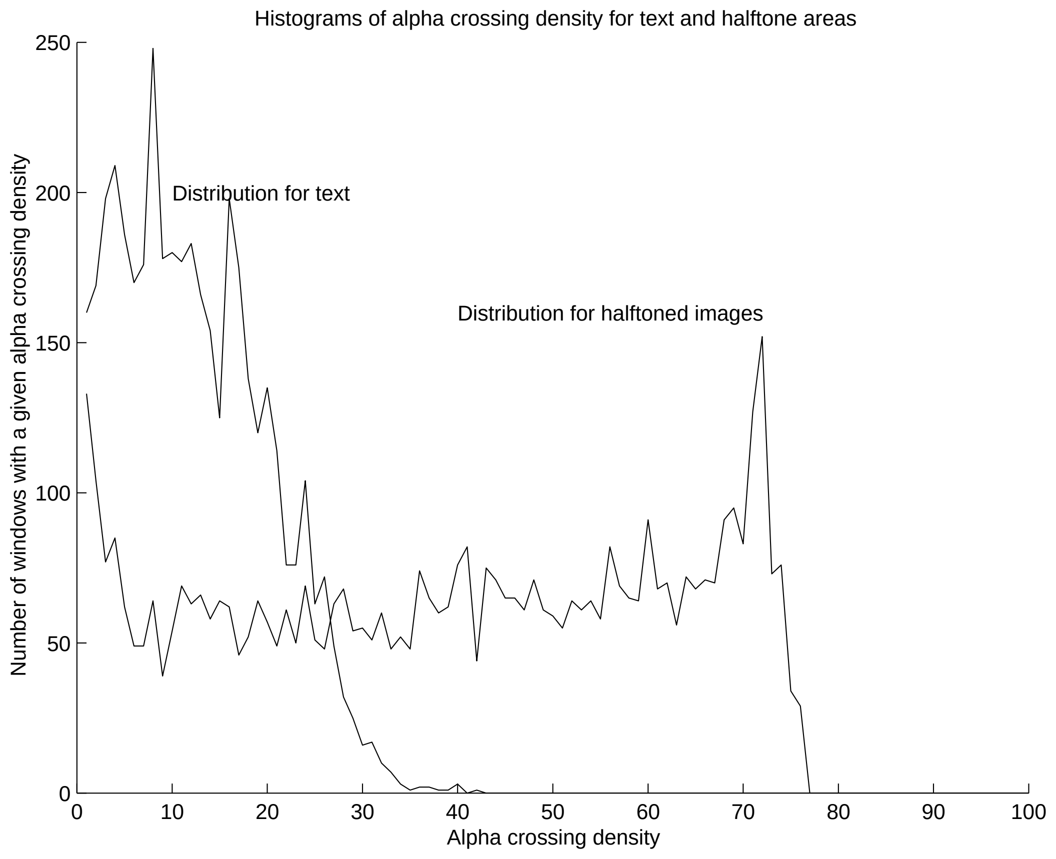


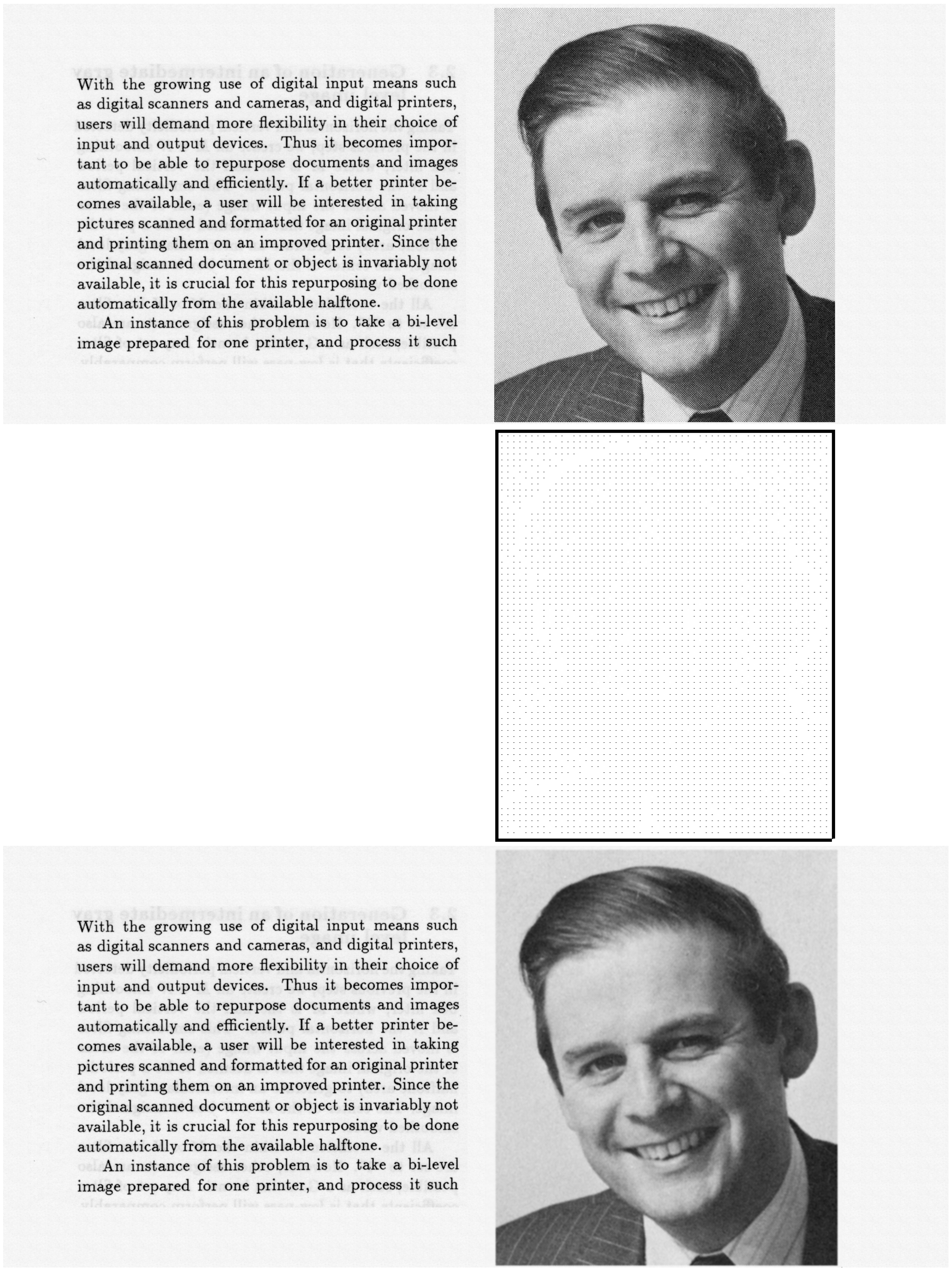
Figure 5: Histogram density plots for picture and text areas.

In order to label the set of pixels covered by the window, we choose to examine the value of $Z_d(k,l,\alpha)$ and make a decision according to the following:

```
If ($Z_d(k,l,\alpha)$ ≥ DENSITY_THRESHOLD)
      Area label = 1 (picture)                                (5)
Else
      Area label = 0 (other)
```

The window is used as described above and the density is computed in each case. The previous rule is then applied, to label those areas in the image and obtain a density map, which contains the labels for the corresponding pixel areas. An example of such density map can be seen in figure 6.

As mentioned above, text areas (including background) rarely have density values above 50%. We found this to be a good threshold value, but it can be set by the user.



With the growing use of digital input means such as digital scanners and cameras, and digital printers, users will demand more flexibility in their choice of input and output devices. Thus it becomes important to be able to repurpose documents and images automatically and efficiently. If a better printer becomes available, a user will be interested in taking pictures scanned and formatted for an original printer and printing them on an improved printer. Since the original scanned document or object is invariably not available, it is crucial for this repurposing to be done automatically from the available halftone.

An instance of this problem is to take a bi-level image prepared for one printer, and process it such

With the growing use of digital input means such as digital scanners and cameras, and digital printers, users will demand more flexibility in their choice of input and output devices. Thus it becomes important to be able to repurpose documents and images automatically and efficiently. If a better printer becomes available, a user will be interested in taking pictures scanned and formatted for an original printer and printing them on an improved printer. Since the original scanned document or object is invariably not available, it is crucial for this repurposing to be done automatically from the available halftone.

An instance of this problem is to take a bi-level image prepared for one printer, and process it such

Figure 6: Original image (top), density map with bounding boxes (middle) and result of descreening (bottom). Each dot in the density map represents high *α-crossing* density. In this particular example, no high density areas were found outside the picture.

2.3 Clustering

Once a density map is obtained, we perform clustering to group points that belong to picture areas. We make the assumption that picture areas have rectangular shape, which is usually the case.

We use the union-find algorithm [Sedgewick], which joins points in a cluster using a pre-defined clustering criterion. The criterion we use corresponds to a threshold on the city-block distance between points, that is:

Given two points a and b:

If $D(a, b) < K$ (6)
a, b belong to the same cluster

Where (i, j), (m, n) are the x and y coordinates of the points a and b respectively.
$D(a, b) = |i-m| + |j-n|$ is the distance between a and b
K is a constant.

Each cluster is then assigned a bounding box, defined to be the smallest rectangle that contains all the points in the cluster. The first round of clustering results in bounding boxes that overlap. Further steps merge overlapping bounding boxes to yield the final set of bounding boxes for halftone areas, meaning that all pixels from the original image that fall within bounding boxes belong to picture areas. Similarly, those that are not contained in the bounding boxes correspond to text and/or background areas of the document. At this stage, all pixels in the image will be labeled, as defined by the bounding boxes. The next step corresponds to descreening.

3. DESCREENING

3.1 Detection of periodicity

We make a distinction between two types of halftones: those that contain screens, such as a 45° screen used in printing grayscale images and those that do not contain screens, such as those created by error diffusion techniques. Re-halftoning of screened halftones is more problematic since moiré patterns tend to appear more often if the frequency of the re-halftoning process does not match the original screen frequency. We concentrate on those.

In order to effectively remove the screen present in the picture, first it is necessary to detect the frequency of the existing screen.

3.2 *Fast Accumulator Function*

To detect the frequency in halftone areas, we apply a *Fast Accumulator Function (FAC)* over the corresponding bounding boxes. The *FAC* uses the pixel intensity values in those areas to detect the frequency of the halftone screen. Since horizontal and vertical screen frequencies may differ, the *FAC* is applied separately in the horizontal and vertical directions.

The *FAC* consists of a 1-dimensional array which is initialized with zeros and used to accumulate pixel intensity values across rows or columns to compute horizontal or vertical frequencies. The length G of the array is equal to the size of the bounding box in the corresponding direction (i.e. $G=B_m-1$ in the horizontal direction, where B_m is the width of the bounding box area for the which we want to detect the horizontal frequency). Figure 7 shows how the *FAC* accumulates values in the horizontal direction- every row is shifted and its values added to the *accumulator array*.

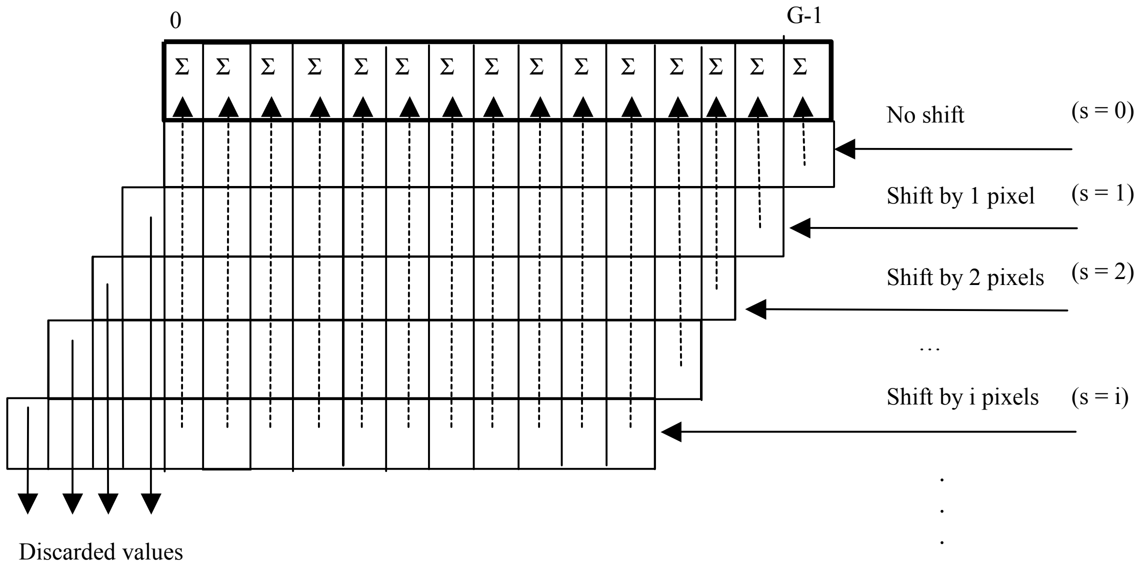


Figure 7: *Fast Accumulator Function.* Every row (or column) is shifted by s pixels and its values added to the *accumulator array.*

The process in the horizontal direction can be summarized with the following algorithm:

```
Accumulator_array = 0; // Set all elements of the array to zero
For (s = 0; s < picture_height; s++){
        new_row = expand (row_s, s); // copy current_row to new_row and expand right side of new_row by s, filling those additional entries with zeros
        shifted_row = discard (new_row, s); // Discard first s elements of the current row
        Accumulator + = shifted_row; // add each entry of the row to the corresponding entry in the accumulator (using the same index)
]
```

In summary, to find the horizontal periodicity, each row from the picture is shifted by an amount s and the pixel intensity values in that row are added to the accumulator. It is important to note that the shifting can be done from right to left or left to right (as long as the same direction is used for the entire operation).

In Figure 8, we show 3 cases that may occur when applying the *FAC* in the horizontal direction to a bounding box area and how they depend on the values of B_m and B_n (width and height of the bounding box).

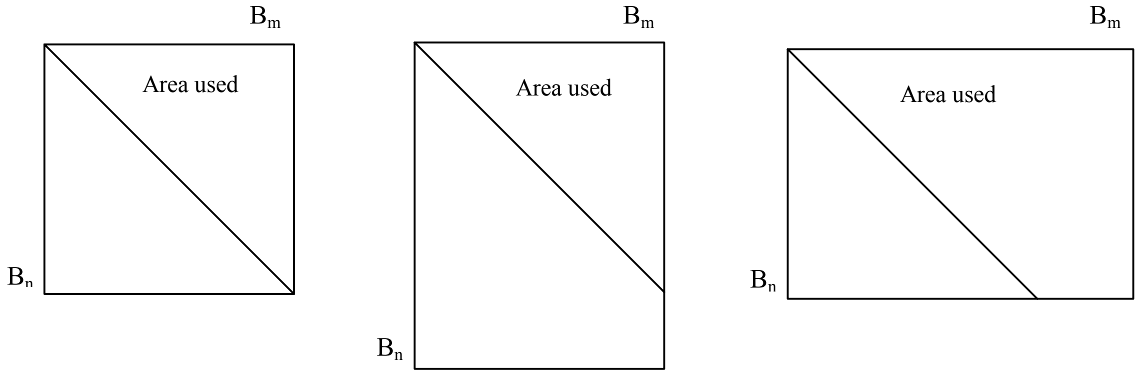


Figure 8: Illustration indicates sections of a bounding box B used by the *FAC*. Width and height of the bounding box is given by B_m and B_n, respectively. Notice that the "diagonal" may go from left to right or right to left. Likewise, it is possible to use values on the right or left of the diagonal.

Our experiments show that the variations depicted in figure 8 do not affect the effectiveness of the process- since a significant area of the picture is covered in any of the cases, it is enough to effectively detect the periodicity.

Periodicity in the vertical direction is obtained in a similar way by using columns instead of rows. The angle of the diagonal (incremental amount by which shifting is done) has been taken to be 45° since most screens in printed material use this angle. Implementing an accumulator for a different angle would imply shifting every B_n th row by B_m pixels, for an angle $\theta = \tan^{-1}(B_n / B_m)$.

The *accumulator array* serves as a one-dimensional representation of the frequency content of the picture, as can be seen in figure 9.

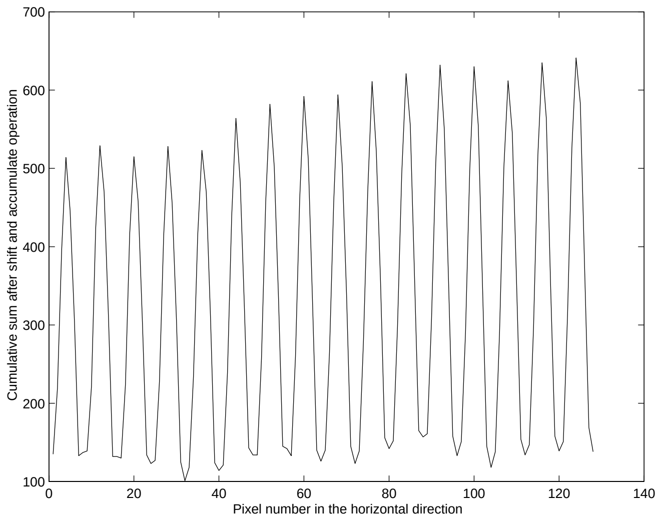


Figure 9: *Accumulator array* resulting from the application of the *FAC* to the halftone image of figure 6.

3.3 First order differencing and spectral analysis

We can denote the accumulated sum (*accumulator array*) resulting from the *Fast Accumulator Function* by a sequence *S*. In the language of time series analysis [Vandaele], *S* typically contains trends which may mask the periodicity that we wish to isolate. We perform a first order differencing operation to remove the trends. This results in a new sequence S_2 where the ith element of S_2 is

$$S_2(i) = S(i + 1) - S(i) \tag{7}$$

We perform a difference over the original sequence (the sequence resulting from applying the *Fast Accumulator Function* in the horizontal and vertical directions respectively). In order to suppress the noise amplification characteristics of the differentiation operator, we first smooth *S* by a Gaussian filter with a small kernel size such as 5. The chosen Gaussian coefficients are {1, 64, 256, 64, 1}.

Next step is to obtain the Fourier Transform of this one-dimensional signal by applying the Fast Fourier Transform (FFT) algorithm as described in [Press et al],

$$S_{2n} = \sum_{K=0}^{K=N-1} s_{2k} e^{2\pi ikn/N} \tag{8}$$

Where S_{2n} is the nth element of the FFT, N is the length.

The power spectrum P is defined to be a sequence of length N where the kth element is

$$P_k = S_{2k} \times S^*_{2k} \tag{9}$$

When a strong frequency component is present in the picture (as is the case when a screen is present), an analysis of P clearly provides a correct measure of what that frequency is. Finding the location of the maximum value of P gives the value of highest frequency present in the halftone area (figure 10). Occasionally, more than one peak may be found in P, but in most cases the ratio between the highest peak and the second highest peak has been found to be greater than 5.

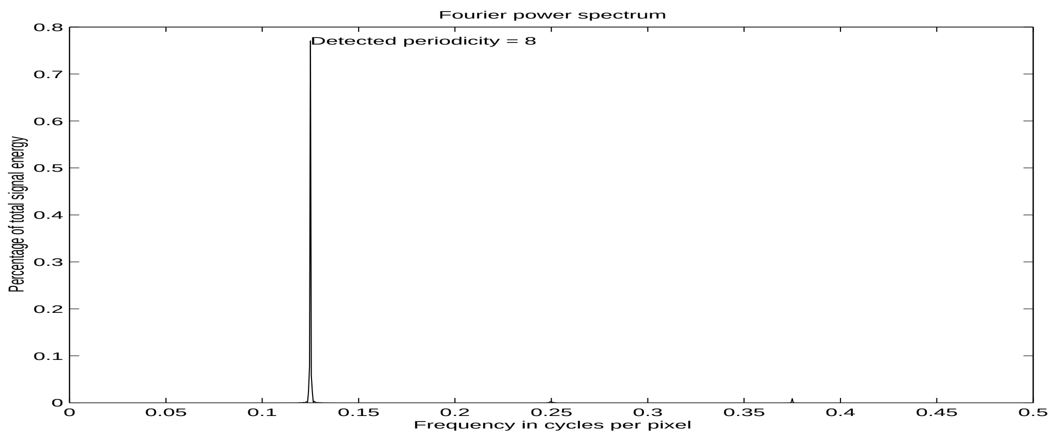


Figure 10: Fourier Power Spectrum for the *accumulator array* depicted in figure 9. In this case, it can be easily seen that there is only one dominant frequency.

3.4 Descreening

Once the screen frequency in the horizontal and vertical directions has been detected by applying the process described above, we are in condition to apply a filter to suppress the screen found in the picture. Given that the screen usually contains high frequencies, a low-pass (smoothing) filter is selected. We use a box filter algorithm for the descreening process with a separable implementation (a 1-dimensional operation is performed in the horizontal and vertical directions).

In the ideal situation, we would be able to recover the original continuous tone image from the halftoned image. Take, for example the illustration in figure 11.

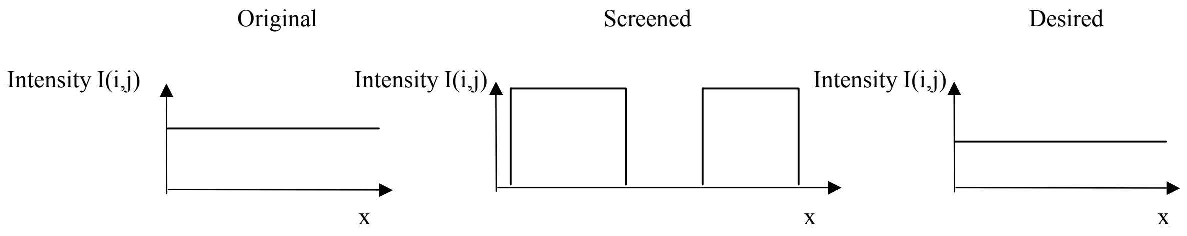


Figure 11. In constant gray level areas, the screen can be characterized by the function on the left, in which case the desired filtering result is as shown on the right.

It is clear from this example that a low-pass filter would produce the required effect. One of the drawbacks of this type of filter, however, is the fact that other high frequency areas will also be affected (blurring edges and high-texture frequency areas). We have found that such effects are minimal and the benefits of using a box filter in terms of removal of the screen and also in terms of computational efficiency are much greater than those presented by other more complex filters.

4. PERFORMANCE

We applied the approach described (*segmentation, frequency detection* and *halftone-screen suppression*) to a set of 20 document images of different sizes from diverse sources, including some local newspapers and out of print books. These included images in which no halftone areas were present and images that contained more than one halftone area. Screen frequencies varied between 5 and 9 and in all cases, if more than one halftone area was present, the screen frequency used during off-set printing was the same throughout the publication (i.e. an edition of a book or newspaper usually contains only one screen pattern) .

4.1 Segmentation

Performance of segmentation of the document into picture and non-picture areas varies according to the parameters used. Experimentally, we found that a good combination is as follows: window size n = m = 25; sampling frequency h = v = 15; density threshold = 50%.

These parameters worked successfully in 16 of the images without further adjustment. In the remaining four, two types of errors occurred in the segmentation: (1) picture areas missed (not surrounded by bounding boxes); (2) non-picture areas included inside bounding boxes.

In all cases where a halftone was present, even if there were errors of type (1) or (2), the algorithm found high *α-crossing* density clusters (bounding boxes). This means no document images that contained halftones were rejected for descreening. This is important, since once a halftone is detected adjustment of the parameters and/or segmentation is easily done by the operator.

Most of the errors in segmentation occurred in complex documents and results suggest more sophisticated clustering techniques could provide improvement.

4.2 Frequency detection

Our algorithm was successful in detecting the correct frequencies in all cases. Adding random noise (upto 60%) to the picture areas did not prevent detection of the frequencies (addition of noise was only tested in this stage and not in segmentation).

4.2 Screen suppression

Since our algorithm for detecting screen frequencies was successful in the cases tested, the filter sizes used were the correct ones. This produced visually pleasing results in terms of removal of the screen (see figure 6). We tested more complex filters but found the best (subjective) results to be given by the box filter.

In the books on-demand scenario, the errors we encountered are not of great importance because in most cases all pages of a single book are printed using the same “style”- they have similar typefaces, backgrounds and most importantly screen frequencies. If the default parameters (as above) need adjustment, this adjustment usually has to be done only once for the entire book, newspaper, or publication to be scanned.

5. CONCLUSIONS

We have presented efficient and robust techniques for the following: (1) segmentation of grayscale document images into halftone (picture) and other areas (text/background), and (2) the detection and removal of halftone screens in picture areas. The use of these techniques together with operator review provide an acceptable solution to the image quality problems encountered in applications such as on-demand printing.

On-going work includes testing our algorithms on a larger set of images and providing more quantitative results at each level. Extensions include improvements to the clustering algorithm (to allow more complex halftone image shapes) and automatic determination of some of the parameters.

REFERENCES

1. D. H. Ballard and C. M. Brown, Computer Vision]. Prentice-Hall, 1982.
2. A. Jain and B. Yu. "Document Representation and Its Application to Page Decomposition". IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 20 No. 3, March 1998.
3. H. R. Kang, Color Technology for electronic imaging devices. Bellingham, Washington, USA: SPIE, 1997.
4. Yang J. C.-Y. and W.H. Tsai, "Suppression of moire patterns in scanned halftone images by double scans with grid movements." Pattern Recognition Letters 18 (1997) 213-227.
5. X. Liu, "Analysis and reduction of moire patterns in scanned halftone pictures," PhD Dissertation, Viriginia Polytechnic Institute, Blacksburg, Viriginia, USA, May 1996.
6. T. Pavlidis and J. Zhou, "Page segmentation and classifiaction," CVGIP- Graphical models and image processing, Vol 54, 1992, pp. 484-496.
7. W. Press, B. P. Flannery, S. A. Teukolsky, and W. T. Vetterling, Numerical Recipes in C: The Art of Scientific Computing]. Cambridge University Press, 1988.
8. R.Sedgewick, Algorithms. Addison Wesley, 1983. London.
9. A. R. Rao, F. Mintzer and G. Thompson, "Descreening printer-ready images," IS&T International Conference on Digital Printing Technologies, Toronto Canada, pp. 285-289, October 1998.
10. Shu Shou-Pyng J, Springer R. and Yeh C.L. "Moire factors and visibility in scanned and printed halftone images". Optical Engineering, Vol. 28 No. 7, July1989.
11. L. G. Tatum, ``Control charts for the detection of a periodic component,'' Technometrics], no. 2, pp. 152--60, May 1996.
12. R. Ulichney. Digital Halftoning. MIT press, Cambridge, Massachusetts, 1987.
13. J. Vandaele, Applied time series and Box-Jenkins models. New York: Academic Press, 1983.
14. P. Wong, ``Inverse halftoning and kernel estimation for error diffusion,'' IEEE Trans. Image Processing], no. 4, pp. 486--498, 1995.
15. Z. Xiong, M. Orchard, and K. Ramachandran, ``A wavelet-based approach to inverse halftoning,'' in SPIE Vol. 3018].