Trainable Multiscript Orientation Detection

Joost van Beusekom[a,b], Yves Rangoni[a], Thomas M. Breuel[a,b]

[a]Image Understanding and Pattern Recognition (IUPR) Research Group
German Research Center for Artificial Intelligence (DFKI) GmbH
D-67663 Kaiserslautern, Germany
[b]Department of Computer Science, Technical University of Kaiserslautern
D-67663 Kaiserslautern, Germany
beusekom@rhrk.uni-kl.de, yves.rangoni@dfki.de, tmb@informatik.uni-kl.de

ABSTRACT

Detecting the correct orientation of document images is an important step in large scale digitization processes, as most subsequent document analysis and optical character recognition methods assume upright position of the document page. Many methods have been proposed to solve the problem, most of which base on ascender to descender ratio computation. Unfortunately, this cannot be used for scripts having no descenders nor ascenders. Therefore, we present a trainable method using character similarity to compute the correct orientation. A connected component based distance measure is computed to compare the characters of the document image to characters whose orientation is known. This allows to detect the orientation for which the distance is lowest as the correct orientation. Training is easily achieved by exchanging the reference characters by characters of the script to be analyzed. Evaluation of the proposed approach showed accuracy of above 99% for Latin and Japanese script from the public UW-III and UW-II datasets. An accuracy of 98.9% was obtained for Fraktur on a non-public dataset. Comparison of the proposed method to two methods using ascender / descender ratio based orientation detection shows a significant improvement.

Keywords: Orientation detection, Multiscript, Character similarity

1. INTRODUCTION

With the advent of high performance automatic document feeding (ADF) scanners the often neglected preprocessing step of detecting the document orientation is becoming more important. As normal flatbed scanners require manual adjustment of the page on the glass window, the orientation of the page is most often corrected manually, thus an automatic reorientation is not needed. In a large scale digitization process, however, the orientation has to be detected automatically, as many subsequent processing steps (e.g. line finding or reading order detection) assume upright positioning of the document image.

In this paper we present a novel approach for orientation detection that uses a character similarity measure. The idea is to compare the characters of the image with unknown orientation — the *input image* — in all four directions to a set of correctly oriented characters called *dictionary*. The orientation in which the characters from the input image are most similar to the correctly oriented characters is considered to be the right orientation. By exchanging the set of correctly oriented characters with characters of another script, this approach can easily be trained to other scripts. This can simply be done i.e. by taking one correctly oriented page containing the script to be analyzed as dictionary.

The remaining parts of this paper are organized as follows: Section 2 gives a short overview of related methods. Section 3 presents the details of our approach for orientation detection. Evaluation and results are to be found in Section 4. Section 5 concludes this paper.

2. RELATED WORK

In literature, methods exist for skew detection and some of those claim to work for all rotation angles. Following this view, orientation detection could be considered as a subproblem of skew detection. Most skew detection methods however only consider rotation angles between $-90\,^\circ$ and $+90\,^\circ$, as can be seen in Cattoni's et al.[1] survey. A clear definition of skew detection and orientation detection is not to be found in literature. Therefore we define orientation detection as the process

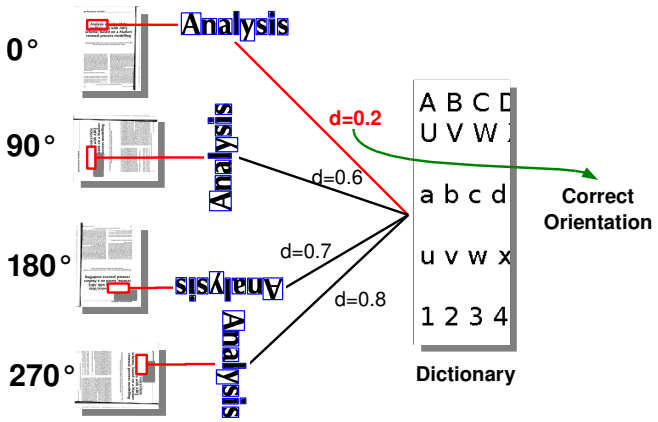


Figure 1. Overview over the proposed approach: the similarity of the connected components to the components in the dictionary is computed for all four orientations. The most similar one is chosen as the correct orientation.

of detecting the orientation of the page among the four possible orientations: correctly oriented page, page rotated by $90\,^\circ$, page rotated by $180\,^\circ$ and page rotated by $270\,^\circ$.

Cattoni et al.[1] present a survey of the state-of-the-art methods for document image understanding and, as a part of this, also for skew detection back in 1998. None of these methods is suited for orientation detection as the presented methods allow for rotation angles between $-90\,^\circ$ and $+90\,^\circ$.

In 1994 Le et al.[2] present an orientation detection system for detecting portrait and landscape mode images. Document images rotated by 180° are not considered. It bases on rules for projection profiles analysis. Hough transform is then used in the second step to determine the skew of the page. An error rate of 0.1% on a non-public dataset of pages of medical journal articles was reported.

In 1995 Bloomberg et al.[3] proposed to use the ascender to descender ratio of the English language for orientation detection. Ascenders and descenders are extracted using morphological operations. The error report on UW-I[4] dataset contains only 1 wrongly detected orientation versus 938 correctly detected. The orientation of 41 documents could not be extracted. An implementation of this method is available in *Leptonica*[*], an open source image processing library.

Recently, another method using the ascender to descender ratio for orientation detection has been presented by Avila et al.[5] The number of ascenders and descenders is computed using the x-height and the base line of the text lines. If the number of descenders is higher than the number of ascenders, the page is considered being upside down. The reported error rate on a non-public dataset is below 0.1%.

In 2006 Lu et al.[6] presented a method for language and orientation detection using the distributions of the number and position of white to black transitions of the components in the line. The performance of their method is reported on a partially non-public dataset and achieves a success rate of 98.2% for documents with at least 12 text lines.

Lu[7] et al. presented in 2007 a method to detect the orientation as well as the skew of a document image. Similar to their previous work, white run histograms are used to detect a characteristic peak. This is then used to estimate the skew angle. Orientation detection is classically solved using ascender to descender ratio. A comparison between their method and other methods found in literature is done. Unfortunately, the test set consists of only 52 documents of a non public dataset.

In our previous work[8] we presented a resolution independent method for orientation detection using a text-line model for Latin script. Evaluation on the UW-I dataset, a subset of the UW-III dataset, proved good accuracy rates of about 99.1%.

[*] http://www.leptonica.com

Comparison to previous work is often difficult: most often neither the implementation used by the authors nor the dataset that was used for evaluation is available. This makes it difficult to compare methods. In this paper we compare the results of our new approach to the line-based approach presented in our previous work[8] and to Bloomberg's approach[3]

3. METHOD DESCRIPTION

The main idea of this approach is to use character similarity to detect the right orientation of a document image. The correct orientation of the input image should be the one where the characters look most similar to characters of a reference document whose orientation is known. This reference image is called the *dictionary*. Examples can be found in Figure 2.

An overview of our approach can be found in Figure 1. In a first step, connected component analysis is done. Using simple statistics about width and height of the components, very small and very big components are rejected. Then, the extracted connected components are compared for each orientation to the connected components of the dictionary. The orientation that leads to the lowest total distance is returned as the correct orientation.

3.1 Distance Measures for Single Characters

In order to compare the characters for all four orientations to the characters of the dictionary, a dissimilarity or distance measure for characters is needed. Instead of characters, connected components are used for comparison. These are easy to extract, and should represent most characters well enough for this purpose. Many different methods for character comparison have been used and studied in literature. In principle, any of these methods could be used, as long as the following condition holds:

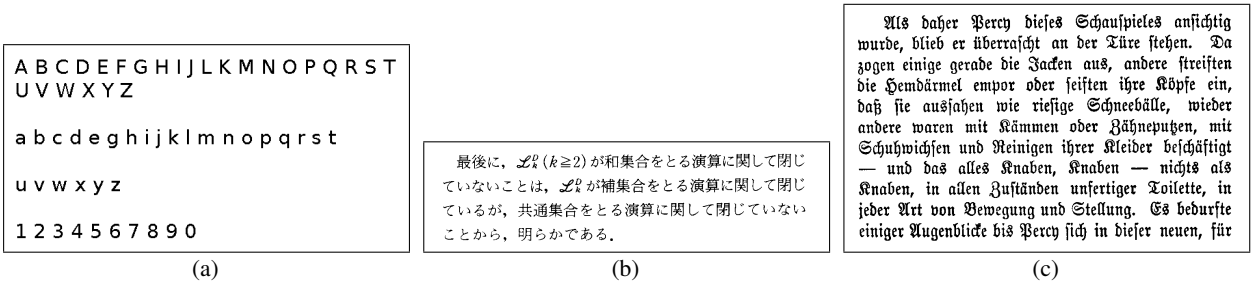
- **rotation variance** is needed as the wrongly oriented (rotated) connected components should be less similar then the correctly oriented connected components

Furthermore, the following attributes are useful for the accuracy and the robustness of the approach:

- **translation invariance** is advantageous to be more robust to noise and font variations
- **scale invariance** is needed to avoid the need of having examples of characters for each admissible font size to the dictionary
- **font invariance** is needed to avoid adding character examples for each different font

For our evaluation of the approach, two different similarity measures were tested:

- *pixel-wise* difference on the downscaled images of the connected components
- complex *Fourier shape descriptors* of the connected components



(a) (b) (c)

Figure 2. Dictionaries used for evaluation on UW-III (left image), UW-II (middle image) and on the Fraktur (right image) dataset. The left image has been generated using standard photo manipulation software while the other two have been cropped from the first image of the UW-II dataset and the Fraktur dataset respectively.

As a preprocessing step for the pixel-wise difference, the connected components are rescaled to the same size. The difference between two rescaled connected components is the sum of the absolute difference of the pixel values:

$$d_{pix}(c_i, c_j) = \sum_{x=0}^{W-1} \sum_{y=0}^{H-1} \text{diff}(c_i(x,y), c_j(x,y)) \tag{1}$$

where

$$\text{diff}(c_i(x,y), c_j(x,y)) = \begin{cases} 1 & \text{if } c_i(x,y) = c_j(x,y) \\ 0 & \text{else} \end{cases} \tag{2}$$

where, W and H are the width and the height of the rescaled components c.

The pixel-wise difference serves as a base-line, as it is a very rudimentary way to compare connected components. In principal, as we are working on a connected component level, it is translation invariant but not robust to noise. i.e. a noise pixel on one side of the character may lead to a translation increasing the pixel difference significantly). Scale invariance is obtained by the rescaling. Font invariance is obtained in a limited way also by the downscaling.

Fourier shape descriptors are also tested as a distance measure for the connected components. The boundary pixels of the connected component are extracted. Then a complex Fourier transform is applied on the sequence of values $x + iy$, where x and y are the coordinates of the boundary pixels. This results in a sequence of Fourier coefficients $f_1, \ldots, f_m$. The low frequencies are represented by the first Fourier coefficients. As the low frequencies represent roughly the shape of the character, considering only the n first coefficients will lead to a certain robustness against font variations and noise pixels around the border. The distance between two sequences of Fourier coefficients is computed as follows:

$$d_{coeff}(c_i, c_j) = \sum_{k=0}^{n} |f_{k,i} - f_{k,j}| \tag{3}$$

where $f_{k,i}$ is the kth Fourier descriptor of component c_i. The invariance to translation is obtained by computing the positions relative to the bounding box position. Scale invariance can be obtained by normalizing the Fourier descriptors by the first Fourier descriptor.[9]

3.2 Distance Measure for Character Sets

Distances between pairs of connected components can now be computed. The problem to be solved is how to use this distance measure to compute the distance between two sets of connected components, in our case, one set coming from the rotated image and one coming from the dictionary.

This can be modeled as a bipartite graph matching problem: the distance between each possible combination of connected components of the image and the dictionary is computed using a component-based distance measure (c.f. Section 3.1). This results in a matrix of size $n \times m$, where n and m are the number of components in the dictionary and the image respectively. The problem to be solved is how to compute a global distance measure between two documents out of the before mentioned matrix. This can be done by solving the minimum weight edge cover problem. This assigns components of the image to components of the dictionary while minimizing the total distance. Each component of the image is assigned to at least one component of the dictionary. The total distance between the two sets is obtained by summing up the distances of the assigned component pairs. An algorithm to solve the minimum weight edge cover problem can be found in.[10] An illustrative example for the bipartite graph representation of the problem can be found in Figure 3.

4. EVALUATION

Evaluation is divided into two parts: in the first experiment, the performance of the two proposed component distances, the pixel-based and the Fourier coefficient based distance, is measured on three different Scripts: Latin, Japanese and Fraktur. In the second part, a performance comparison between our proposed approach, a previously published approach[8] and the approach proposed by Bloomberg[3] is done.

As performance measure we used the accuracy of detecting the correct orientation. The accuracy is defined as the number of images where the orientation could be correctly detected divided by the total number of images in the dataset.

$$\text{accuracy} = \frac{\text{number of document images with correctly detected orientation}}{\text{number of document images in the dataset}}$$

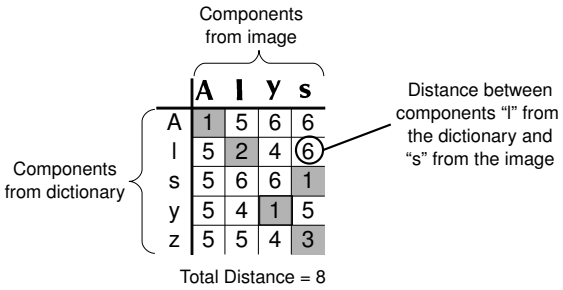


Figure 3. Illustrative example for the minimum weight edge cover. Each component is assigned at least to one other component while minimizing to total distance. The cells with grey background symbolize the assignments that are needed to obtain the minimum weight edge cover.

4.1 Comparison of Connected Component Distance Measures

These experiments has been done on three different sets: the University of Washington document images datasets, UW-III[4] and UW-II, and a private dataset containing pages with Fraktur [†].

The UW-III dataset contains 1600 binarized skew-corrected document images of scientific journals. In a first step we manually removed the 18 images where the orientation is ambiguous, e.g. images containing text in more than one orientation. The following images were removed: *A002, D03N, D04K, D05M, D067, E03F, H008, H00T, H04I, I032, S03F, V009, W0E2, W0H4, W0SB, W1E2, W1H4, W1SB*.

The remaining images have been modified to obtain a set of images with all four orientations equally distributed. This has been done by applying the rotation by $0\,^\circ$, $90\,^\circ$, $180\,^\circ$ and $270\,^\circ$ to the sequence of images. Image A001 thus stays as it is, image A003 is rotated by $90\,^\circ$, A004 by $180\,^\circ$, A005 by $270\,^\circ$, A006 by $0\,^\circ$ and so on.

From the UW-II dataset we took the Japanese journal subset consisting of 477 binary images. The images of UW-II present small skew angles which were not removed beforehand. The images were rotated in the same way as done for the test on UW-III. Japanese script consists of three scripts: “Kanji”, “Hiragana” and “Katakana”. “Kanji” consists of a high number of ideograms, each representing a different idea or word. However, “Hiragana” and “Katakana” consist of fixed alphabets, which allows us to apply our approach to Japanese script.

The Fraktur dataset contains 281 document images of a German edition of the book “Percey Wynn”. Fraktur uses the Latin alphabet but has a very different typeface compared to presently used typefaces. The same rotation scheme as for the other two datasets is applied. The pages containing no text are removed beforehand. Thus only 273 images remain in the dataset.

For each image in the three datasets the orientation is then detected by our method and the accuracy is measured.

The dictionary used for the test on the UW-III dataset can be seen in Figure 2(a). It contains one example of each character and one example of each digit. For the Japanese dataset the dictionary has been generated by cropping a paragraph of text out of the first image of the UW-II dataset. No further preprocessing has been done on this dictionary. The dictionary image used in the experiments can be seen in Figure 2(b). The same procedure has been used for generating the dictionary for the Fraktur dataset that can be seen in Figure 2(c).

For the pixel-based difference measure the components were resized to the size of 16×16 pixels. For the Fourier descriptors, the components were not resized but only the 32 first Fourier coefficients were used to computed the distance.

The results for the two different component distance measures can be found in Table 1. The accuracy rates for the pixel-based measure differ considerably between the different scripts, whereas the Fourier descriptor produces more stable results.

The confusion matrices for Latin script can be found in Tables 4.1 and 3. Most confusions that occur are cases where the computed orientation is $180\,^\circ$ off of the real orientation. This is partially due to the considerable number of Latin characters that have a same appearance but different meaning when rotated by $180\,^\circ$, like e.g. a “b” that becomes

[†] the dataset can be downloaded from http://www.dfki.uni-kl.de/~beusekom/data/percey-wynn.tar.gzip

distance measure	Latin	Fraktur	Japanese
pixel-based	91.5%	99.6%	100%
Fourier descriptors	99.2%	98.9%	99.8%
Line-based approach	99.0%	98.2%	84.1%

Table 1. Results for our approach on Latin, Fraktur and Japanese script. The last line contains the results for the tree datasets using a previously published method[8] using a line-based technique.

a "q", a "n" that becomes a "u", "d" becomes "p" or that are even rotation invariant for 180 °, like e.g. "H", "I", "N", "s", "o" and "l". However this cannot explain why the Fourier descriptor based difference measure works better than the pixel based one. A closer look to the mis-oriented images showed that italics and broken and touching characters are very frequent in these images. The conclusion we draw is that these broken characters introduce much noise into the set of connected components, leading to unpredictable assignments and thus influencing the total distances unpredictably. Again, this problem is also present for the Fourier descriptor based approach. Our final conclusion is that italic characters do not fit well to the not italic characters from the dictionary using pixel-based representation. The Fourier descriptor-based representation is much more robust to italic characters. However, adding characters in italic to the dictionary could solve the problem for the pixel-based distance measure.

Analysis of the assignments obtained for the different component distance measures (examples can be seen in Figure 4) showed that the Fourier descriptors are less sensitive to font variations.

For the test on the UW-II dataset containing images with Japanese script, it can be seen in Table 1 that the accuracy for both distance measures is high, 99.8% and 100% for Fourier descriptor and pixel-based measures respectively. Compared to the Latin script test, it can be noticed, that the difference between both distance measures is neglectable. The fact that italic is rare in Japanese script and also in the dataset supports the hypothesis, that the pixel-based distance measure is sensitive to italic script.

The evaluation for the Fraktur dataset shows good results for both distances measures. Again, the absence of italic text and the uniformity of the font type leads to good results for the pixel-based measure that even slightly outperforms the Fourier descriptors.

The results on the Fraktur and the Japanese dataset show, that the dictionaries do not have to be complete or contain only perfectly rendered characters. The influence of the quality of the dictionary has to be analyzed in future work.

		0 °	90 °	180 °	270 °
Pixel-based	0 °	360	0	34	2
	90 °	0	357	0	39
	180 °	41	0	354	0
	270 °	0	18	1	376
		Total accuracy: 91.5%			

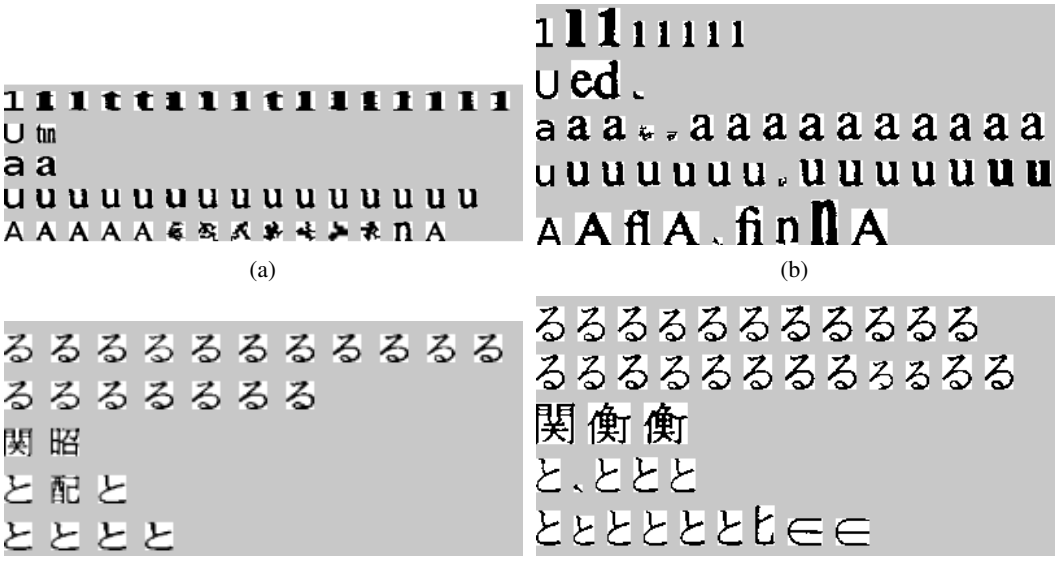
Table 2. Confusion matrix and accuracy for our approach on Latin script using the pixel-based component distance measure.

		0 °	90 °	180 °	270 °
Shape-based	0 °	394	0	2	0
	90 °	0	393	1	2
	180 °	4	0	391	0
	270 °	0	4	0	391
		Total accuracy: 99.2%			

Table 3. Confusion matrix and accuracy for our approach on Latin script using the shape-based component distance measure.

	Bloomberg[3]		Line-Based[8]		Our Results		Ground-Truth
Orientation	Found	Correct	Found	Correct	Found	Correct	GT
top up	936	935	963	962	969	968	970
top left	2	2	5	5	6	6	9
top down	0	0	8	0	1	0	0
top right	0	0	3	0	3	0	0
No result	41	–	–	–	–	–	–
Total Correct		95.8%		98.8%		99.5%	–

Table 4. A comparison of orientation detection results on UW-I dataset. The table shows that our method detects the right orientation for a larger number of documents than the Bloomberg's method. It is also slightly more accurate than our previously published method[8]



(a) (b) (c) (d)

Figure 4. Visualization of the results of the different component measures on Latin and Japanese script. The leftmost columns show the connected components from the dictionary. The other components in each row show the components of the image that have been assigned to the respective dictionary component. The left images show the result for the pixel-based and the right images the result for the Fourier shape-based distance measure.

4.2 Comparison to other Methods

The comparison to the results of the line-based method in Table 1 show that the proposed method gives better results, even for the UW-III dataset, although the difference is not very pronounced. For Fraktur, the difference is higher but still not to far off. As the line model in the previously published approach also fits Fraktur quite well this result is not surprising. Finally one can see that the performance comparison on the Japanese document images identifies the character comparison approach as the clear winner. The reason why the line-based method still works surprisingly well on Japanese could be due to the fact that in the test images the writing direction is horizontal and many documents contain some lines with Latin script. Another possible reason is that most scripts known to the authors have the concept of a base line. Most characters will thus be aligned to the base line. That will lead to ascender lines where less components touch than on the base line.

Performance comparison on the UW-I dataset, a subset of the UW-III dataset shows, that our approach outperforms Bloomberg's approach in terms of accuracy, as can be seen in Table 4.2. In terms of speed however, Bloomberg's method that is part of *Leptonica* is about as twice as fast as our approach, which takes about 11 seconds for one page on a AMD Opteron machine.

5. CONCLUSION AND FUTURE WORK

In this work we presented a new approach for orientation detection. We use a character comparison approach to detect to correct orientation of a document. The connected components from the documents are rotated in all four orientations and are then compared to components that are oriented correctly (dictionary). Using a distance measure to compute the dissimilarity between the components, the orientation with the smallest total distance to the dictionary is chosen as the correct orientation. The proposed approach has been evaluated on two public datasets, UW-III and UW-II for Latin and Japanese script and one private dataset for Fraktur. Overall the Fourier descriptors gave the best and most stable result of 99.2% for Latin script, 99.8% for Japanese script and 98.9% for Fraktur. Comparison to two other methods showed that the proposed method outperforms both.

One important aspect not yet treated is the computational cost of the proposed approach. Therefore in future work it is planned to do a detailed analysis of these costs and also to work on decreasing this costs. Sampling of characters instead of taking all characters could be tried. The connection between accuracy and the number of the Fourier descriptors used is

also of interest, as using fewer descriptors will decrease the computation time. Finally, to prove the wide usability of this method, test sets with other fixed alphabet scripts and a wider variety of fonts have to be collected and used for evaluation.

6. ACKNOWLEDGMENT

This work was partially funded by the BMBF (German Federal Ministry of Education and Research), project PaREn (01 IW 07001).

REFERENCES

[1] Cattoni, R., Coianiz, T., Messelodi, S., and Modena, C. M., "Geometric layout analysis techniques for document image understanding: a review," Tech. Rep. 9703-09, IRST, Trento, Italy (1998).

[2] Le, D., Thoma, G., and Wechsler, H., "Automated page orientation and skew angle detection for binary document images," *Pattern Recognition* **27**(10), 1325–1344 (1994).

[3] Bloomberg, D. S., Kopec, G. E., and Dasari, L., "Measuring document image skew and orientation," in [*Proc. of SPIE Document Recognition and Retrieval II*], 302–316 (February 1995).

[4] Phillips, I. T., "User's reference manual for the UW english/technical document image database III," tech. rep., Seattle University, Washington (1996).

[5] Ávila, B. T. and Lins, R. D., "A fast orientation and skew detection algorithm for monochromatic document images," in [*Proc. of the 5th ACM symposium on Document engineering*], 118–126 (November 2005).

[6] Lu, S. and Tan, C. L., "Automatic document orientation detection and categorization through document vectorization," in [*Proc. of the 14th ACM Int. Conf. on Multimedia*], 113–116 (September 2006).

[7] Lu, S., Wang, J., and Tan, C., "Fast and accurate detection of document skew and orientation," in [*Proc. of the 9th Int. Conf. on Document Analysis and Recognition*], 684–688 (September 2007).

[8] van Beusekom, J., Shafait, F., and Breuel, T. M., "Resolution independent skew and orientation detection for document images," in [*Proc. of SPIE Document Recognition and Retrieval XVI*], **7247** (January 2009).

[9] Jähne, B., [*Digital Image Processing*], 531–536, Springer, Berlin (2002).

[10] Keijsper, J. and Pendavingh, R., "An efficient algorithm for minimum-weight bibranching," *Journal of Combinatorial Theory* **73**(2), 130–145 (1998).