https://doi.org/10.2352/ISSN.2470-1173.2018.13.IPAS-305

Separation of Scanned Media using a strip based methodology

Osborn de Lima[a], Eli Saber[a,b], Kevin Merrill[c], Mark Shaw[c]
[a]Chester F. Carlson Center for Imaging Science, Rochester Institute of Technology, Rochester, NY
[b]Department of Electrical and Microelectronic Engineering, Rochester Institute of Technology, Rochester, NY
[c]HP Inc, San Diego, CA/Boise, ID

Abstract

The use of Multi-Function Printers (MFPs) in the office and home-office setting continues to trend upwards despite the advent and use of mobile technology in that space. One such novel use case, is scanning multiple distinct objects that are either related or different from each other in terms of content. The proposed algorithm seeks to achieve separation of distinct objects placed on a flat-bed scanner and save each object as a different file. Further processing and identification can then be performed on each item for enhanced use. However, due to the memory limitations on some MFPs a strip based processing technique is utilized in this case, wherein only a portion of the acquired scan is processed each time. The technique utilizes edge detection, thresholding, morphology and a connect component analysis scheme along with label propagation from one strip to another to achieve the goal of media separation. The results obtained are promising for the speed and memory constraints imposed.

Introduction

The printer, scanner, copier and their manufacturers are currently in a battle for relevance with the internet and the smartphone. Technology has largely replaced the need for printers in the home and to a certain extent the home-office spaces due to the reliability of digital storage and the changing way in which business is done, not to mention the cost savings and positive environmental impact. However, the Multi-Function Printer (MFP) still occupies a significant place in the Enterprise space. Paper is still used for a wide variety of business, particularly in the financial sector.

The scanner forms an important backbone of business, as it converts an analog form, i.e. paper, to digital content, which in turn can be transmitted or backed up. In some cases, there are hardware modifications used to scan certain types of documents such as checks. In other cases, there are multiple instances of smaller size objects that need to be scanned. Therefore, a software-based solution that is computationally inexpensive and fast, enables the consumer to speed up the process by which data is scanned. This solution can lead to potential cost savings by removing dedicated hardware. The context behind this research is to develop this software solution using image processing techniques that can be easily implemented and supported on a slower, cheaper platform that is local and does not necessarily have the horsepower of a cloud-based system. Additionally, the memory constraints of the target devices this algorithm is intended to operate on, call for a strip-based methodology, wherein only a portion of the image is available for processing.

A bulk of the work related to document image processing seeks to classify regions of the document as text vs non-text. This serves as a major pre-processing step for Optical Character Recognition (OCR) methods as well as other document retrieval schemes. A benchmark of such algorithms can be found in [1]. Further in that regard, the printing industry [2] has taken a keen interest in page classification algorithms to offer adaptive rendering and printing schemes. Erkilinc et al [3] seek to discriminate and identify text, photos and lines in scanned documents. They achieve this by applying a different technique to each class of data, namely, a MRF based approach for photos, wavelet transforms and Run Length Encoding (RLE) for Text and, a variety of edge and line detection based methods such as the Hough Transform for Lines. This information is then fused together in a final clustering step that uses a K-Means approach. While this technique seeks to address multiple modalities of information within a document which was a target of our research, the high cost of implementation and runtime proved challenging to adapt to the problem at hand.

In [4], Chiu et al, utilize OCR to determine the text pixels in a document image and then a normalized cuts clustering technique to highlight the non-text pixels. The method seeks to exclusively determine pictures in document images and utilizes the strength of OCR in locating text pixels. In addition to the high complexity of running the OCR engine, this method only targets pictures to segment from the document. While the clustering process can be tuned to other media it would not be efficient. Clustering appears to be a common theme when it comes to the process of document image segmentation. In contrast to using normalized cuts in the previous approach, Lin et al [5] utilize texture features to separate document images into graphics, text and space. Gray Level Co-occurrence matrices (GLCMs) are calculated for each block of the image. Texture features do tend to fall on the high end of the computational spectrum, however, the idea to divide the document into different regions utilizing properties due to material differences that manifest itself in image data was considered.

A statistical approach by Srivastava et al in [6] hypothesis that objects in document images have a pixel value distribution representing a narrow gaussian about nominal mean value and utilize a simple mahalanobis distance measure to classify candidate pixels as belonging to a certain object. A graph based and structural analysis approach to improve OCR accuracy by removing non-textual information is shown in [7].

Based on the literature review, the current research poses a two-fold problem. One is the light computational power it is expected to be, and two and more importantly, the strip based functionality. So much so, that the algorithm is expected to be flexible enough to operate with strip size as a parameter, i.e. it could process anywhere from a strip of a few lines to half a page, to the entire document. Padding by replication will be utilized for the

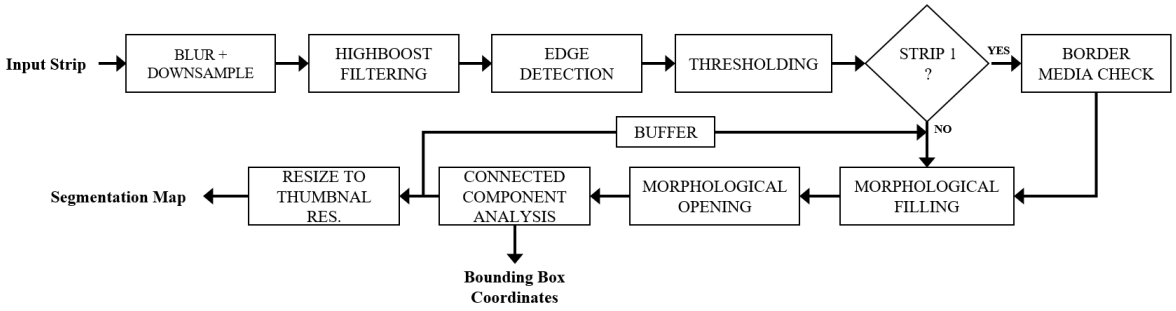

Figure 1. *Framework of the Proposed Media Segmentation Algorithm*

last strip. Another point of note is the desire to not restrict content and media types to a select few, but enable segmentation of vast variety of media shape, size and type. Since the primary objective is not media identification but rather segmentation and thereafter separation, emphasis was placed on concentrating on the edges of the media. Additionally, since the research goal stems from the need to separate scanned receipts, a requirement is imposed that the overarching shape of the object be maintained and extracted.

To pursue the aforementioned goals, and with the desire to implement the algorithm on a target system with limited resources, the concept of using pre-processed data was employed followed by a simple scheme of edge detection, morphological operations and connected component analysis.

The remainder of this paper is organized as follows; the following section goes over the proposed algorithm and all the associated details. Results and observations are drawn in the section after, and finally, Conclusions and Future Research are discussed in the final section.

Proposed Algorithm

The Media Separation framework can be broadly divided into two main modules, namely, Media Segmentation and File Creation. The media segmentation process (Fig. 1) takes in the input image strip and returns the segmentation map and bounding box parameters. The first two steps can be considered as part of pre-processing the image data. After running a series of edge kernels over the high pass filtered image, a hard threshold is applied to the gradient magnitude. Then morphological processing results in a binary image that consists of full contiguous regions. These regions are then labeled using a connected component analysis technique. This technique operates on each strip while maintaining label information from one strip to another through a single line buffer that is essential for not only the labeling process but the morphological filling process as well. In addition to generating the final segmentation map, the two pass technique employed for connected component analysis generates coordinates for a rectangular bounding box for each object. The coordinates are for the low resolution down sampled image. A functional block that only operates on the first strip and not every first strip is *Border Media Check*. After the segmentation map for the strip is obtained a simple nearest neighbor interpolation scheme is used to resize the strip to the original resolution. The bounding box coordinates are resized to the target resolution during the file creation process.

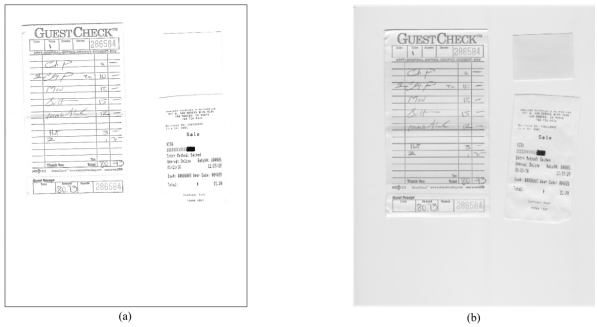

Figure 2. *Post-processed 300 dpi in (a) and pre-processed in (b)*

Pre-Processed Data

As highlighted in the Introduction, the two main challenges facing this research was the strip based approach and designing an algorithm to discern the edges of any types of media. The latter challenge can be further understood when receipt media is scanned. The edges are impossible to distinguish visually and only the content can be extracted. Any sort of fast and computationally light method operating on strips of the image would find it hard to deal with such data. A system based solution was adopted due to the availability of pre-processed data in the scan pipeline. This data is of a lower resolution (around 100 dpi) and is available in monochrome. This demosaicked pixel data is void of any image and color enhancement techniques such as white balance adjustment. As can be seen in Fig. 2, the edges are clearly distinguishable in (b) than in (a). The sticky note is not even visible in the post-processed high resolution data. The lower resolution doesn't affect our process of separation, since the goal is detection and not recognition. The availability of this data in the system allows us to pursue an edge based detection scheme.

Pre-Processing

In the pre-processing stage, the input image data strip is downsampled and high pass filtered. Downsampling by a factor of two allows for a reduced computational load without sacrificing a great deal of performance in the proposed edge based algorithm. The image data is blurred using a mean filter prior to the downsampling in order to reduce aliasing and spread the edge information over a small neighborhood. The integral image technique is used for mean filtering in place of convolution to speed up this process and reduce complexity. The size of mean filter kernel

was determined experimentally based on segmentation results. It was set at 3. The high pass filtering was a simple unsharp masking scheme with the blurring done through a mean filter (integral image implementation) rather than a gaussian. The parameters for this were set at 3 for the filter size and 5 for the scale factor K to add the difference back to the original. These values, in particular K were determined experimentally. A high scale factor can result in increased high frequency noise, which is already inherent in pre-processed data.

Edge Detection

Multiple edge kernels are utilized to generate a gradient map. For each kernel, the absolute value of the result is added together to form a total gradient magnitude image. A hard threshold is then applied to the magnitude image resulting in a binary image that is ready for morphological processing. Initially, only horizontal and vertical sobel kernels were used. However, due to the need for strongly linked edges and closed boundaries for each scanned object for the filling process, additional edge and line detection kernels were utilized. These kernels are shown below in Fig. 3. This results in an improved segmentation performance at the cost of time and computation. Each kernel is independently convolved with the image strip.

$$\begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$
(a)

$$\begin{bmatrix} 0 & 1 & 2 \\ -1 & 0 & 1 \\ -2 & -1 & 0 \end{bmatrix} \begin{bmatrix} -2 & -1 & 0 \\ -1 & 0 & 1 \\ 0 & 1 & 2 \end{bmatrix}$$
(b)

$$\begin{bmatrix} -1 & -1 & -1 \\ 2 & 2 & 2 \\ -1 & -1 & -1 \end{bmatrix} \begin{bmatrix} -1 & 2 & -1 \\ -1 & 2 & -1 \\ -1 & 2 & -1 \end{bmatrix}$$
(c)

$$\begin{bmatrix} -1 & -1 & 2 \\ -1 & 2 & -1 \\ 2 & -1 & -1 \end{bmatrix} \begin{bmatrix} 2 & -1 & -1 \\ -1 & 2 & -1 \\ -1 & -1 & 2 \end{bmatrix}$$
(d)

Figure 3. *Horizontal, vertical and diagonal edge kernels in (a) and (b), similar line kernels in (c) and (d)*

Morphological Processing

The morphological processing stage consists of filling and opening. The latter process is equivalent to a low pass filtering process that removes noise that stems from dust and other tiny non-uniformities that come with using pre-processed data (Fig. 4b). The size of the structural element for the opening process was set to 11 experimentally. The opening process allows no modifications to the closed uniform object regions generated by the filling process.

Morphological Filling

The filling process takes a binary edge magnitude map that consists of non-flat content in the object. As is shown in Fig. 4a and b, in order to allow for seamless labeling, the content within the detected object boundaries needs to be uniform. This will allow for the region within each object to be a connected component of each other and allow for a single label for that object. The filling process is implemented through the technique of morphological reconstruction as is shown in [8]. The complement of the input image is used as a mask and the its border pixels initialize the marker. A 3 x 3, 4 connected (NSEW) structuring element dilates the marker at each iteration. This continues until stability is achieved, which in this case, is achieving the same image result in two successive iterations. A tuneable paramater that effects the time in which the process converges is the size and shape of the structuring element. This is explored along with other options/methods in the results section.

One of the requirements of the above described filling process is to have closed boundaries for each of the detected object regions. Using a strip based methodology, the entire object may or may not be available when processing the current strip. This is overcome by padding the left, right and bottom of the binary strip with 1's (or 255's depending on the convention). This allows for regions as shown in Fig. 4 to be filled appropriately. In order to continue the filling process for the next strip that may include parts of the objects in the current strip, the last line of the strip is saved in a buffer and used as padding for the top of the next strip. The padding described here works in all cases except for the first strip when there are objects touching the top of the scanned image. When a similar padding approach is used, i.e. left, right and bottom, the resulting image is not what is expected due to incomplete boundary data. This is further described in the next section.

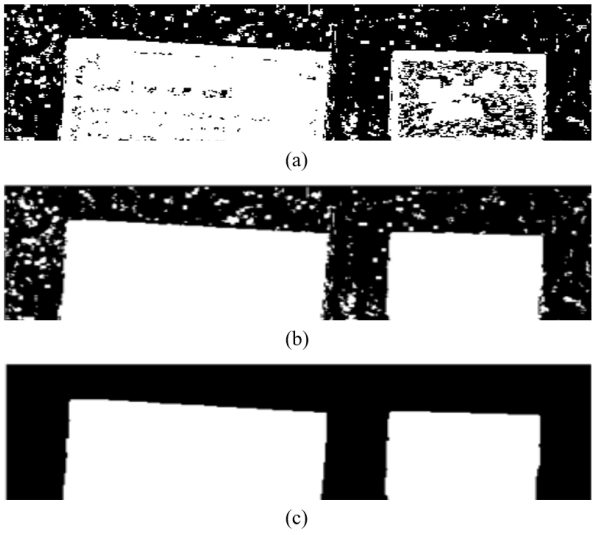

Figure 4. *For a given strip, Binary gradient magnitude in (a), morphologically Filled in (b) and morphologically opened in (c)*

Border Media Check

When items are placed on the flatbed to be scanned, they are often lined up with the edges of the device to ensure they are straight and aligned. Media touching the top of the scanned area poses problems for the segmentation algorithm as the filling process cannot accomplish creating contiguous regions. In order to overcome this, a check is performed for the first strip. This entails checking for top border content using the first 5 lines of the strip and a vertical projection profile. The resulting energy, i.e. squared sum of the projection profile determines if border media is present when compared to a specified threshold (127,500). If the condition holds, implying presence of media near the top, then the left, right and top of the binary gradient thresholded strip is padded with 1's/255's. A regular morphological filling process is performed. This leads to the objects to be unfilled at the bottom, but relatively filled at the top. The first line of the image strip gives us information of media borders for the first strip. This

in turn guides the filling process resulting in correctly segmented media.

Connected Component Analysis

The distinct regions with reduced noise in the background obtained from the morphological process are primed to be labeled. This is done using a simple two-pass connected component algorithm scheme. During the first pass, each pixel is examined from left to right, top to bottom and assigned a label starting from 1. A pixel under consideration is assigned the same label as the pixel to the left, top-left or above it. If there is no labeled pixel in the aforementioned neighborhood, the label index is increased by 1 and is assigned to the pixel. In the case a pixel has neighbors with different labels, the lower of the two labels is assigned and the two labels are deemed equivalent. During the second pass, these equivalence's are resolved. The labels on the last line of the strip are propagated through the buffer to the next strip and utilized to maintain the continuous labeling scheme. Additionally, during the second pass, when a pixel is assigned a final label, it is compared against a global list of maximum-minimum coordinates for each object. This is used to create a bounding box for each media entity.

File Creation

After all the strips have been processed, the segmentation map along with the bounding box coordinates are used to create separate files. For each object, the bounding box coordinates are transformed to the upsampled space, and the target object is extracted. Additionally, the segmentation map is utilized to remove any objects not belonging to the object being saved. To that effect, while the images is being written, the background and foreign objects are bleached and set to the peak pixel value. This value is arbitrary and can be set to any value or pattern. It should be noted here, that the segmentation map and bounding box coordinates can alternatively be transformed into any target high resolution space based on the application.

Results, Discussion and Optimization

The entire algorithm was implemented in both MATLAB and C. Test data was acquired through a device in which pre-processed data could be obtained. A set of 17 documents (Fig 5). were obtained that consisted of varying media types such as receipts, regular paper, card stock paper, business cards, ID cards, photographs and even thick items such as keys. In order to simulate the strip based technique, the 100 dpi pre-processed letter size scans are broken up into strips prior to each being sent to the framework as illustrated in Fig. 1. Performance was evaluated in terms of both quality of results (segmentation maps) and speed (processing time per strip). For the latter, the algorithm was tested on both a PC (3.5 GHz 8 Core Processor and 16 GB RAM) as well as a *BeagleBone Black* (1 GHz AM3358, 512 MB RAM, -O3 optimization). The *BeagleBone Black* was chosen as it is representative of the target system that this algorithm is intended to operate on.

Varying Strip Size: Qualitative Analysis

Strip sizes of 119 and 62 were tested along with processing the entire page. The results are shown in Figs 6, 7 and 8. For a strip size of 119, there is one incorrect segmentation in contrast to a strip size of 62 that had 3 incorrect segmentation maps. However, a strip size of 62 correctly segmented the document that the prior one missed. This implies that the incorrect segmentation, resulting from a weekly detected edge point, depends on the location of strip boundaries. The page based processing results attest to this, albeit on a limited test set, where it correctly segments all the documents provided. It also can be deduced that a smaller strip size leads to issues when dealing with border media. This can be due to the fact in which the padding for the border me-

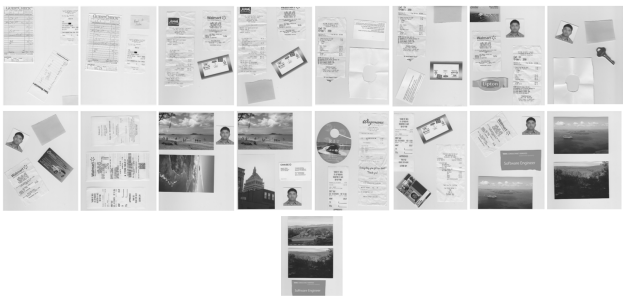

Figure 5. *Set of 17 test documents obtained for evaluation*

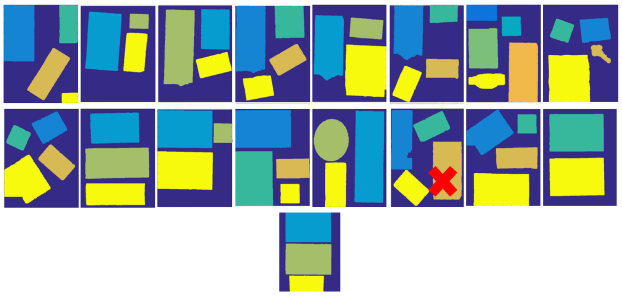

Figure 6. *Segmentation results with a strip size of 119*

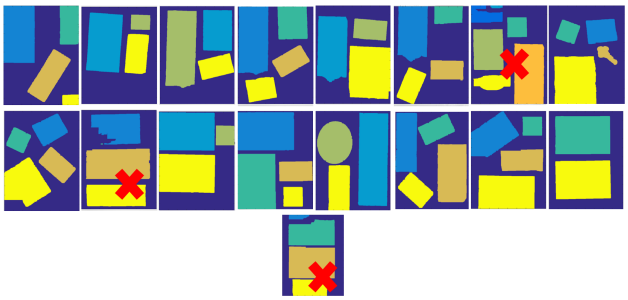

Figure 7. *Segmentation results with a strip size of 119*

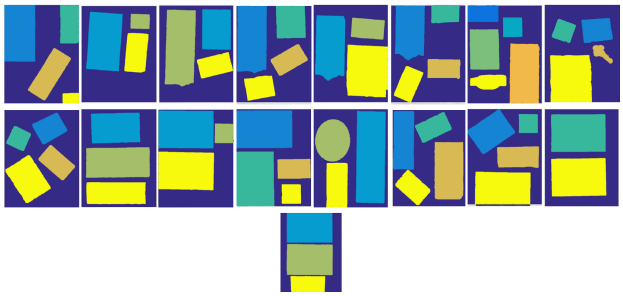

Figure 8. *Segmentation results for processing the entire page at once*

dia is detected. For a small strip size, the *border media check* filling process doesn't have enough content to work as expected. File creation depends entirely on the segmentation results, since the bounding box coordinates for each detected object depends on the created map. The results are promising for the strip based and especially for the page based schemes. The strip boundaries, a single line buffer and a global pre-defined threshold contribute to the failures in the strip based scheme.

Timing Analysis: PC

The timing analysis for the proposed algorithm was done for the media segmentation aspect only. The initial strip simulation and file creation were not included in the performance numbers since they are more system dependent and are variable based on implementation and output file types. With a controlled strip size of 119 the average processing time per strip on the PC was 175.77 ms. Before, moving to the *BeagleBone*, the timing is broken down by each module and analyzed and shown in Fig. 9. Based on this, for the 3 x 3 SE, Border Marker, the Morphological filling process appears to be taking the longest time, which is expected due to its iterative nature. In order to improve the processing time for our target system, steps are taken to optimize the morphological filling process.

Optimization for Morphological Filling

In order to improve the speed of the filling scheme, the number of iterations needs to be reduced. This was achieved in two steps. The structuring element used for the geodesic dilation in the reconstruction process was increased in size from a 3 x 3 to a 5 x 5 NSEW connected. Additionally, as was highlighted by Hassan et al in [9], the marker that is being operated on is initialized differently. The initial implementation described in the algorithm used a marker that is equal to the complement of the image on the border only and zero elsewhere. This is termed as a border maker. In order to reduce the number of iterations, the marker is instead initialized to be close enough to the objects to be detected. Since, this cannot be known prior and the noise on the image due to dust could be the start of an object, the marker is set to 1 from the second line (excluding the buffer) going down until an ON pixel is reached in the original binary image for each column. The resulting image is called a background maker. Fig. 10 shows the initialized markers for each case for the given strip.

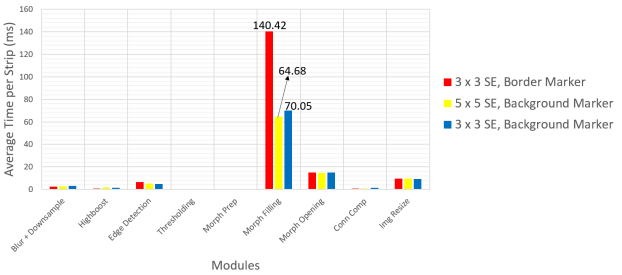

Figure 9. *PC timing analysis breakdown for each module*

The above process reduces the number of iterations for a given strip from around 72 to 45. However, when this was tested on the test document set, incomplete segmentation of objects were seen in 4 of the 17 documents. This is an increase of 3 incorrect results for the fixed strip size of 119. This can be attributed to the larger neighborhood of the 5 x 5 structuring element and its tendency to miss an edge pixel with its NSEW structure. However, when the size of the element was changed back to 3 x 3, the results went back to being in agreement with the non-optimized case. In terms of performance improvement, as is seen in Fig. 9, the 5 x 5, background marker reduces the filling time from 140.42 ms to 64.68 ms on average. This increases to 70.05 ms for a 3 x 3, background marker, but gives the superior result qualitatively. All in all, performance is improved from 175.77 ms average time per strip, to 105.15 ms.

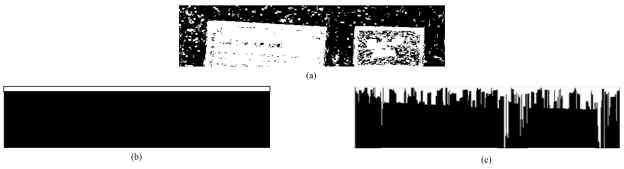

Figure 10. *For the given input strip in (a), the border marker is shown in (b) and the background marker is shown in (c)*

Timing Analysis: Beagleboard

The Beagleboard results in increased processing times due to its low computational prowess. The average time taken per strip is 338.48 ms for a strip size of 119. When the processing times for each document is analyzed, the time appears to be proportional with the PC. The breakdown of performance for each module on the Beagleboard (Fig. 11) shows that the filling process takes the longest time (around 203.33 ms) The next biggest bottleneck is the opening, which is the case with the PC as well. However, one interesting diversion from the PC trend is that of the blur + downsampling modules that takes a relatively longer time (23.14 ms versus 3 ms on the PC) to run on the beagle board. In contrast the edge detection process appears to perform better on the beagleboard than the PC, posting an average time of 4.11 ms compared to a 4.67 ms for the latter. This outcome needs to be investigated further and could be related to the -O3 optimization used when compiling the C code on the embedded system. When processing the entire page at once, the algorithm had an average run time of 17.35 s on the beagleboard. Further optimizations could result in the page based processing times being reduced significantly.

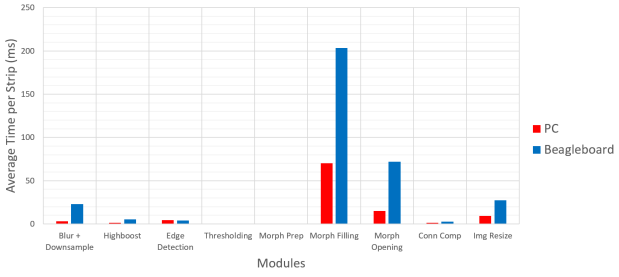

Figure 11. *Beagleboard timing analysis breakdown for each module*

Conclusions and Future Work

A media separation algorithm targeting a scanner system was proposed. The algorithm operates in a strip based fashion, lever-

aging pre-processed data in the scan pipeline. Edge detection and morphological processing are used to detect the objects, along with a two pass connected component analysis scheme that labels the data and also generates bounding boxes that enables easy individual file creation. The algorithm was developed to operate on any size strip or the entire page and uses a single line buffer to propagate label information from one strip to the other. Experimental results were promising for different strip sizes on a limited test document set. However, the best results were seen on the algorithm operating on the entire page. Furthermore, the morphological processing, in particular the filling, was optimized to improve performance with room for further improvement in terms of algorithmic implementations. Future work includes incorporating an adaptive non-global thresholding scheme that would lead to less errors in the filling process. Additionaly, further testing to see if increasing the buffer size improves segmentation can also be valuable. Also, comparing as well as incorporating machine learning based methods for object detection could improve as well as enhance the framework while sacrificing performance time and memory requirements. On the validation side, acquiring a larger test data set with more types of media and varying orientations will be necessary in future development cycles.

References

[1] Faisal Shafait, Daniel Keysers and Thomas Breuel, Performance Evaluation and Benchmarking of Six-Page Segmentation Algorithms, IEEE Trans on Pattern Analysis and Machine Intelligence, 30, 6 (2008).

[2] Shriram Revankar and Zhigang Fan, Picture, Graphics and Text Classification of Document Image Regions, Proc. SPIE 4300, Color Imaging: Device-Independent Color, Color Hardcopy, and Graphic Arts VI. (2000).

[3] M. Sezer Erkilinc, Mustafa Jaber, Eli Saber, Peter Bauer, Dejan Depalov, Text, photo, and line extraction in scanned documents, Journal of Electronic Imaging, 21, 3 (2012).

[4] Patrick Chiu, Francine Chen and Laurent Denoue, Picture detection in document page images, Proc. 10th ACM Symposium on Document Engineering, pg. 211. (2010)

[5] Ming-Wei Lin, Jules-Raymond Tapamo and Baird Ndovie, A texture-based method for document segmentation and classification" South African Computer Journal, 36, (2006)

[6] Madhur Srivastava, Satish K. Singh and Prasanta K. Panigrahi, A semi-automated statistical algorithm for object separation, Cirucits, Systems and Signal Processing, 32, 6 (2013).

[7] F. Zirari, A. Ennaji, S. Nicolas and D. Mammass, A document image segmentation system using analysis of connected components, Proc. of 12th International Conference on Document Analysis and Recognition, pg. 753 (2013).

[8] Rafael C. Gonzalez and Richard E. Woods, Digital Image Processing 3rd Ed., Pearson Prentice Hall, Upper Saddle Creek, NJ, 2008, pg. 660

[9] Mokhtar M. Hassan and Pramod K. Mishra, Improving morphology operation for 2D hole filling algorithm, International Journal of Image Processing, 6, 1 (2012).

Author Biography

Osborn de Lima received his BS in Electrical Engineering from Christian Brothers University (2010) and his MS in Electrical Engineering from the Rochester Institute of Technology (RIT)(2013). He is currently pursuing his PhD in Imaging Science from RIT. His research work has focused on the development of document image processing algorithms for content separation, identification and retrieval where he has been funded by corporate partners. He is a member of IEEE and IS&T.