2011 International Conference on Electrical Engineering and Informatics
17-19 July 2011, Bandung, Indonesia

Recursive Text Segmentation for Indonesian Automated Document Reader for People with Visual Impairment

Teresa Vania Tjahja[#1], Anto Satriyo Nugroho[*2], James Purnama[#],
Nur Aziza Azis[*], Rose Maulidiyatul Hikmah[*], Oskar Riandi[*], Bowo Prasetyo[*]

[#]*Faculty of Information Technology, Swiss German University*
EduTown BSD City, Tangerang 15339, Indonesia
[1]teresa.tjahja@student.sgu.ac.id

[*]*Center for Information and Communication Technology (PTIK),*
Agency for the Assessment and Application of Technology (BPPT)
Jalan M.H. Thamrin No. 8, Jakarta 10340, Indonesia
[2]asnugroho@ieee.org

Abstract— **This research is conducted to accommodate the needs of visually impaired people through an intelligent system, which reads textual information on papers and produces corresponding voice. Indonesian Automated Document Reader (I-ADR) is operated via a voice-based user interface to scan a document page. Textual information from the scanned page is then extracted using Optical Character Recognition (OCR) techniques. A user can then choose to have the system read the whole page, or they can opt to listen to a summary of the information in page. SIDoBI (Sistem Ikhtisar Dokumen untuk Bahasa Indonesia) is integrated into the system to provide summarization feature. The result of either the whole-page reading or summarization is converted to speech through a text-to-speech synthesizer. This whole system is developed under the Free Open Source Software policy and will be distributed openly to all users in need without any cost. This paper is focused on the text segmentation algorithm implemented in I-ADR to extract text from documents with complex layout. We implemented I-ADR text segmentation module using Enhanced CRLA and propose an improved algorithm for text extraction. Evaluation of the proposed system with various page layouts showed promising results.**

Keywords— **visual impairment, text segmentation, text summarization, text-to-speech synthesizer, OCR.**

I. Introduction

Despite the advance of technology that allows for storing information electronically, textual information presented on papers remains the most common, especially in Indonesia. However, such information is not available for visually impaired citizens. To improve their ability to access textual information, Agency for the Assessment and Application of Technology (Badan Pengkajian dan Penerapan Teknologi; BPPT) develops Indonesian Automated Document Reader (I-ADR), an application that reads texts from scanned documents and represents the textual information in the form of speech.

Previously, a study has been conducted, producing a prototype of I-ADR application (formerly named TextReader) that covered important modules such as Optical Character Recognition (OCR) and Text-to-Speech (TTS) synthesizer [1]. This research is a continuation of the previous study that focuses on improving the text segmentation module as an integral part of OCR. Current implementation of I-ADR employs Enhanced Constrained Run-Length Algorithm (CRLA) [2] for text segmentation. There are various ways of using Enhanced CRLA, particularly on how to extract text regions, which is not explained in detail in Enhanced CRLA paper. We evaluated a number of combinations that will be discussed in this paper.

The next sections of this paper are structured as follows: section II provides an overview of assistive technology for visually impaired people and section III explains the proposed system. Experimental results are discussed in section IV and concluded in section V.

II. Assistive Technology for Visual Impairment

Visual impairment refers to some degree of loss of vision. It implies that a visually impaired person does not necessarily have zero ability to see – they may still have limited sight. To support people with visual impairment, various technologies have been invented, including accessibility support to assist visually impaired people when using computers. Screen magnifiers and screen readers are two most common assistive tools [3]. Screen magnifiers assist a user by providing a zoomed part of the computer screen, while screen readers produce speech (or Braille with special-purpose hardware) that represents screen elements in order to help a visually impaired person navigate throughout the screen.

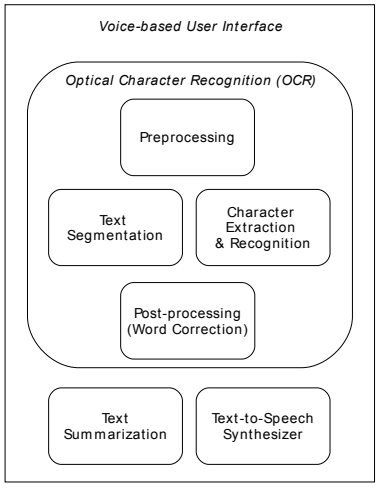


Fig. 1 I-ADR modules.

In the past, assistive software – such as screen magnifier and screen reader – was operated with a keyboard. However, such interface might be more suitable for people with certain degree of sight, and is relatively difficult for users with no vision. In addition, the software usually reads from digital form of documents and still has less support for reading from images (i.e. scanned documents). To respond to such needs, there are an increasing number of assistive technologies being developed to read textual information from scanned documents, which can be operated via a voice-based user interface.

III. Proposed System

In order to help visually impaired people to get textual information from paper documents, I-ADR is proposed. The system consists of five general modules as shown in Fig. 1. Post-processing, text summarization, and text-to-speech synthesizer modules are tailored to Indonesian language, while the other modules can also process documents in any language (which uses Latin alphabets).

Given the design of I-ADR, this study is focused mainly on text segmentation. Voice-based user interface, text summarization, and text-to-speech synthesizer implement existing modules, which are integrated into I-ADR system.

A. Pre-processing

Pre-processing aims to enhance the image quality and simplify the image representation for subsequent operations. In the current implementation of I-ADR, median filter is used to reduce noise and simplification of image representation is achieved by binarization (using Otsu's thresholding scheme) [4].

B. Text Segmentation

Text segmentation module uses Enhanced CRLA [2] to detect text region in an image. In general, Enhanced CRLA assigns labels to each connected component in an image according to its size and uses Selective CRLA to group similar connected components, hence forming homogeneous regions that consist of components of similar height. A connected

```
DETECT-TEXT-REGIONS(IMG, Xstart, Xend, Ystart, Yend)
1  for j ← Ystart to Yend
2    if COUNT-VERTICAL-HISTOGRAM(IMG, j, Xstart, Xend) > 0
3      then ys ← j
4           ye ← GET-REGION-YEND(IMG, ys, Xstart, Xend)
5           j  ← ye
6      for i ← Xstart to Xend
7        if COUNT-HORIZONTAL-HISTOGRAM(IMG, i, ys, ye) > 0
8          then xs ← i
9               xe ← GET-REGION-XEND(IMG, xs, ys, ye)
10              i  ← xe
11            for k ← ys to ye
12              if COUNT-VERTICAL-HISTOGRAM(IMG, k, xs, xe) > 0
13                then y2s ← k
14                     y2e ← GET-REGION-YEND(IMG, y2s, xs, xe)
15                     k   ← y2e
16                     ▷ Calculate MBRL and MTC
17                     if MBRL ≥ MBRLmin AND MBRL ≤ MBRLmax
18                       AND MTC ≥ MTCmin AND MTC ≤ MTCmax
19                       then height ← y2e − y2s
20                         ifheight > EXPECTED_LINE_HEIGHT ANDxs ≠Xstart
21                           AND xe ≠ Xend AND y2s ≠ Ystart AND y2e ≠ Yend
22                           then DETECT-TEXT-REGIONS(IMG, xs, xe, y2s, y2e)
23                           else ▷ Mark region as text
24                       else
25                         if xs ≠ Xstart AND xe ≠ Xend AND y2s ≠ Ystart
26                           AND y2e ≠ Yend
27                           then DETECT-TEXT-REGIONS(IMG, xs, xe, y2s, y2e)
28                           else ▷ Mark region as non-text

COUNT-VERTICAL-HISTOGRAM(IMG, row, Xstart, Xend)
1  count ← 0
2  for i ← Xstart to Xend
3    if IMG(i, row) ≠ BACKGROUND_PIXEL
4      then count ← count + 1
5  return count

COUNT-HORIZONTAL-HISTOGRAM(IMG, col, Ystart, Yend)
1  count ← 0
2  for i ← Ystart to Yend
3    if IMG(col, i) ≠ BACKGROUND_PIXEL
4      then count ← count + 1
5  return count

GET-REGION-XEND(IMG, x2s, xe, ys, ye)
1  x2e ← -1
2  for i ← x2s to xe AND x2e ≠ -1
3    if COUNT-HORIZONTAL-HISTOGRAM(IMG, i, ys, ye) > 0 OR i = xe
4     x2e ← i
5  return x2e

GET-REGION-YEND(IMG, y2s, ye, xs, xe)
1  y2e ← -1
2  for i ← y2s to ye AND y2e ≠ -1
3    if COUNT-VERTICAL-HISTOGRAM(IMG, i, xs, xe) > 0 OR i = ye
4      y2e ← i
5  return y2e
```

Fig. 2 Pseudocode for detecting text regions.

component can be either a character or a picture (or parts of a picture). Therefore, in a document page, there may be hundreds to thousands of connected components. Given the number of connected components in a page, Connected Component Labeling (CCL) [5] may take a relatively long time. To speed up the performance, the image is scaled to a smaller size prior to CCL.

After applying Enhanced CRLA, each region (e.g. text line, picture) should be classified as either text or non-text by calculating its MBRL and MTC values. For the calculation, each individual region needs to be accessed. This step is not explained in detail in Enhanced CRLA paper [2]. Therefore, we evaluated several approaches and propose one that utilizes histogram analysis with a combination of iteration and recursive function.

First, histogram analysis is applied vertically, counting the number of non-background pixels in every row. When a row is known to have non-background pixels, it is determined as a possible starting row of one or more regions. Then, the ending row is defined as the first row containing no non-background pixels after the starting row. Since the image being analyzed is a label image, the value of each pixel is not limited to only 0 and 1 as in a binary image. A pixel in a label image may contain 0 to represent background, or one of the size labels assigned according to the component's height that the pixel

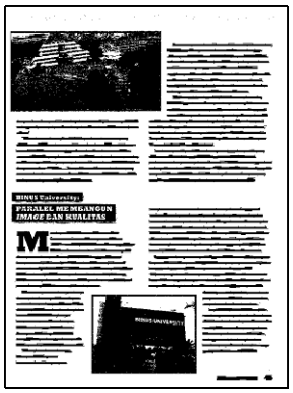


Fig. 3 Result of implementing Enhanced CRLA.

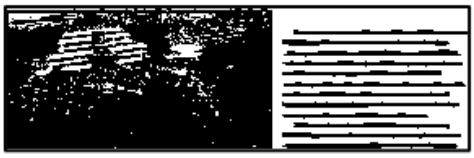


Fig. 4 Example result of first vertical scan.

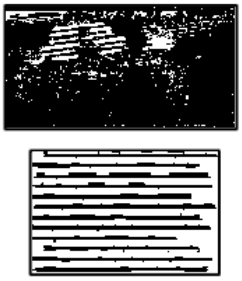


Fig. 5 Example result of horizontal scan.

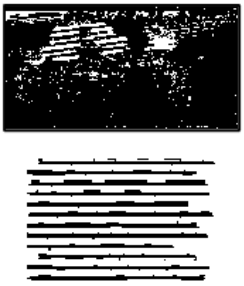


Fig. 6 Example result of second vertical scan.

belongs to. So, a non-background pixel has a value other than 0 (i.e. might be one of the size labels).

Once the starting and ending rows are obtained, similar histogram analysis is performed horizontally. While the first vertical scan aims to get a line of text (or text with a picture expanding to several lines of text), the horizontal scan is done to separate the region into columns. At this point, we have a range in the image that is estimated to include one or more homogeneous regions. To ensure that there is only one individual region in the range, vertical histogram analysis is performed once more in that range, further dividing the range vertically in case the range contains more than one region. After this third histogram analysis, we have a new range that should consist of only one region. Pseudocode for the proposed algorithm can be found in Fig. 2.

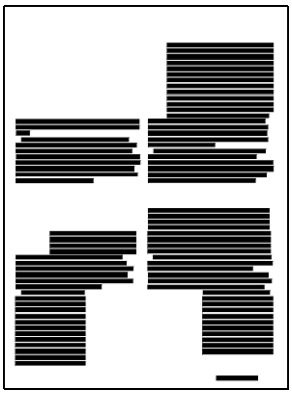


Fig. 7 Extracted text regions.

Figure 4, 5, and 6 shows the example result of each scan during detection of text regions. The first vertical scan divides an image vertically. Fig. 4 shows the first range from the image in Fig. 3 obtained with this first scan. The horizontal scan divides the range into two parts as shown in Fig. 5: left part, which consists of the picture, and right part, which includes the text lines. Finally, each part is further divided vertically to get individual regions. The picture remains as one region, while the text is divided into individual text lines (see Fig. 6). The extracted text regions are shown in Fig. 7.

Once an estimated individual region is acquired, its MBRL and MTC are calculated. If the values of MBRL and MTC indicate that it is a text region, it is not marked directly as text region, but rather is examined further based on its height. If its height is much larger than the expected height of a text line (e.g. double of the average height), histogram analysis from the first row-by-row scheme is performed again on that particular region (recursive). Otherwise, the region is marked as text. Similarly, when the region is detected as non-text, the 3-step histogram analysis is performed again in that region. Finally, when the range obtained after the 3-step histogram analysis is the same as the original range before the analysis was started (i.e. the range cannot be divided any further), it is marked as text or non-text depending on its MBRL and MTC.

Recall that the image was scaled to a smaller size before applying CCL in the initial step. Due to loss or addition of pixels during the scaling process, most characters have rugged shape compared to their original shape, which in turn will affect character recognition performance. Therefore, the image needs to be rescaled to its original size after text regions were detected. At this point, characters are still grouped in homogeneous text regions (as in Fig. 7). The rescaled label image is then combined with the original binary image, which was produced earlier by pre-processing module, in order to extract only text from the original image.

C. Character Extraction and Recognition

Given the resulting image from layout analysis, the next step is to extract individual characters (e.g. letters, numbers, and symbols/punctuation marks). Since it is assumed that the

image contains only text, histogram analysis is sufficient to separate and extract each character.

When an image of character is extracted, the image is normalized to a predefined size. In the normalization process, blank space surrounding the character is eliminated, so the image now contains minimum blank space around the character. Then, the character image is normalized to 12x12 pixels.

After character extraction and normalization, each extracted character from the previous step is recognized. I-ADR implements OCR with Multilayer Perceptron (MLP) Neural Network trained with Backpropagation algorithm. In the current implementation, the MLP neural network architecture has 3 layers (input, hidden, and output layers) with the number of nodes in each layer 144, 100, and 73, respectively. The terminating condition for MLP neural network training phase is when it has performed 10,000 iterations or the Mean Square Error (MSE) is less than or equal 0.0001.

At this point, a string of recognized characters is obtained. To produce an understandable text, the characters must be arranged into words by inserting spaces between the characters. The position of a space between two words is approximated as 2.6 (obtained experimentally) times of the average horizontal distance between characters in a word. Thus, for each detected character through histogram analysis, the distance between it and the previous character is calculated. When the distance is larger than 2.6 times the average distance in the word, a space is inserted.

D. Lexicon-based Post-processing

In character recognition, a character is often recognized as another with similar shape. For example, I is recognized as l or the number one (1), o as the number 0, etc. Therefore, to produce meaningful speech, words produced by character recognition module should be corrected before passed to the text-to-speech synthesizer. I-ADR implements a lexicon-based post-processing to perform word correction. The algorithm compares recognized words with a list of Indonesian dictionary words. To measure the similarity between two words, Longest Common Subsequence (LCS) [6] is used.

E. Text Summarization

I-ADR integrates MEAD [7][8] and its Indonesian database provided by SIDoBI (Sistem Ikhtisar Dokumen Bahasa Indonesia) [9][10] to provide automatic summarization feature. MEAD performs extractive summarization, where units in documents (sentences and words) are assigned salience scores, which determine whether the units should be extracted to construct the summary.

In MEAD, scores for the units in documents are calculated based on features. Feature calculation involves comparing sentences, which uses an IDF database. IDF database for Indonesian language is provided by SIDoBI.

Automatic summarization feature in I-ADR is implemented by executing MEAD through its command-line interface from within the ADR program.

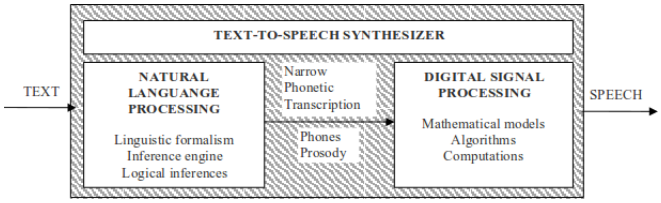


Fig. 8 Functional diagram of TTS synthesizer [11].

F. Text-to-Speech (TTS) Synthesizer

Text-to-speech is the production of speech by machines, by way of automatic *phonetization* of the sentence to utter [11]. Oral reading process by a text-to-speech system should be intelligible and natural. Hence, text-to-speech synthesizer is different with any other talking machine such as cassette player. In the TTS synthesizer, new sentences are generated automatically. Text-to-speech synthesizer is different with specific talking machine termed as voice response system (for example machine for announcing arrival in train station). Voice response system simply used concatenation of recorded words of limited vocabulary whereas it is impossible for TTS synthesizer to record all the words of the focus language.

Figure 8 shows the functional diagram of general TTS synthesizer. It consists of Natural Language Processing (NLP) module and Digital Signal Processing (DSP) module. Natural language module is responsible for conversions of digital text into corresponding phonetic transcription and prosody (intonation and rhythm) information. Digital signal processing module then transforms the resulting phonetic transcription and prosody information into natural-sounding speech.

To perform the conversion from text to speech, I-ADR uses a free and open source speech synthesizer named MBROLA [12] with Indonesian voice database that has been created by Arry Akhmad Arman. MBROLA takes phoneme commands (phonetic transcription with prosody) as input and produces an audio (WAV) file as output. It thus requires a phoneme commands script generated from the recognized text (or summary).

IV. Experimental Results and Discussion

I-ADR system was experimented to read pages from magazines. Magazine pages were chosen due to their arbitrary, complex layout with text divided into columns and pictures among text columns. Images of the magazine pages were scanned using a 300 dpi scanner and the proposed system was evaluated with gray-scale images. We developed the system and conducted our experiments on a laptop with Intel® Core™ i7 CPU @ 1.7 GHz and 4 GB RAM on Ubuntu Linux 10.10 platform. The following subsections discuss the experimental results that are focused on text segmentation and multi-font character recognition.

A. Text Segmentation Result

Before examining homogeneous regions produced by enhanced CRLA, the range (i.e. coordinates of starting and ending rows and columns) of each region should be determined. It is implemented using histogram analysis.

As explained in section III, the analysis is performed on a label image, meaning that every connected component in the image has one of the size labels according to its size. Assuming that components with size label -1 are estimated as text, a simple approach is to count only the number of pixels with -1 label during histogram analysis. The result of such approach is shown in Fig. 9(b). As can be seen in Fig. 9(b), there are several spots where small parts of non-text object are extracted. Although the algorithm is expected to extract only text objects, this might happen because those parts of non-text objects have certain MBRL and MTC values to be classified as text, especially in gray-scale images.

An alternative is to count the total number of non-background pixels in a particular row or column during histogram analysis. This approach produces text extraction result as shown in Fig. 9(c). The resulting image is cleaner than the previous approach, with the unexpected spots eliminated.

One particular problem is that some inner parts of non-text objects were also labelled -1 since they had height similar to text objects. Those inner parts were actually surrounded by large non-text objects that had different labels. However, with the former approach, histogram analysis practically saw only -1 and background labels. So, when histogram analysis was conducted to find the starting and ending points of a region, it failed to see that those -1 labels were actually parts of a larger non-text object. In addition, MBRL and MTC calculation showed that those spots had similar characteristics to a text region, causing them to be marked as text.

On the contrary, histogram analysis in the second approach is able to see size labels other than -1 label. As for the above case, the algorithm treated the non-text object as a whole. That means, since it saw that there were other non-background labels around those spots, the non-text object was considered as one region; it did not see the inner parts as different regions.

At a glance, 3-step histogram analysis appears to be sufficient to extract all text, without having to use recursive scheme. However, there are cases of which the recursion as described in section III is needed. Figure 10(b) shows text segmentation result with only 3-step histogram analysis without recursion, while Fig. 10(c) contains the result obtained by applying 3-step histogram analysis and the recursion. There are several lines of text missing in Fig. 10(b), while all text lines in the article are extracted successfully in Fig. 10(c).

In our experiments, 7 images of magazine pages were used (average size: 2381x3153 pixels). The images were scaled to their 25% size before Connected Component Labeling. In all 7 images, all lines in the text body were detected successfully. The average time required for text segmentation was 11.4 seconds. Titles and captions as shown in Fig. 11 were not extracted since they were considered non-text in the current implementation.

B. Character Recognition Result

Multi-font database is used for character recognition, particularly in training the neural network classifier. Since this

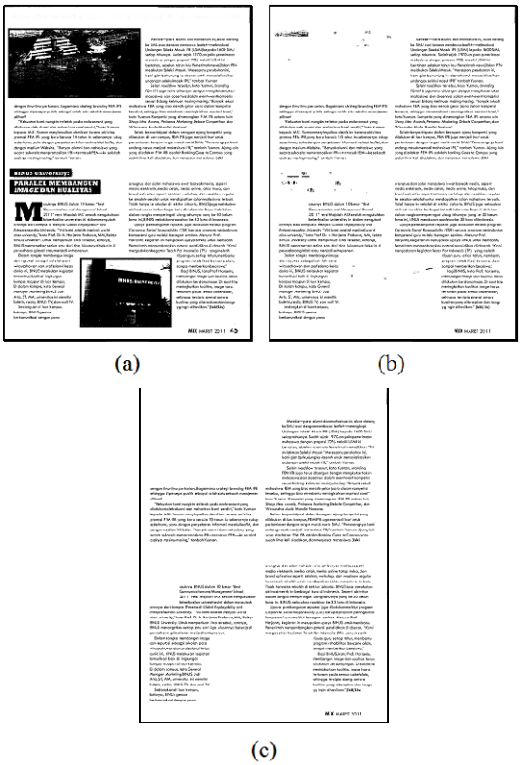


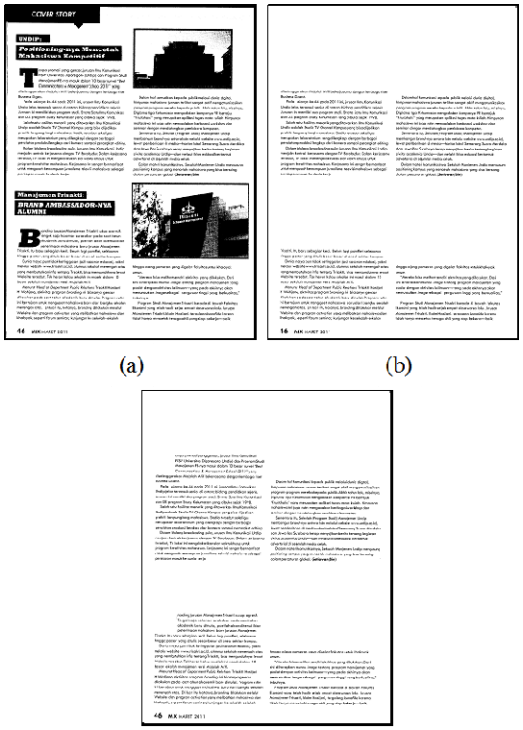
Fig. 9 (a) Original image. (b) Text segmentation result of counting only the number of "text" pixels when examining homogeneous regions. (c) Text segmentation result of counting the number of non-background pixels when examining homogeneous regions.

research is focused on reading from printed documents, the database was constructed by 15 commonly used computer fonts. The database includes 73 characters: 26 upper case letters, 26 lower case letters, 10 numbers, and 11 symbols and punctuation marks.

The most common problem in character recognition is characters with similar appearances. For example, i, I, l, and 1; I, l, or 1 italic and /; o, O, and 0; and several lower-case and upper case letters (e.g. s, v, u, etc.). Due to the font type used in the magazines, a and o also have identical shapes. Here, we assume that errors caused by similar-shaped characters are tolerable, and they are expected to be corrected by post-processing module. Among 3,485 characters in the image, 104 were not recognized correctly (2.98% error rate) – excluding tolerable errors. Most errors were found on characters that had imperfect shape (lost some or obtained additional pixels during image scaling process). In addition to imperfect character shape, recognition errors were also caused by bad character segmentation, i.e. more than one character was segmented as one character image. The process up to character recognition required an average time of 11.9 seconds.

V. CONCLUSION

I-ADR is an assistive system aiming to support people with visual impairment to get textual information printed on paper. It reads texts from scanned documents and converts them into speech. The current version of I-ADR covers text



(a) (b) (c)

Fig. 10 (a) Original image. (b) Text segmentation result without recursion. (c) Text segmentation result with recursion.

(a) (b)

Fig. 11 Titles with appearance like pictures (does not have black as foreground and white as background) are not extracted. (a) Original image. (b) Text segmentation result.

segmentation, character recognition, text summarization, and text-to-speech modules. Voice-based user interface to operate the system is to be integrated in the near future.

The main focus of this paper is text segmentation module, which extracts text from a page. Enhanced CRLA is a text segmentation algorithm capable of extracting text from complex page layouts [2]. During our implementation, we considered various implementations of Enhanced CRLA and evaluated them to get the best result.

First, due to the large number of connected components (characters, pictures, fragments of pictures, etc.) in an image, CCL took a long time. So, the image was scaled to a smaller size to speed up the process. Next, we follow Enhanced CRLA from CCL to forming homogeneous regions with selective-CRLA. From this point, there are different approaches of how we access each individual homogeneous region in order to determine whether the region is a text or non-text region. We propose histogram analysis to divide the image by recursive scanning in 3 steps (vertical, horizontal, then vertical again) that produces the best result. Furthermore, Enhanced CRLA is performed on a label image – in which a pixel does not have an intensity value, but rather a size label depending on the size of component that it belongs to. Assuming that we use size label -1 to indicate an expected text pixel, a simple approach is to consider only pixels with -1 label during histogram analysis. However, we obtained better, cleaner results if we consider also other size labels (non-background pixels) as discussed in section IV (Fig. 9). The algorithm works well to extract all text in the page body with minimum noise.

In the future, I-ADR will be improved, especially to provide significant features such as color images processing and integration with voice-based user interface. Also, histogram analysis scheme in the current implementation is more suitable for pages with Manhattan layout. It needs modification to handle non-Manhattan layout.

ACKNOWLEDGMENT

The authors would like to thank Swiss German University for providing the opportunity to conduct and publish this research.

REFERENCES

[1] A. S. Nugroho, I. Suwadi, V. Pragesjvara, R. Irbandini, O. Riandi, M. Gunawan, D. Handoko, "TextReader: An Intelligent System for People with Visual Impairment," in *Proc. of International Conference on Advance Computer Science & Information System (ICACSIS 2010)*, pp. 393-396, Bali, Indonesia, 2010.

[2] H. M. Sun, "Enhanced Constrained Run-Length Algorithm for Complex Layout Document Processing," *International Journal of Applied Science and Engineering*, 4, 3: 297-309, Dec. 2006.

[3] M. A. Hersh et al., *Assistive Technology for Visually Impaired and Blind People*, 1st ed., M. A. Johnson, Ed., London, UK: Springer-Verlag, 2008.

[4] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1): 62-66, 1979.

[5] L. Di Stefano, A. Bulgarelli, "A Simple and Efficient Connected Components Labeling Algorithm" in *Proc. of International Conference of Image Analysis and Processing (1999)*, pp. 322-327, Venice, Italy, 1999.

[6] N. C. Jones and P. A. Pevzner, *Bioinformatics Algorithm*, Bradford: 2004.

[7] D. Radev, T. Allison, S. Blair-Goldensohn, J. Blitzer, A. Celebi, S. Dimitrov, E. Drabek, A. Hakim, W. Lam, D. Liu, J. Otterbacher, H. Qi, H. Saggion, S. Teufel, M. Topper, A. Winkel, Z. Zhang, "MEAD - a platform for multidocument multilingual text summarization" in *LREC 2004*, Lisbon, Portugal, May 2004.

[8] http://www.summarization.com/mead [Accessed May 26, 2011]

[9] B. Prasetyo, T. Uliniansyah, O. Riandi, "SIDoBI: Indonesian Language Document Summarization System," in *Proc. of International Conference on Rural Information and Communication Technology*, pp. 378-382, Bandung, Indonesia, 2009.

[10] http://www.sourceforge.net/project/sidobi [Accessed May 26, 2011]

[11] T. Dutoit, *An Introduction to Text-to-Speech*, Kluwer Academic, 1997.

[12] http://tcts.fpms.ac.be/synthesis/mbrola.html [Accessed May 26, 2011]