Page Segmentation Performance Using Horizontal Image Strips Instead of Full Page Images

David Poole
Marvell Semiconductor Inc.
Boise, ID USA
Email: dpoole@marvell.com

Elisa Barney Smith
Boise State University
Boise, ID USA
Email: EBarneySmith@BoiseState.edu

***Abstract*—Most page segmentation algorithms are geared toward full page analysis. However, in resource constrained environments, such as dedicated consumer desktop scanners or MFPs (Multi-Function Printers), a full page is not available. Storing a full page would require a cost prohibitive amount of memory.**

We study two page segmentation algorithms (Voronoi and RAST) to discover how each algorithm behaves in the full page environment relative to a smaller image subset. The OCRopus page segmentation toolkit is used as the basis of this work. The tests are run against two sets of scanned document images with known ground truth indicating text/non-text zones, the UW-III LINEWORD images and a BSU dataset. The input images are divided into strips. The results of the performance on the smaller strips is compared to the full page segmentation performance.

Our evaluation shows that RAST performs well in an image strip page segmentation. The results are encouraging enough to study modifications to RAST to improve its performance in an image strip page segmentation.

I. Introduction

High image quality is a trait desired in all document images. In the printer environment this can be increased by differentiating the processing of text versus image zones. Being able to differentiate a text region from a graphics region will allow the firmware/hardware to optimize the output. A text region should use a heavy contrast (clip to black/white) and sharpen. A graphics region should preserve as much dynamic range as possible. A smooth or even a halftone descreen could also be performed on a graphics region. A page can be divided into zones of different content such as text paragraph, business graphic, halftoned image, etc. through page segmentation.

Page segmentation algorithms are usually designed under the assumption that a full page image will be provided. In the consumer market, a scanner/MFP (Multi-Function Printer=a scanner+printer combination) is a commodity and, as such, cost is the driving factor. In a consumer level scanner or MFP a full image, even at low resolution, exceeds available memory. A full 300 dpi monochrome image is 8M. A color scan/copy is 25M. At higher qualities, a 600 dpi scan would be over 100M. A typical consumer MFP has between 8M and 128M. Therefore a page segmentation algorithm that performs well on a small portion, or strip, of a page image is needed so these devices can use content based preprocessing.

In addition to MFPs strip based page segmentation would have application in other areas where a full page image could be too large to store in available memory, such as a microfiche scanner. With the proliferation of mobile devices (phones, tablets, etc.), camera-based imaging has become a topic of interest. However, mobile devices are memory constrained in much the same way as desktop scanner/copier products. Processing on small strips could be performed as described in this paper, rather than storing an entire image in active memory.

This paper explores the behavior of two page segmentation algorithms on image strips. An image strip is the full-page width, but a fraction of the height of the original image. Most page segmentation algorithmns are tested against a full-page image. The strip results are compared to the full-page results. The goal of the experiment is to find a page segmentation algorithm that best scales downward to image strips. The best algorithm would be the topic of further study toward tuning the algorithm for page segmentation with image strips.

II. Previous Work

There are many page segmentation algorithms. Page segmentation algorithms are usually divided into top-down and bottom-up categories [9], [16] with some algorithms considered a hybrid mix of the two. A top-down algorithm examines the entire image, finding structure in the image, then drilling into deeper detail. A bottom-up algorithm examines the image at the pixel level then builds a larger picture of the document from smaller components. A good description of top-down vs. bottom-up segmentation algorithms is available in [2].

Examples of top-down algorithms are X-Y Cut, [12], Whitespace Analysis [2], and Constrained Text Line Detection [4]. X-Y Cut recursively breaks an image into rectangular blocks using horizontal and vertical projection histograms. Rectangles are recursively split based on valleys in the histogram. Whitespace analysis starts with bounding boxes of black connected components. Rectangles of whitespace are built around the black components such that no rectangle intersects with a black component. The resulting rectangles are repeatedly combined to form a cover. The bounding box of the uncovered region is treated as a text segment. Constrained Text Line Detection first finds whitespace rectangles that represent gutters. The gutters become obstacles in computing the bounding boxes of text lines.

Examples of bottom-up algorithms are Docstrum [13], Smearing [19], Voronoi [8], and RAST [5]. Docstrum uses the distance between clusters of connected components. The

k-nearest-neighbors of the components are measured and the docstrum is the plot of all nearest neighbor pairs on a page. Smearing transforms the image at the pixel level by setting 0 pixels to 1 based on their neighbors' values. Black areas are connected together. The algorithm is run along rows then along columns and the two images combined. The Voronoi segmentation starts with centroids of connected components. A point voronoi diagram is built around those points. Extraneous edges are filtered out leaving an area Voronoi diagram of island regions. RAST is a bottom-up algorithm that starts with bounding boxes of connected components. Statistics on the boxes are used to decide if the box contains a character. Next, the whitespace cover is computed, similar to the Constrained Text Line algorithm [4]. The text lines are found based on contiguous character boxes.

Baird [3] has a novel and interesting approach to page segmentation. Each pixel is classified by around 100 features. Thousands of images are used to build a training set. K-Nearest Neighbors is used to classify each pixel in a test document. As one of the admitted assumptions in the paper is enormous computing resources, this algorithm would be inappropriate for the resource constraints targeted in this paper.

In this paper Voronoi and RAST are compared for their potential for use in a memory constrained environment. A third algorithm will be included before the final publication date. Voronoi was judged to be of high quality in [16]. RAST is not part of the performance comparison in [16] or [9]. RAST was judged to have high quality in [18] and is one of the core algorithms in OCRopus [6].

III. Performance Metrics

The most referenced paper studying multiple page segmentation is [16] which used the performance metric as defined in [15]. Another method used in other works is the Page Segmentation Evaluation Toolkit (PSET) [10], [11]. For this paper, the zone comparison program is a simplified version of the metric chosen for the ICDAR 2007 Handwriting Segmentation Contest as described in [1] which is also based on [15].

The text area matching algorithm of [15] calculates a text area matching score by first calculating the intersection of the ground truth bounding box and the result bounding box. The intersection is then divided by the larger of the two areas. A set of metrics on how well the zones were discovered are next calculated. The one-to-many and many-to-one matches are calculated both from the perspective of the ground truth and the detected areas.

For example, a single ground truth bounding box could have been split into multiple boxes in the result. That would be a one-to-many error. Multiple ground truth boxes combined into a single box would be a many-to-one error. Metrics of bounding boxes of zones, measurements such as horizontally/vertically merged lines, false detections, etc. are combined to form a Text-line Accuracy.

The metric chosen in this paper is based on finding only text/non-text zones. The UW-III ZONEBOX files were particularly well adapted to this simple metric. The [18] test image set was divided into bounding boxes for text and non-text zones. The metric itself was calculated using the zone comparison program written as part of [18].

TABLE I. Weights used in the zone comparison program

w1	w2	w3	w4	w5	w6
1.0	0.75	0.75	1.0	0.75	0.75

The weights used in the zone comparison program are shown in Table I. The correct one-to-one matches are weighted more heavily than the one-to-many and the many-to-one matches, i.e., the correct matches are weighed more heavily than the wrong matches.

IV. Strip Based Page Segmentation

The RAST and Voronoi segmentation algorithms are used in this paper based on the implementation of Winder in [18] and further reported in [17]. The segmentation algorithms in OCRopus v0.4.3 were extended with additional page segmentation capabilities. The RAST algorithm was improved by classifying and merging graphics bounding boxes and handling text boxes overlapping graphics. The Voronoi algorithm in OCRopus did not support zone classification and that work was completed by Winder.

The Winder work improved average RAST performance across all image classes by 25%. The original Voronoi implementation did not include zone classification but the overall Voronoi performance was not as good as RAST. Where RAST would segment at 100%, Voronoi would get around 80%.

The RAST and Voronoi page segmentation programs read a PNG file and output a segmented PNG image and an XML zone file. The metric comparison is the same as used in [18] which is the same as used in [15] [1]. The metric comparison reads a ground truth XML zone file and an output XML file. A metric with values in the range $[0, 1.0]$ is output where 0 indicates no match and 1.0 indicates full match.

V. Data

The page images used in this study come from two sources. The first is the UW-III dataset set of 1600 LINEWORD pages. The second is the 91 images of the Winder data set used in [18]. The UW-III dataset is a large corpus of ground truthed scanned documents used in many papers and competitions. The Winder data set is much cleaner than the UW-III dataset. It contains simple high quality scanned pages consisting of one or two columns of text, some pages interspersed with images. Tables II and III describe the contents of these datasets and the categories of page images they contain. The tables list the image classes and an abbreviation (used in the figures) and a short description of the classes is also included.

The UW-III LINEWORD dataset are scans of journals or scans of photocopies of journals. The images are of highly varying quality. Many of the scans have page shadow (dark area as the page fold rises above the scanner glass). Several of the scans have partial images of opposing pages. The UW-III dataset is a difficult segmentation challenge.

The Winder data set [18] consists of scans of pages designed with a specific function. For example, the "Single Column" pages are a full page of text paragraphs. Only the "Magazine" and "Double Column Pictures Scientific" are not pages generated specifically for this dataset. The "Magazine"

TABLE II. UW-III IMAGE CLASSES

Image Class Abbrev.	*Description*	*Pages*
A, C, D, E, H, I, J, K, V	Scans of first generation English journal photocopies	734
N	Scans from English Journal photocopies (supplied by UNLV)	119
S	Direct scans from English Journal pages	125
W	Binary scans from original English Journals and 1st and 2nd generation English journal photocopies	622

TABLE III. WINDER DATASET IMAGE CLASSES

Image Class	*Abbrev.*	*Description*	*Pages*
Single Column	SC	Single column of only text	10
Single Column with Pictures	SCP	Single column of text with picture intermixed	10
Mixed Column	MC	Text in one or two columns	10
Mixed Column with Pictures	MCP	Text in one or two columns with picture intermixed	10
Double Column	DC	Two columns of only text	10
Double Column with Pictures	DCP	Two columns of text with picture intermixed	10
Double Column with Scientific Pictures	DCS	Two columns of text with technical graphic	21
Magazine	M	Scan of magazine page; Graphics with text flowing around the images	10

pages are scans from actual magazines and contain mixed layout text with halftoned graphics. The "Double Column Pictures Scientific" are scans of scholarly papers. As they are all scans of individual pieces of paper, the document image quality is quite good.

All images tested were 300 dpi. The pages were divided into strips of 1 or 2 inches (300 rows and 600 rows). A window slides down the page image, 10 rows at a time (for 300) or 20 rows (for 600). The sliding window simulates the scanned image in memory. As new rows arrive, the oldest rows are ejected. Each image was split into 330 PNG files of 300 rows each and 165 PNG files of 600 rows each.

The UW-III images contain ZONEBOX .box files indicating rectangular ground truth text areas of the full page images. The UW-III .box files only labeled text areas. The Winder images' ground truth is XML files of rectangular zones labeling both text and non-text.

In order to compare the segmentation results from the strip images against a ground truth for pages, the ground truth files are divided into strips. A full page width strip rectangle of 300 or 600 rows was intersected with the ground truth rectangle. The intersection of the two rectangles became a new ground truth rectangle. The page width strip was repeatedly slid 10 or 20 rows down, then the intersection was tested again. The final output of each "slide" is a .XML file indicating the ground truth in each window.

The [18] RAST and Voronoi page segmentation programs were run against the full page to create a behavior baseline. The RAST and Voronoi page segmentation programs were then run against the new strip images. The output XML was saved to a separate file. The [18] zone comparison program based on [1] was run against the sliced ground truth file to produce experimental results.

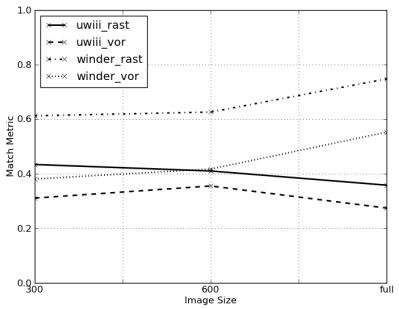


Fig. 1. UW-III and Winder Segmentation Results vs. Image Strip Size

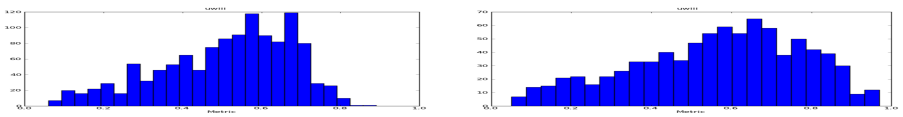
VI. RESULTS

The goal of this research is to find the algorithm that will best maintain the performance of segmenting a full page image when the image is shrunk to a small strip. All images from both the UW-III and Winder datasets with the 300, 600 row strips, and the full page were run with both the RAST and Voronoi algorithms. A comparison of the results of each segmentation algorithm versus strip size per dataset is shown in Figure 1. The Winder data set is much cleaner than the UW-III dataset so the better results for Winder vs. UW-III are no surprise. The RAST algorithm appears to perform much better than Voronoi on both data sets. RAST on a 300 row vs. a 600 row strip in the Winder dataset is little changed. 300 and 600 performed almost identically, but not as well as a full page. There was very little additional improvement in going from a 300 row strip to a 600 row strip which would indicate the smaller strip size (and corresponding lower memory requirements) works well.

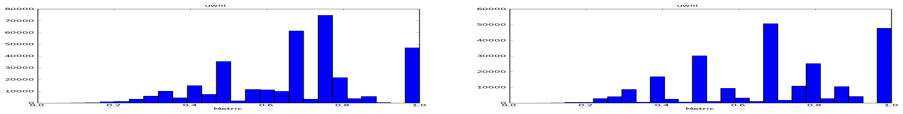
The UW-III results are more puzzling. The degradation of results from the strip images to the full page is understandable given the extra challenges in the UW-III dataset images. When the image was broken up into horizontal strips, we avoided some of the page scanning artifacts. The UW-III Voronoi results show an increase in performance from 300 to 600 then a sharp drop. That result needs further exploration.

Figure 4 shows the performance strip by strip on one somewhat complicated page of two columns with included halftoned graphics. The overall RAST performance was 0.78 on the full page image, while the mean of the sliding window was 0.59. Note in many cases the performance of an individual strip is better than the strips' mean over the whole page and even above the performance of the entire page.

In the figure there are several strips where a metric of zero was reported. There are several reasons a zero/NaN metric would arise. The NaN was replaced with zero during processing. The source of zeros is a complete miss (no matching zones). A NaN results from an empty strip–there is nothing to match and the ground truth also would be empty. The prevalence of white at the bottom of pages especially leads to empty zones. The resulting equation would be 0/0 giving NaN. Because the original metric was not designed to handle empty images (why would the test set contain an empty page?), the metric would give a zero on both an empty image or empty ground truth. The metric measurement program did not expect

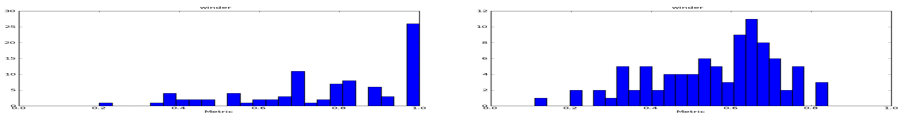


(a) UW-III full page RAST histogram (b) UW-III full page Voronoi histogram

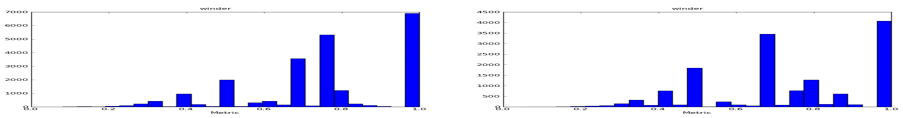


(c) UW-III 300 RAST histogram (d) UW-III 300 Voronoi histogram

Fig. 2. Segmentation results showing histograms of metric occurances on the UW-III dataset.



(a) Winder full page RAST histogram (b) Winder full page Voronoi histogram



(c) Winder 300 RAST histogram (d) Winder 300 Voronoi histogram

Fig. 3. Segmentation results showing histograms of metric occurances on the Winder dataset.

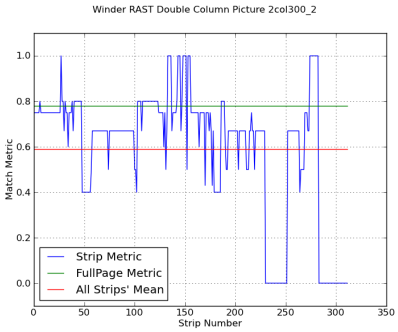


Fig. 4. Segmentation results on a strip by strip basis using RAST. Also shown is the metric on the full page and the mean across all the strips. The source image is from the Winder data set and contains two text columns and a picture.

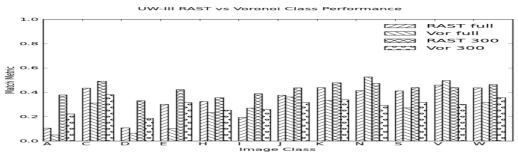


Fig. 5. UW-III Fullpage RAST vs. Voronoi by Class

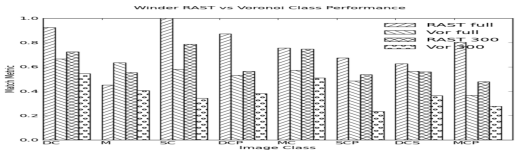


Fig. 6. Winder Fullpage RAST vs. Voronoi by Class

an empty zone box. The bottom third of the page contained a strong graphics location then whitespace causing the zeros which led to the drop from 0.78 to 0.59.

The results of the segmentation on every page and every slice were recorded and are shown as histograms in Figures 3 and 2. Winder has excellent performance on full page RAST and corresponding excellent results with 300 RAST on many strips or pages. The Voronoi results are not as good as RAST. UW-III shows Voronoi performing better than RAST on full pages. However, RAST 300 has much better performance than RAST full page. Not shown in Figures 3 and 2 are the cases when the metric returned a zero or NaN so the zeros would not overwhelm the histogram. Preliminary examination shows 27% of the UW-III results returned a zero and 11% are NaN. Before the final publication date, work will include matching empty ground truth with an empty segmented strip and further study of the cause of zeros.

An analysis of performance by the image classes in Tables II and III is shown in figures 5 and 6. The averages in these figures include the zeros and NaNs (counted numerically as zeros). In the Winder results the full page RAST performs the best in all cases except Magazine (M). The Magazine images contain text flowing around images, sometimes non-rectangular. Voronoi would be better at handling the freer flowing format than the block-oriented RAST. The encouraging result is that RAST 300 is in close second to the full page on several classes. The Multi-Column layout (MC) result is almost identical in RAST full page and RAST 300. Given the restricted size of the segmentation area, the RAST 300

is performing very well. The UW-III results paint a slightly different view. On several of the classes the Voronoi performs better than the RAST. However, the RAST 300 performs as well as or better than all other contenders in ten of the twelve tests!

VII. Conclusion

In this paper, we studied the relative performance of the RAST and Voronoi page segmentation algorithms when the images were constrained to 300 or 600 row strips. The tests were run against the UW-III LINEWORD images and the Winder image test set from [18]. The full page images were divided into smaller images using a sliding window to partially mimic the memory constraints of a desktop scanner.

The RAST results were almost uniformly better than Voronoi. Overall, RAST with a 300 stripsize performed so well that the algorithm is a good candidate for implementation in memory constrained environments. The RAST 600 stripsize had little improvement over the 300 stripsize even in the cases where the full image performed much better than the strip. The one exception was the highly mixed layout from the Magazine images in [18]. The eventual goal would be to implement strip-wise RAST in a hardware circuit (ASIC) for real-time performance in an MFP. Strip-wise RAST in many cases performed better than the full page RAST. Further modifications to strip-wise RAST may contribute to better full page segmentation when memory is not contrained.

Neither the RAST nor the Voronoi program performed any preprocessing on the incoming image other than binarization. The input to the segmentation is the raw source image. There is no attempt to deskew or eliminate page margin issues. The more favorable results with the Winder dataset (compared to the UW-III dataset) is because the Winder dataset is relatively clean. The UW-III dataset has many artifacts that would be unlikely in a modern desktop scanner, especially the large black areas caused by catching multiple page edges in a book scan.

Future work will focus on preprocessing and image clean-up suitable for a strip. The heavy black areas (book page edges) so indicative of the UW-III dataset would be targeted by left/right margin clean-up using connected components. The connected components larger than a specific size and stretched top to bottom of the strip could be ignored.

Acknowledgment

The authors would like to thank...

Amy Winder for her thesis, her code, and her time in explaining both. Dr. David Doermann for the UW-III Data Set CDROM. Python, NumPy [14], and Matplotlib [7] for making life easier.

References

[1] A. Antonacopoulos, B. Gatos, and D. Bridson. Page segmentation competition. In *Document Analysis and Recognition, 2007. ICDAR 2007. Ninth International Conference on*, volume 2, pages 1279–1283. IEEE, 2007.

[2] H. S. Baird et al. Background structure in document images. 1994.

[3] H. S. Baird, M. A. Moll, J. Nonnemaker, M. R. Casey, and D. L. Delorenzo. Versatile document image content extraction. In *Proceedings of SPIE*, volume 6067, pages 215–221, 2006.

[4] T. Breuel. Two geometric algorithms for layout analysis. *Document Analysis Systems V*, pages 687–692, 2002.

[5] T. M. Breuel. A practical, globally optimal algorithm for geometric matching under uncertainty. *Electronic Notes in Theoretical Computer Science*, 46:188–202, 2001.

[6] T. M. Breuel. The ocropus open source ocr system. In *Electronic Imaging 2008*, pages 68150F–68150F. International Society for Optics and Photonics, 2008.

[7] J. D. Hunter. Matplotlib: A 2d graphics environment. *Computing In Science & Engineering*, 9(3):90–95, 2007.

[8] K. Kise, A. Sato, and M. Iwata. Segmentation of page images using the area voronoi diagram. *Computer Vision and Image Understanding*, 70(3):370–382, 1998.

[9] S. Mao and T. Kanungo. Empirical performance evaluation of page segmentation algorithms. In *Proceedings of SPIE Conference on Document Recognition and Retrieval*, pages 303–314. Citeseer, 2000.

[10] S. Mao and T. Kanungo. Pset: A page segmentation evaluation toolkit. In *Fourth IAPR International Workshop on Document Analysis Systems*, pages 451–462. Citeseer, 2000.

[11] S. Mao and T. Kanungo. Software architecture of pset: A page segmentation evaluation toolkit. *International Journal on Document Analysis and Recognition*, 4(3):205–217, 2002.

[12] G. Nagy, S. Seth, and M. Viswanathan. A prototype document image analysis system for technical journals. *Computer*, 25(7):10–22, 1992.

[13] L. O'Gorman. The document spectrum for page layout analysis. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 15(11):1162–1173, 1993.

[14] T. E. Oliphant. *Guide to NumPy*. Provo, UT, Mar. 2006.

[15] I. T. Phillips and A. K. Chhabra. Empirical performance evaluation of graphics recognition systems. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 21(9):849–870, 1999.

[16] F. Shafait, D. Keysers, and T. Breuel. Performance comparison of six algorithms for page segmentation. *Document Analysis Systems VII*, pages 368–379, 2006.

[17] A. Winder, T. Andersen, and E. H. Barney Smith. Extending page segmentation algorithms for mixed-layout document processing. In *Proceedings International Conference on Document Analysis and Recognition*, pages 1245–1249, Beijing, China, September 2011.

[18] A. A. Winder. *Extending the Page Segmentation Algorithms of the OCRopus Documentation Layout Analysis System*. PhD thesis, Boise State University, 2010.

[19] K. Y. Wong, R. G. Casey, and F. M. Wahl. Document analysis system. *IBM journal of research and development*, 26(6):647–656, 1982.