Adaptive and Outline-Based Subsampling of Images Containing Text and Binary Graphics

Lawrence O’Gorman
John D. Hobby

Lucent Technologies Inc.

ABSTRACT

For digital display of scanned documents containing text and other binary graphics including line drawings, there is often the need to reduce the image size so that it can be more completely viewed on a computer monitor. We present an approach entailing two complementary methods: adaptive subsampling and outline based subsampling.

The adaptively subsampling method reduces image size by preferentially removing ‘‘low-information’’ or ‘‘silent’’ rows and columns. These low-information regions can be located in margins, between text lines, and even through lines of graphics and text. The outline-based subsampling method entails boundary determination, polygonal fitting, smoothing of this result, and finally reduction in size of these polygonal boundaries (rather than pixels as in the conventional approach). Together, or separately, these methods yield a higher quality, subsampled image. Results of the approach are shown. The limits in terms of image reduction versus readability are discussed.

1. Introduction

There are many applications where it is desired to display text images on a monitor. Examples include scanned pages in an electronic library [1,2] or displayed on the World Wide Web [3,4]. However, when the size in pixels of the display is smaller than that of the image, the image size must be reduced. The simplest way to do this is by deleting rows and columns at a regular interval throughout the image, that is, by subsampling. In the digital signal processing field, there are standard procedures for subsampling (also called decimation and down-sampling). These require low-pass filtering first to reduce aliasing, followed by deletion of rows and columns. For text images, outline-based subsampling techniques and other ideas explained below produce higher quality results. The criterion for quality in this case is readability.

A standard image resolution for today’s scanners is 300 dpi (dots per inch). For an 8.5x11 inch document page, the scanned size is then 2550x3300 pixels. A common monitor resolution for a PC is 1024x768 (Super VGA). It is evident from these numbers that a full page cannot be displayed completely on this monitor. To fit the image on the monitor, it must be subsampled to the smaller ratio of dimensions, that is to $768/3300$, or 23%. It is clear that a reduction of this size will reduce the quality of the image, and it is this quality reduction we wish to minimize. Even when higher resolution monitors are available to display the full page at its original resolution (300dpi, full-page monitors are available now, but at a much higher price than lower resolution monitors), there will always be the desire to display multiple pages on the same monitor or precisely drawn schematics scanned at 1000 dpi and higher. The original image can be very large also. For instance, the standard sizes for engineering drawings range up to 28x40 inches, which at 300 dpi is 8400x12000 pixels. We claim no magic to fit a highly detailed image of this large size onto a monitor, but our approach enables the display of larger portions with greater readability.

The field of digital signal processing provides us with rules and methods for sampling and subsampling while retaining signal quality [5]. The basic rule is the Nyquist criterion, which states that a signal must be sampled at, or above, twice its highest frequency to maintain fidelity. If a signal is sampled at a rate lower than this, energy from the higher frequencies wraps into the lower frequencies of the sampled signal, causing aliasing. The standard way to reduce this aliasing is to apply a low-pass filter to reduce the power in the signal above the new subsample frequency. However, for text images where the page can be thought of as a collection of black blobs on a white background, a more effective way is to find an outline description and apply a smoothing algorithm to the outlines. The goal of the smoothing algorithm should be to determine the most likely underlying shape that could lead to the observed input. See [6,7] for details. Once the image is in outline form, it can easily be rescaled or rotated. A scan-conversion algorithm can convert the outlines to pixels at the desired resolution.

In this correspondence, we describe a method that combines the outline method with another technique, adaptive subsampling, for more effective subsampling. The objective of adaptive subsampling is to sample at a higher rate within ‘‘high information’’ regions, and a lower rate outside these regions. In other words, we try to improve the readability by first removing ‘‘low information’’ or ‘‘silent’’ rows and columns of the image so that the subsequent (outline-based) subsampling rate need not be as severe. If, as we will describe, the initial step of adaptive subsampling does not affect the image text quality (or affects it minimally), then the resulting image should be more readable than it would be without using this technique. For example, if the objective is to reduce an image to 50% of the original, and if adaptive subsampling reduces the size to 90% (which is approximately typical), then the outline-based subsampling need only reduce the image to $50/90 x 100\% = 55.6\%$

We refer to the low information image regions also as ‘‘silent’’ regions because there is a fairly common, analogous procedure used for audio signals. A feature in some dictating machines and voice-mail systems, enables a recorded message to be replayed at a higher speed than the original — without a corresponding increase of audio pitch (commonly described as a Donald Duck voice). This is done by reducing silent portions of the audio between spoken words. Our method is similar except that it is low-information regions of an image that are reduced.

The adaptive method has been used for two electronic library and electronic publishing projects. This was originally designed for the RightPages Service, an electronic library that delivers scanned copies of journal articles to users [1,2]. It has been in use at AT&T and Bell Labs since 1992, and at outside locations including the University of California at San Francisco. The method has also been used for the September (Secure Electronic Publishing Trial) project, in which the October issue of the IEEE Journal of Special Areas of Communications was published on the World Wide Web [3]. In particular for this project, the method was required for reduction of scanned copies of diagrams to be viewed on the computer monitor. The outline-based method has recently been added to the RightPages preprocessing suite of software.

2. Subsampling Approach

We describe in this section the subsampling approach consisting of the two complementary methods. The adaptive subsampling method is first described, then the outline-based subsampling method. Finally, we describe how both methods are used together for the common document preprocessing task of skew correction and subsampling.

2.1 Adaptive Subsampling Method

We first describe a few conventions pertaining to this method that will be used in the remainder of this paper. We describe adaptive subsampling in terms only of ‘‘rows’’ for brevity, where ‘‘rows and columns’’ should be understood. We describe binary pixel values as ON and OFF, corresponding to values of 1 and 0 respectively. When describing the subsampling operation, we refer to rows being *retained* or *deleted*.

The objective of this method is to reduce the image size from $N_W \times N_H$ pixels to a desired fraction f of those dimensions, by eliminating ‘‘low information’’ rows. The information measure of a row, I_{RC}, is calculated as the sum of information measures of the individual pixels of that row, I_p. Rows are marked as potentially removable if their information measures are below an empirically determined threshold, T_I. Then a portion of these rows are removed, up to the chosen reduction limit, $1 - f$. There is an additional matter of evenly distributing the removed rows, which we will describe below.

We propose an *ad hoc* measure of information at a pixel, I_p, as follows. We examine 1×3 sized templates centered on pixels along each row (and 3×1 templates for columns). We measure to determine the information value of the center, or *core* pixel, with respect to its two *neighborhood* pixels. The different template patterns are shown in Table 1, and described here: a) an ON core isolated by OFF neighbors, b) an OFF core isolated by ON neighbors, c-d) a core on an edge, e) an ON core contained within ON neighbors, f-g) an OFF core neighboring an edge, and h) an OFF core surrounded by OFF neighbors. We assign ‘‘information weights’’ to these templates in Table 1. The absolute values of these are arbitrary, but the relative values are chosen (intuitively and empirically) to reflect the following:

- **Templates (a) and (b):** A core that affects **disconnectivity** between the two neighbors has the **highest information**. For example, ruled lines on a page are ON rows that are essential to maintain disconnectivity between text lines. Similarly, a single blank row maintains disconnectivity between text lines that would otherwise have touching characters.

- **Templates (c) and (d):** A core on an **edge** has the **next highest information**. For example, the ends of an ascender or descender on letters {b, d, h, p} help to differentiate these from other similar letters. Furthermore, removal of edge pixels on any letter tends to give an eroded appearance, which is undesirable.

- **Template (e):** An **ON core surrounded by two ON neighbors** has **less information** because it can be deleted without changing anything but scale. For instance, the middle row through an ‘‘o’’ can be deleted with the result being a slightly squished ‘‘o’’, and if the middle column of this result is also deleted, the ‘‘o’’ will have a similar shape but smaller scale.

- **Templates (f) and (g):** An **OFF core surrounded by one OFF and one ON neighbor** has **little information** in this small mask, but if it is deleted, the neighborhood ON row approaches another ON (outside the mask). An example of this would be shortening the spacing between double-spaced lines of text.

- **Template (h):** An **OFF core surrounded by OFF neighbors** has the **least information**. For example, a blank row in a margin can be deleted without any change of the text.

Using these templates, the information measure of each row is calculated as the sum of information weights of its member pixels,

$$I_{RC} = \sum_{i=0}^{N_W} I_p(i) \qquad (1)$$

Template	Information Weight, I_p	Reason
a) 010	128	very important to maintain disconnectivity
b) 101	128	same as above
c) 011	4	important to maintain an edge
d) 110	4	same as above
e) 111	2	somewhat important to maintain shape
f) 001	1	only important if adjacent row(s) deleted
g) 100	1	same as above
h) 000	0	not important

Table 1. Information weights of pixel templates. (The letters (a) to (h) correspond to explanations in the text of this paper.)

Rows are deleted based on low information measures, with the objective of removing the $I_W(r-1)/r$ rows from lowest to higher information measures.

The process is not quite as simple as just deleting the lowest information rows. This is because page format — or relative spacing — is also an important feature of the page that contributes to its readability. Therefore low information rows are deleted, but with the additional consideration that they should be spread uniformly throughout the page. The alternative is that, for example, if a page is to be reduced to $f=0.9$, that this reduction is applied only to a single, undistributed portion of the page, resulting in that portion being reduced in spacing but not the remaining part of the image. This is clearly not desirable.

Uniform reduction of the page is accomplished by the following approach. The image is examined for intervals of consecutive rows that are below an information measure threshold specified for removal. Within this interval, rows are deleted from lowest to higher information measures until the specified percentage reduction is reached. An additional consideration for row removal within these intervals is that, when many rows have equal information measures, we prefer not to just take these consecutively, but instead to distribute their removal. We do this by randomizing the beginning row of each search for the next minimum measure row in the interval. This eliminates a problem that otherwise occurs when an entire portion of consecutive rows are eliminated — which causes ‘‘stair step’’ error for diagonal lines.

Note that, because this is an adaptive process, the image is not reduced exactly to fraction, f, but to f', where $1.0 \geq f' \geq f$. That is, some reduction is achieved between the objective reduction, f and no reduction, 1.0. The actual value of f that is chosen will direct the reduction higher or lower, but the degree to which the objective is achieved is dependent upon the image, in particular the density of ON-regions in the image. For this reason, the value of f must be chosen empirically based upon the ‘‘typical’’ images in an application. This is determined via experimentation.

To summarize, the algorithm for adaptive subsampling is as follows (with default parameter values used for our electronic library applications given in parentheses):

1. Choose the desired fraction, f of the original image to be reduced by silence reduction (our default value is 30%). Choose a threshold on the information sum of the rows (and columns), T_I.

2. Calculate by equation (1) the information measures of each row (and column) as sums of information measures of the 1×3 (3×1) sized masks centered on pixels along each row (column).

3. For each row (and column):

   i. Find an interval of consecutive rows (columns) where every I_{RC} is below threshold, T_I, (our default value is 60) and that is bounded by rows (columns) above threshold.

   ii. Within this interval, remove up to $1 - f$ of the rows (columns) from lowest to higher value of I_{RC}. For rows with equal values of I_{RC}, distribute their removal evenly throughout the interval. We do this by making consecutive searches for the next minimum beginning at a different, randomly chosen starting row (column) within the interval for each search.

2.2 Outline-Based Subsampling

Outline-based subsampling can be thought of as a generalization of standard subsampling. It can be applied to the result of adaptive subsampling, or it can be used to combine subsampling with other transformations such as skew correction (see Section 2.3).

Figure 1a–b shows a portion of a binary image and a set of outlines that represent the same information. Treat the black pixels as unit squares and define a black region to be a maximal collection of contiguous squares. The outlines are just the boundaries of these squares — the black region in the figure produces two such outlines.

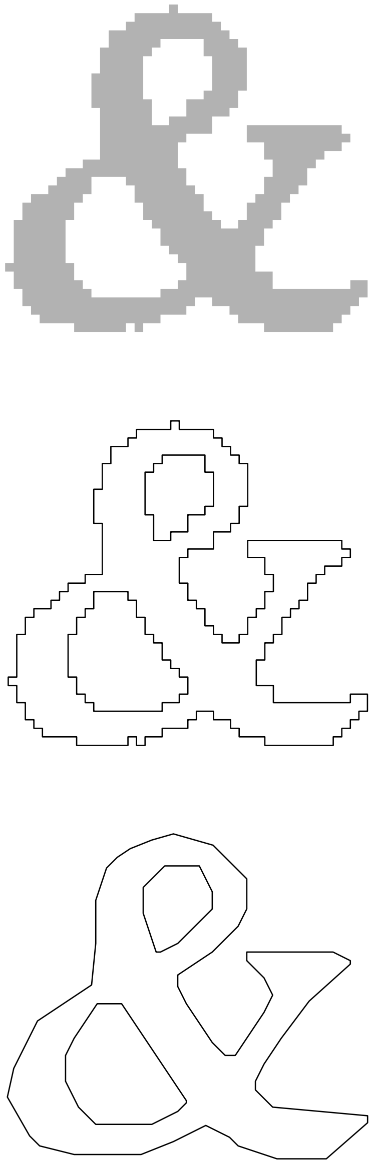


Fig.1 — example of outlines derived from a sample image. a) original image, b) unprocessed outlines, c) smoothed outlines.

The basic idea of outline-based subsampling is to find outlines as in Figure 1b, apply a polygon-smoothing algorithm to obtain ‘‘better’’ outlines as in Figure 1c, and then use a polygon scan-conversion algorithm to convert the outlines to pixels at the desired resolution. The precise definition of ‘‘better’’ is beyond the scope of this correspondence but it should involve a reduction in high frequency noise. See [7] for a discussion of the goals and [6,7] for appropriate polygon-smoothing algorithms.

The outlines of the black regions can be found via a well-known sweep-line algorithm that updates lists of outline vertices each time it considers a new row of the input image. See [8] for details.

The final step is to rescale the outlines to the desired subsample rate and scan-convert them. If the output is to be a binary image and not much reduction is needed, it usually suffices to turn on the pixels whose centers lie inside the outlines. Otherwise, it is best to produce gray scale output and then threshold it at a level sufficient to prevent thin lines from breaking up. For gray scale output, it is important to anti-alias. This is typically done by examining each pixel square and coloring the pixel according to the fraction of the square that is covered by the outlines. An equivalent formulation is to convolve with a unit square — this is analogous to the low-pass filtering that is done during standard subsampling [9]. It can be generalized to use different kernels for better low pass filter characteristics (see [10] for details).

Outline-based subsampling with anti-aliasing reduces to standard subsampling if the polygon-smoothing step is omitted. If this step is as beneficial as Figure 1b-c suggests, it is reasonable to expect improved readability. In addition, the outline representation makes it easy to use non-integer scale factors or apply additional transformations such as image rotation.

2.3 Deskewing and Subsampling

In Section 2.2, we explained that outline-based subsampling techniques readily generalize to geometric transformations such as image rotation. Suppose we want to take advantage of this to correct for a known skew angle as well as subsample. If we simply do adaptive subsampling as in Section 3, then deskew, and then down sample, the rows and columns considered during adaptive subsampling will not be aligned with the text in the image. This will not take optimal advantage of low-information row and column reduction via this method.

We could use the outline-based method to deskew and subsample, and then do adaptive subsampling, but this leaves the final image size dependent on the amount of low information space in the image. That is, this does not meet the objective of reducing to a specified size. Hence, we need a slightly more complicated procedure:

1. Deskew and scan-convert the result of the outline based method. If the skew angle is not known, it can be found by any of several methods; e.g., [11].

2. Apply adaptive subsampling to the deskewed image.

3. Take the deskewed outlines from Step 1 and modify them to reflect the omission of rows and columns in Step 2.

4. Complete the subsampling process using the outlines from Step 3; i.e., rescale the outlines and scan-convert them as explained in [10].

The purpose of Step 3 is to obtain the result of extracting smooth outlines from the image produced by Step 2, but avoid the degradation introduced by scan-conversion in Step 1. The modification is just to apply a suitable function to the x and y coordinates of the outline vertices. Suppose row i occupies y coordinates $i < y < i+1$ and $R(j)$ is the number of rows i with $i < j$ such that row i is removed by adaptive subsampling. If we use linear interpolation to define $R(y)$ for non-integer y, the mapping on y coordinates is to replace y by

$$y - R(y),$$

and the mapping for x coordinates is similarly based on the number of removed columns.

3. Results and Discussion

We illustrate the results of the method in Figures 2–5. Figure 2 shows the results of the adaptive subsampling method on a page from a technical article. In Figure 2b, black lines are drawn through rows and columns that are found to have low information. Note how these are mainly in margins and spaces between blocks of text. However, as seen in Figure 3, they are also between text lines and even within text lines — through characters, especially descenders.

Figure 4 illustrates the results of the method on a graphics image. Note that low-information rows and columns can run directly through lines of graphics. It is important that the method recognizes low-information space not just as white space, but also rows containing these graphics lines.

Another example of how it is important that adaptive subsampling deals appropriately with lines is for a table of text enclosed in a box (not shown here). One can understand that, no matter how sparse the enclosed text may be, adaptive subsampling would yield no reduction without the ability to recognize the lines of the box as contributing low information. In Table 1, the information template corresponding to a pixel on a line is (e), and the information weight is relatively small, 2.

Figure 5 illustrates the improved quality of text using the combination of adaptive subsampling and outline-based subsampling. Figure 5a shows a portion of text subsampled to $1/3$ of the original by the conventional method of filtering (3x3 uniform filter) and row-column deletion. Figure 5b shows the results of our method, where the adaptive subsampling result is 93% x 87% of the original, that is the width has been reduced to 93% and the height to 87%.

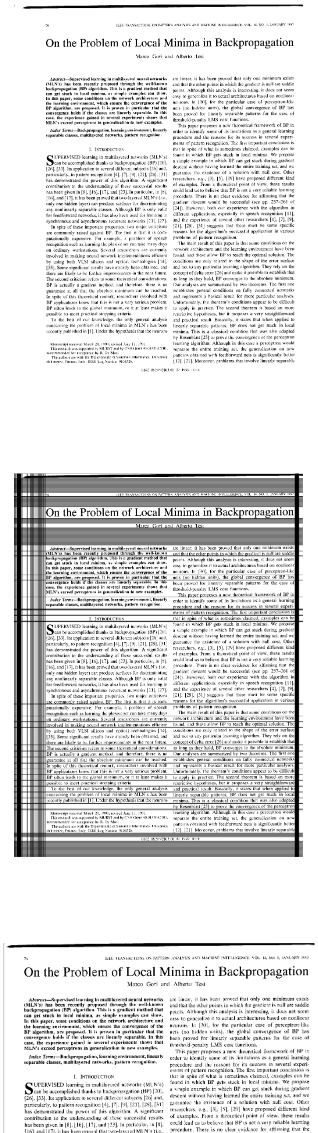


Fig. 2 — Images illustrate the adaptive subsampling method upon a page from the IEEE Trans. on PAMI. The top image is the original, the middle shows lines through low-information rows and columns, and the bottom shows the final result. The final reduction is 93x87%.

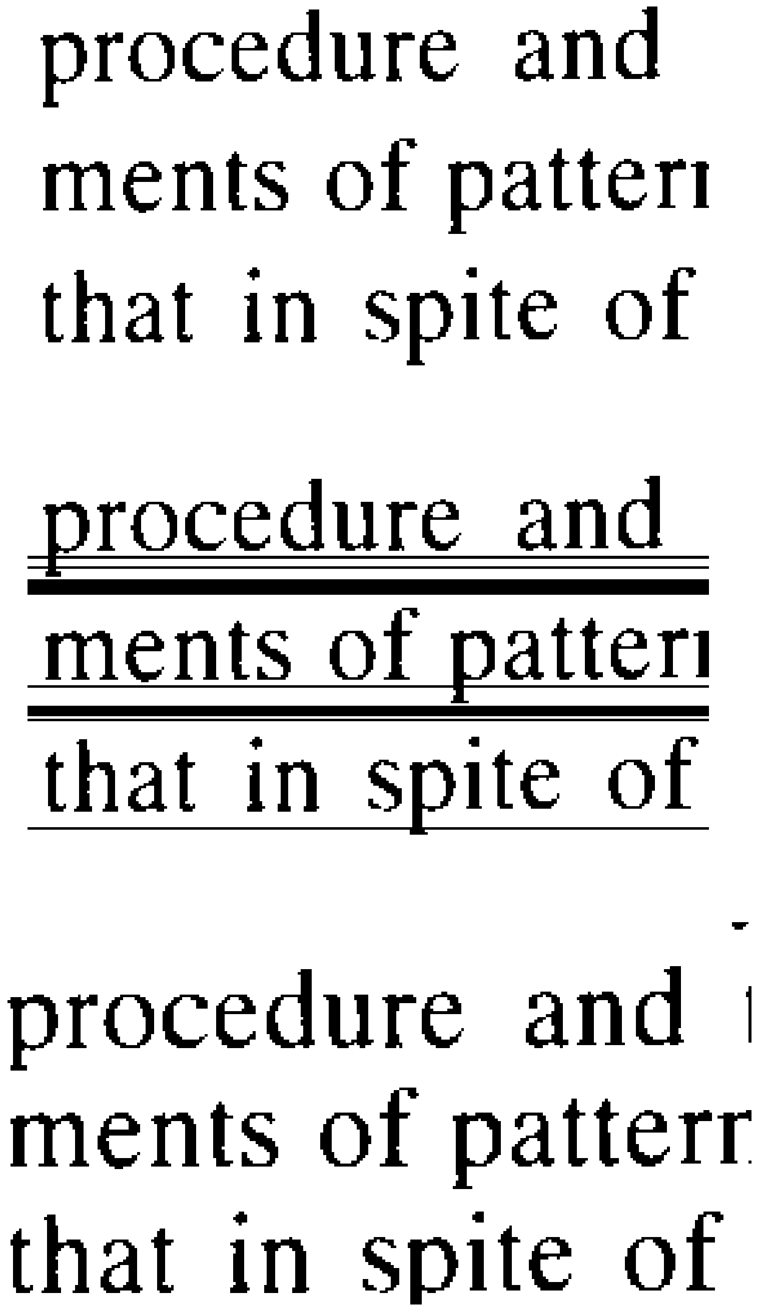
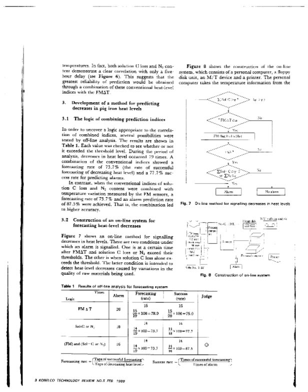
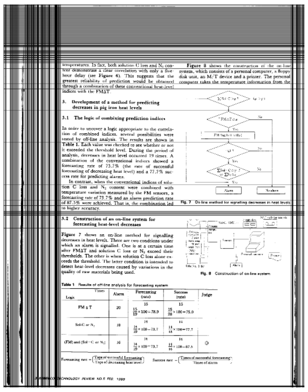


Fig. 3 — This figure illustrates a zoomed portion of the page of Figure 2 showing exactly where the low-information rows and columns occur within the text. The top image is the original, the middle shows lines through low-information rows and columns, and the bottom shows the final result.





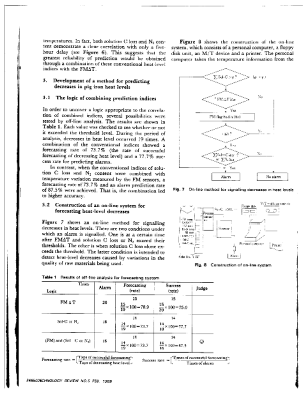


Fig. 4 — Images illustrate the adaptive subsampling method upon a page from [12] containing graphics and text. The top image is the original, the middle shows lines through low-information rows and columns, and the bottom shows the final result. This final reduction is 90x91%.

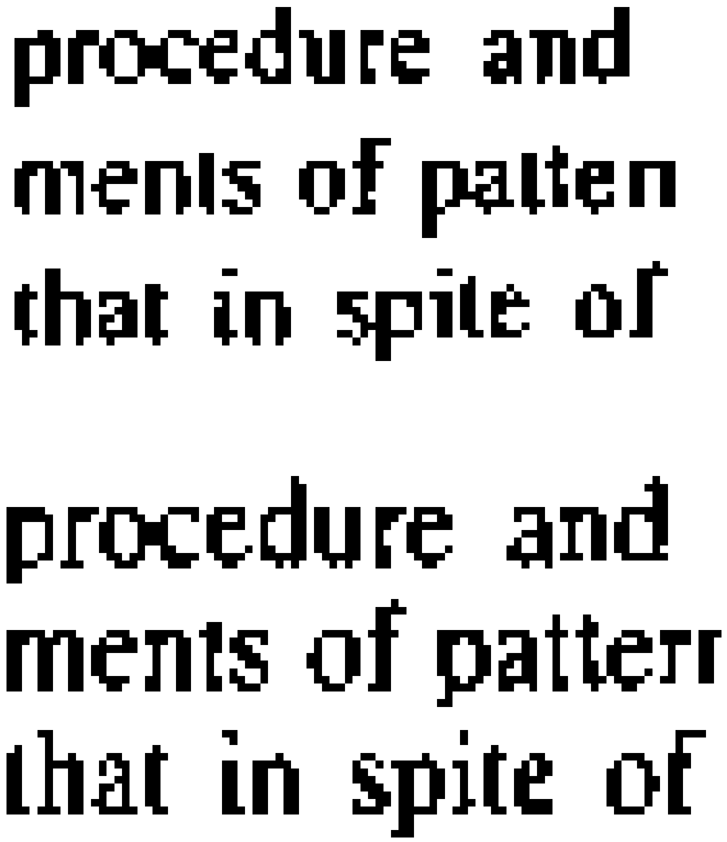


Fig. 5 — This figure illustrates the improvement in readability due to the combination of the two methods for the document preprocessing task of skew correction and subsampling. The image is a portion of the same text image used in Figures 2 and 3. Above is the image using deskewing with bilinear interpolation and subsampling with an averaging filter. Below is the result of using our methods as described in Section 2.3. The image is reduced by a factor of 3.

We claim the result of this method is greater readability. We performed a test of this with human readers in a paper describing an earlier version of the adaptive subsampling method [13]. The results indicated (as intuition also leads one to believe) that an x% reduction of white space, yielding a correspondingly larger text area, also yields a correspondingly greater readability. We have not repeated this test here due to its subjective nature, but we achieve on average about a 5-15% reduction of image size by the adaptive subsampling method for our electronic library applications, which are done in batch mode with default parameters given in Section 2.1, and whose content is primarily technical journal pages.

Table 3 shows run times for the complete algorithm on a 150 Mhz. SGI Challenge XL using just one of the 12 MIPS R4400 processors. As [7] suggests, the run time for the outline method is roughly proportional to the length of the raw outlines. Since the algorithm in [6] is faster than the one in [7], it was used for all but the largest outlines. (For large outlines, [7] gives better results.)

	Text Image	Graphics Image
outline length [pixels]	187,302	101,469
outline method [sec]	6.10	3.29
adaptive method [sec]	3.94	2.90
I/O overhead [sec]	2.54	0.99
total [sec]	12.58	7.18
avg. per outline pixel [μsec]	67	71

Table 2. Table shows the amount of computation time required for each method and for a program combining both methods as explained in Section 2.3. The times for the adaptive method include Steps 2 and 3.

The default values we use (mentioned in Section 2) have worked well for our applications. For the adaptive subsampling method, we use $f = 30\%$ and $T_I = 60$. For the outline-based method, we use $\varepsilon = 1$ for outlines whose dimensions exceed 80 pixels and $\varepsilon = 0.5$ for smaller outlines. (The ε parameter is explained in [7]). These values can be changed — perhaps other values yield better results for other applications.

We use the methods mainly in the ''batch'' mode. That is, for all scanned pages, the document preprocessing of Section 2.3 is applied to all images. using default parameters, and with no operator interaction (though images are visually checked for scanner misfeeds, etc. afterward). The methods can also be used interactively — as was done for the diagrams of Figure 4. The interaction consists of the f parameter value being changed, then the result observed to determine the tradeoff between reduction and image quality.

For graphics images, there are often a lot of white space rows and columns that can be removed, but this can change the aspect ratio of a diagram. This is fine for something like a circuit diagram where only the qualitative appearance matters, but may not be appropriate diagrams drawn to scale.

4. Summary

Outline-based subsampling and adaptive subsampling each have potential for significantly improved subsampling of binary images. The improvement is two-fold: smoother-rendered text via the outline-based method and larger text per display area via the adaptive subsampling method. Combined, these two methods yield improved readability.

We have introduced simple, attractive algorithms for both, and we have seen how to combine them. When other transformations are needed such as image rotation for skew-correction, the outline-based paradigm avoids the need to lose information by representing intermediate results in bitmap form.

Improved text and diagram subsampling is important to the application of digital libraries, where scanned documents are displayed on computer monitors with lower resolution than the original image. In fact, excellent readability of the documents is the most important aspect of these systems, and the methods described here are shown to improve this readability.

REFERENCES

1. G.A. Story, L. O'Gorman, D. Fox, L. Schaper, and H.V. Jagadish, ''The RightPages image-based electronic library for alerting and browsing'', IEEE Computer, Sept. 1992, pp. 17-26.

2. L. O'Gorman, ''Image and document processing techniques for the RightPages Electronic Library System'', Int. Conf. on Pattern Recognition (ICPR), The Hague, Sept. 1992, pp. 260-263.

3. J. Brassil, A. Choudhury, D. Kristol, A. Lapone, S. Low, N. Maxemchuk, L. O'Gorman, ''SEPTEMBER: Secure Electronic Publishing Trial'', IEEE Communications Magazine, vol. 34, no. 5, May 1996, pp. 48-55.

4. R.E. Kahn, ''An introduction to the Computer Science Technical Report Project'', http://www.cnri.reston.va.us/home/cstr.html, Dec. 1995.

5. R.E. Crochiere, L.R. Rabiner, *Multirate Digital Signal Processing*, Prentice Hall, 1983.

6. J.D. Hobby, ''Smoothing Digitized Contours'', Theoretical Foundations of Computer Graphics and CAD, Springer Verlag, 1988, pp. 777-793.

7. J.D. Hobby", ''Polygonal Approximations that Minimize the Number of Inflections'', Proc. of the Fourth Annual ACM-SIAM Symp. on Discrete Algorithms, Jan. 1993, pp. 93-102.

8. C. Ronse, P.A. Devijver, *Connected Components in Binary Images: the Detection Problem*, Research Studies Press, Letchworth, England, 1984.

9. L. O'Gorman, A.C. Sanderson, ‘‘A comparison of methods and computation for multi-resolution low- and band-pass transforms for image processing’’, Computer Vision, Graphics, and Image Processing, Vol. 37, pp. 386-401, 1987.

10. T. Duff, ‘‘Polygon Scan Conversion by Exact Convolution’’, Raster Imaging and Digital Typography, 1989, pp. 154-168.

11. H. S. Baird, ‘‘The Skew Angle of Printed Documents,’’ in L. O'Gorman & R. Kasturi (Eds.), *Document Image Analysis*, IEEE Computer Society Press, Washington, 1995

12. I.T. Phillips, S. Chen, R. M. Raralick, ‘‘{CD-ROM} Document Database Standard’’, *Proceedings of the International Conference on Document Analysis and Recognition*, 1993, pp. 478--483.

13. L. O'Gorman, G.A. Story, ‘‘Subsampling text images’’, 1st Int. Conf. on Document Analysis and Recognition, St. Malo, France, Sept. 1991, pp. 219-227.