ELSEVIER

Pattern Recognition Letters 21 (2000) 311–317

Pattern Recognition Letters

www.elsevier.nl/locate/patrec

Algorithm for text page up/down orientation determination

Robert S. Caprari *

Department of Defence, Defence Science and Technology Organisation, P.O. Box 1500, Salisbury, SA 5108, Australia

Received 20 January 1999; received in revised form 2 November 1999

Abstract

We present an algorithm that operates on a bit-mapped text pattern array to determine the up/down orientation of the page, that is, whether the page is upright or inverted. The algorithm exploits an up/down asymmetry of passages of text composed of roman letters and arabic numerals. A computer program that implements the algorithm is listed. Experimental results from electronically received facsimile (fax) pages verify the effectiveness of the algorithm. © 2000 Elsevier Science B.V. All rights reserved.

Keywords: Facsimile document processing; Text image orientation; Upright/inverted text discrimination

1. Introduction

Automated document analysis systems are being developed for a variety of applications. Every capability of such a system requires a mathematical algorithm to accomplish what a human operator can do instinctively and intuitively. The capability that we address here is that of determining the up/down orientation of text documents, which is essential for further analysis, such as the duplicate document detection procedure that we have recently developed (Caprari, 2000). Inverted (i.e. upside down) pages can arise from scanning paper documents into a computer. A common circumstance in which this occurs is the receipt of facsimiles (i.e. faxes) by fax modems connected to a computer. One reason is that there is often no strong indication of which way to feed the paper original through the transmitting fax machine. This ambiguity is exacerbated by the fact that the way that produces 'upright' paper faxes at the receiving end, usually produces inverted electronic fax images! The algorithm described here was specifically developed to solve this fax page up/down orientation determination problem. Effectiveness of the algorithm relies on the document being substantially text, which is quite consistent with the content of most faxes.

The algorithm operates on a page of scanned text represented as a bit-mapped pattern array. Although patterns actually may have multiple graylevels, success of the algorithm requires strong contrast between a 'white' background of graylevel ~ 0, and 'black' characters of graylevel ~ 1. That is, the text page must be effectively binary valued. Faxes are intrinsically binary valued, however White $= 1$ and Black $= 0$ in the digital images, so fax images must be complemented prior to being analysed by the algorithm. The algorithm requires that the original document be reasonably

* Tel.: 61-8-8259-5097; fax: 61-8-8259-5254.
E-mail address: robert.caprari@dsto.defence.gov.au (R.S. Caprari).

PII: S 0 1 6 7 - 8 6 5 5 (9 9) 0 0 1 6 1 - 0

well vertically aligned with respect to the scanner axes. By ‘reasonably well’ we mean that the original paper page should not have been presented to the scanner in a deliberately skewed orientation, but neither is extraordinary care and attention to paper page alignment necessary. Fax machine feeds mechanically prevent excessive skewing of the paper page, although mild skew is inevitable – such alignment quality is satisfactory for the success of the algorithm. In the absence of a mechanism to prevent excessive page skewing, the scanned document image may need to be operated upon by a crude (hence simple and fast) deskewing procedure, to approach within about 10° of exact vertical alignment. This up/down orientation determination method is designed for the roman alphabet, but serendipity may render it effective also for other alphabets, and even iconic languages.

Automated text document vertical/horizontal (i.e. portrait/landscape) orientation determination has been successfully solved by several researchers (Akiyama and Hagita, 1990; Farrow et al., 1994; Le et al., 1994) using variations of the same method. Essentially the method relies on text having frequent interline gaps of similar size to character heights, but smaller and ‘vertically’ unaligned intraline gaps. Text document up/down orientation determination has been successfully addressed by Bloomberg et al. (1995), on the basis of similar character properties to those exploited by our technique, although using a very different method to infer orientation from the significant properties. Some specialised document analysis techniques (Rafal and Ward, 1989; Hönes and Lichter, 1994; Ali, 1997) are capable of determining arbitrary text orientations, although without being able to distinguish 180° orientation disparities (which is essentially the problem that we solve here).

2. Up/down text asymmetry

The distinction between upright and inverted text exploited here is a consequence of the subtle characteristic of roman letters and arabic numerals that is illustrated in Fig. 1. The middle row contains characters that neither protrude below their own line, nor above half height level of their line. The top row contains characters that do not protrude below their own line, but have ascending protrusions that penetrate the half height level of their line. The bottom row contains characters that do not protrude above half height level of their own line, but have descending protrusions that penetrate into the line below.

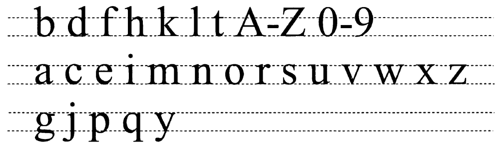

Fig. 1. Alphanumeric characters grouped according to their possesion of ascending protrusions (top row), descending protrusions (bottom row), and neither (middle row).

Up/down text asymmetry is a result of the statistical excess of ascending characters over descending characters in most sizeable pieces of text. Specifically, for lower case letters in common written English, the frequencies of occurrence of letters in the top, middle and bottom rows of Fig. 1 are 26.5%, 67.25% and 6.25%, respectively (Kahn, 1967, p. 100); a ratio that is skewed further in favour of characters in the top row when upper case letters and numerals also are taken into account. The corresponding relative frequencies for unaccented German (i.e. ‘ä’ replaced by ‘ae’, ‘ö’ by ‘oe’, and ‘ü’ by ‘ue’) are 23.7%, 71.9% and 4.4%, respectively (Bauer, 1997, p. 270), thus demonstrating the same qualitative asymmetry. For text containing a sufficiently large number of characters, the up direction is that in which there are more protrusions from the bulk of the text lines, the latter being from the line bottom to half height level. The algorithm presented here is a method of accounting for such protrusions in a direction sensitive manner, from text pages supplied as bit-mapped patterns.

3. Inference of text asymmetry

We describe the algorithm in this section, and include an implementation of it as a MATLAB computer program in Appendix A. The program is a faithful transcription of the algorithm exactly as we describe it here. As we progress through the algorithm we graph important quantities in Fig. 2. Note that Fig. 2 has the downward vertical direction of the text pattern array displayed along the right horizontal direction. In the following discussion, directions to which we refer are always with respect to a vertically oriented text page.

For a perfectly vertically aligned page, the first operation would be to compute a horizontal projection of the complete text pattern; 'horizontal projection' being defined as the integral of the pattern along individual pixel rows. However, experimentation with fax pages indicates that the mild misorientations encountered in practice frustrate the effectiveness of the algorithm. We restore the algorithm effectiveness by computing the horizontal projection across a sufficiently narrow vertical strip of the text page; sufficiently narrow for the vertical descent in the image plane of the text horizontal direction, as the width of the strip is traversed, to be much less than the magnitude of ascending and descending character protrusions. We represent the width of the vertical strip in units of pixels by the variable `range`, and for fax page patterns with 1728 columns of pixels, it transpires that $\texttt{range} = 200$ is most satisfactory for 10–12 point character font sizes (standard business correspondence sizes). Many strips fit across a page, and we identify the horizontal position of the strip by its leftmost pixel column `base`. The whole pattern itself is stored in array `page`, and Graph (a) illustrates a vertical strip of text of height 200 pixels. 'Standard resolution' fax pixels are twice as high as they are wide, so the Graph (a) aspect ratio is not quite equal to that of the original document image. The horizontal projection across the page strip is represented by `p` (*projection*), and is computed as

$$\texttt{p}(i) = \sum_{j=\texttt{base}}^{\texttt{base+range}-1} \texttt{page}(i,j) \quad \forall\, i \tag{1}$$

for row index i ranging over all of the rows of the text pattern array. `p` is shown in Graph (b) correctly aligned with (a).

Inspection of the graph of `p` reveals two indicators of text up/down asymmetry. One is the base structure of the peaks, and the other the plateau structure of peaks. At the base, the bottom spike is consistently smaller than the top spike. On the plateau, the bottom spike is quite consistently larger than the top spike. However, this behaviour is restricted to text composed of characters with serifs, as in roman fonts. Experimentation showed that for sanserif characters the plateau structure is reversed, although the base structure remains qualitatively the same. Accordingly, for the algorithm to be effective for both groups of character typefaces, it should only respond to the base structure of horizontal projection peaks. We accomplish this by thresholding the projection peaks at around the plateau level, thus producing `tp` (*thresholded projection*) as

$$\texttt{tp}(i) = \min(\texttt{p}(i),\ \texttt{thresh} \times \max_k \texttt{p}(k)) \quad \forall\, i, \tag{2}$$

where $\texttt{thresh} = 0.4$ proved to be a suitable value for the threshold relative to the global maximum of the projection function. `tp` is displayed in Graph (c). For roman typeface only, the thresholding operation mildly compromises performance of the algorithm, because the excised plateau structure would have reinforced the retained base structure. If there is certainty that the typeface is roman, it is best to dispense with the thresholding operation, and to proceed with `p` in place of `tp`.

Next we differentiate the thresholded projection by a finite difference method. Using the forward difference formula we compute `dtp` (*differentiated thresholded projection*) as

$$\texttt{dtp}(i) = \texttt{tp}(i+1) - \texttt{tp}(i) \quad \forall\, i, \tag{3}$$

which is shown in Graph (d). Notice from the graph of `dtp` that each text line is characterised by either one or two large positive spikes at the top, and one typically even larger negative spike at the bottom, as well as several smaller spikes. This verifies that the text up/down asymmetry has persisted through to this stage of the algorithm.

The integral of `dtp` over all rows is the difference between values of `tp` for the bottom and top rows of the text page, which is not an indicator of up/down orientation. To make this strategy

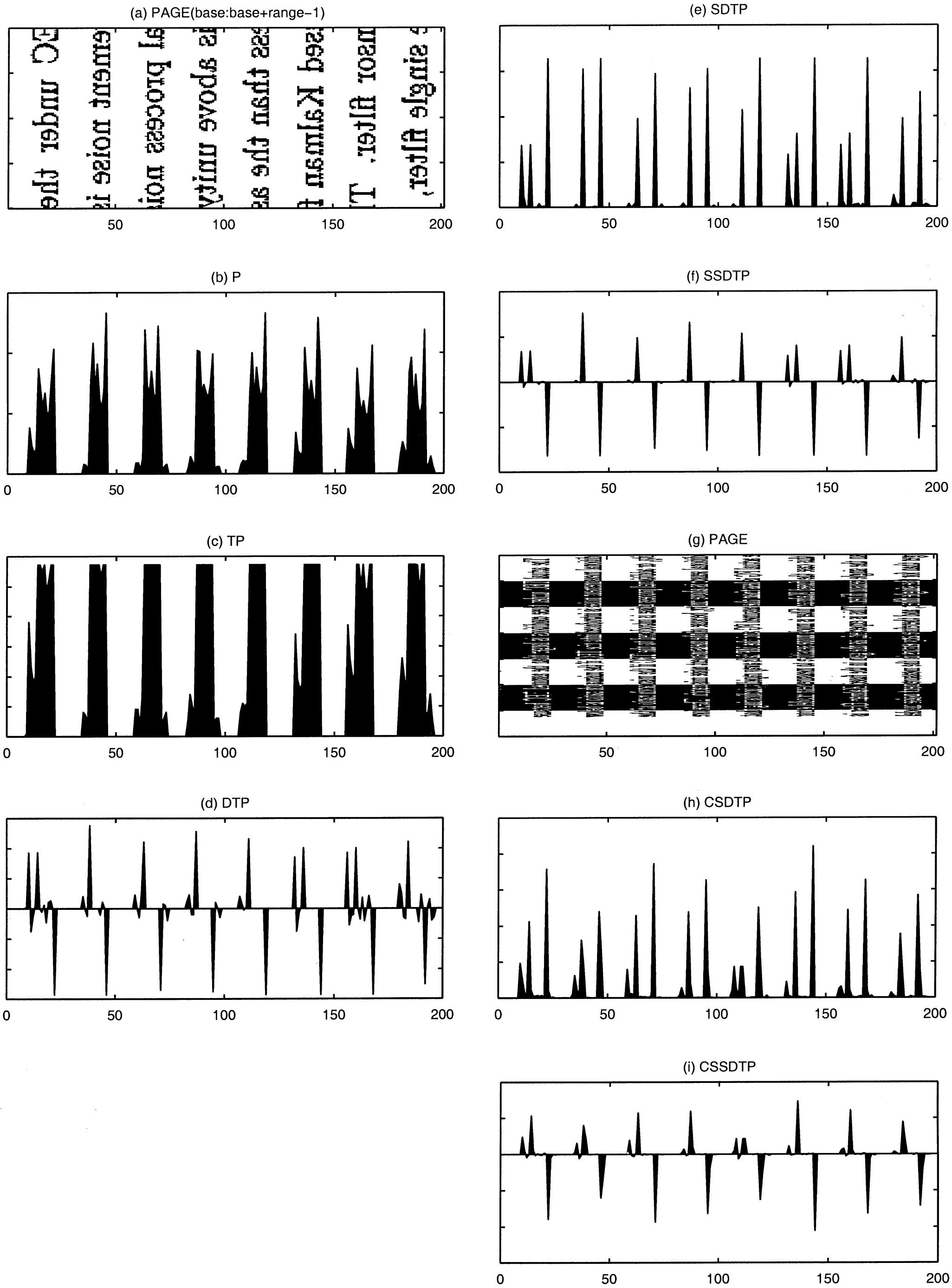

Fig. 2. Graphs of significant functions that arise during execution of the algorithm.

effective, we nonlinearly transform dtp in a manner that accentuates the height differences of spikes; a square law point transformation is suitable. Consequently, we compute sdtp (*squared differentiated thresholded projection*) as

$$\mathtt{sdtp}(i) = \mathtt{dtp}^2(i) \quad \forall\, i, \tag{4}$$

which is illustrated in Graph (e). Since the squaring operation destroys the important sign information that is present in dtp, we restore this information by imposing the sign structure of dtp onto sdtp, thereby forming ssdtp (*signed squared differentiated thresholded projection*) as

$$\mathtt{ssdtp}(i) = \mathrm{sign}(\mathtt{dtp}(i)) \times \mathtt{sdtp}(i) \quad \forall\, i, \tag{5}$$

shown in Graph (f). It is probably satisfactory to replace the squaring followed by sign restoration nonlinear operation by a simple cubing operation.

We now have reached the stage where the results for the present vertical strip of the text page can be combined with corresponding results for other such strips. In the case of fax pages with range = 200, about seven vertical strips fit across a page width of 1728 pixel columns, leaving aside blank margins on either side. Graph (g) illustrates the coverage of the fax page by the seven adjacent strips used in our implementation of the algorithm (the reverse contrast of adjacent strips is for dramatic effect only – the original page does not look like this!). The previous steps are repeated individually for all vertical strips, and the functions sdtp and ssdtp are accumulated across all strips to form csdtp (*cumulative squared differentiated thresholded projection*) and cssdtp (*cumulative signed squared differentiated thresholded projection*), respectively:

$$\mathtt{csdtp}(i) = \sum_{n_{\mathrm{strip}}} \mathtt{sdtp}(n_{\mathrm{strip}}; i) \quad \forall\, i, \tag{6}$$

$$\mathtt{cssdtp}(i) = \sum_{n_{\mathrm{strip}}} \mathtt{ssdtp}(n_{\mathrm{strip}}; i) \quad \forall\, i. \tag{7}$$

Page up/down orientation is inferred from functions csdtp, shown in Graph (h), and cssdtp, shown in Graph (i), on the basis that an upright text page will have negative spikes of cssdtp dominating positive ones on average over all rows. A normalised measure of this up/down asymmetry is asym defined as

$$\mathtt{asym} = -\frac{\sum_i \mathtt{cssdtp}(i)}{\sum_i \mathtt{csdtp}(i)}, \tag{8}$$

which is indicative of an upright text page when positive, and an inverted text page when negative, as reflected in boolean variable up:

$$\mathtt{up} = \mathtt{asym} \geqslant 0. \tag{9}$$

The confidence level of decision up increases with increasing absolute value of asym.

Since the denominator of asym (8) is positive definite, the outcome of the orientation test is only determined by the numerator of asym, that is, the orientation test statistic is effectively the numerator of asym alone. The denominator of asym has the important effect of normalising the test statistic so that regardless of the latter's magnitude, a given magnitude of asym is always associated with the same level of confidence in the orientation decision. Note that the accumulation of results over vertical strips expressed by (6) and (7) can extend over several pages, with corresponding benefit to the confidence level of the final orientation decision, if there is certainty that all of the pages have the same orientation (e.g. all pages are from the same document).

4. Experimental results

The algorithm described in Section 3 was implemented by the computer program listed in Appendix A, and tested on a diverse sample of faxed (standard resolution) text pages. Testing was conducted on: 39 pages of 12 point upright roman text, both clean and degraded by 10% impulse noise (i.e. individual pixels independently complemented with probability 0.1); a different set of 38 pages of 10 point inverted roman text, both clean and noisy; and another different set of 36 pages of 10 point upright sanserif text, both clean and noisy. No deskewing of page images was undertaken. Experimental results are presented in Table 1. The single best indication of algorithm performance is the absolute value of the quotient

Table 1
Experimental results from applying the up/down orientation determination algorithm[a]

Font	Size	Up/ Down	Noise	Pages	asym min.	asym max.	asym mean	asym S.D.	|Mean÷/S.D.|
Roman	12 pt	up	0	39	0.0556	0.1379	0.0846	0.0150	5.6
Roman	12 pt	up	10%	39	0.0172	0.0732	0.0511	0.0131	3.9
Roman	10 pt	down	0	38	−0.1163	−0.0460	−0.0826	0.0171	4.8
Roman	10 pt	down	10%	38	−0.0768	−0.0041	−0.0455	0.0179	2.5
Sanserif	10 pt	up	0	36	0.0266	0.1114	0.0749	0.0205	3.7
Sanserif	10 pt	up	10%	36	0.0093	0.0669	0.0420	0.0155	2.7

[a] 'Mean' is sample mean; 'S.D.' is unbiased sample standard deviation. Note that the up/down orientation of all 226 pages is determined correctly.

of the sample mean and standard deviation of up/down asymmetry parameter `asym`, which reflects the margin of error of the algorithm.

Table 1 reveals that the correct orientation decision was reached for all 226 pages tested, even for the ones severely degraded by noise. All 113 clean pages have `asym` more than one standard deviation in the correct direction from the neutral value 0. The error margins listed in the rightmost column demonstrate that algorithm performance is excellent for 12 point roman text, which is the most common type occurring in faxes. Smaller font sizes and sanserif fonts moderately reduce algorithm effectiveness.

For a finite region of text, the uncertainty in the value of `asym` as an estimate of its value for an infinitely large region of text, is roughly inversely proportional to the square root of the area of the text region (the bigger the text region, the more accurately this rule holds). This behaviour represents a graceful degradation of performance of the algorithm with decreasing area, because for every fourfold decrease in area, the standard deviation of `asym` only doubles, that is, the margin of error only halves. Consequently, satisfactory performance of the algorithm may be obtained by operating over text regions of size much smaller than a whole page.

Acknowledgements

The assistance and support of Tim Benham, Warwick Holen, Garry Newsam, Richard Price and Jon Vaughan in the conduct of this research is gratefully acknowledged.

Appendix A. Computer program for the algorithm

```
%         UP_DOWN.M (a MATLAB script file)
%         ~~~~~~~~
% This program determines the UP/DOWN orientation
% of a page of scanned text stored as a binary
% bit-mapped pattern, with White = 1 and Black = 0,
% assuming that the text is reasonably well
% vertically aligned with the scanner axes. The
% relative amount by which UP orientation exceeds
% DOWN orientation is returned in 'asym', with
% measures of larger absolute value indicating
% greater UP/DOWN asymmetry. The decision is
% returned in boolean variable 'up'. Values of
% parameters 'first', 'range', 'last' and 'thresh'
```

```
% are chosen to suit the characteristics of the
% particular text patterns.
%
first = 170 ;
range = 200 ;
last = 1370 ;
thresh = 0.4 ;
page = imread('text_page.tif', 'tiff') ;
nrows = size(page,1) ;
page = (~page)' ;
csdtpzeros(1,nrows - 1) ;
cssdtp = zeros(1,nrows - 1) ;
for base = first: range: last
   p = sum(page(base: base + range - 1,:)) ;
   tp = min(p, thresh*max(p)) ;
   dtp = tp(2: nrows) - tp(1 : nrows - 1) ;
   sdtp = dtp.^2 ;
   ssdtp = sdtp;
   ssdtp(dtp<0) = -sdtp(dtp<0) ;
   csdtp = csdtp + sdtp ;
   cssdtp = cssdtp + ssdtp ;
end
asym = -sum(cssdtp)/sum(csdtp);
up = asym> = 0 ;
```

References

Akiyama, T., Hagita, N., 1990. Automated entry system for printed documents. Pattern Recognition 23, 1141–1154.

Ali, M., 1997. An object/segment oriented skew-correction technique for document images. In: Proceedings of the Fourth International Conference on Document Analysis Recognition, pp. 671–674.

Bauer, F., 1997. Decrypted Secrets – Methods and Maxims of Cryptology. Springer, Berlin.

Bloomberg, D., Kopec, G., Dasari, L., 1995. Measuring document image skew and orientation. Proc. SPIE 2422, 302–316.

Caprari, R., 2000. Duplicate document detection by template matching, Image and Vision Computing 18 (in press).

Farrow, G., Ireton, M., Xydeas, C., 1994. Detecting the skew angle in document images. Signal Proc.: Image Communic. 6, 101–114.

Hönes, F., Lichter, J., 1994. Layout extraction of mixed mode documents. Machine Vision App. 7, 237–246.

Kahn, D., 1967. The Codebreakers – the Story of Secret Writing. Macmillan, New York.

Le, D., Thoma, G., Wechsler, H., 1994. Automated page orientation and skew angle detection for binary document images. Pattern Recognition 27, 1325–1344.

Rafal, H., Ward, M., 1989. Cartographic character recognition. Proc. SPIE 1199, 1345–1357.