The ISL Document Image Analysis Toolbox

Richard Rogers, Jisheng Liang, Robert M. Haralick, and Ihsin T. Phillips
Intelligent Systems Laboratory
Department of Electrical Engineering, Box 352500
University of Washington
Seattle, Washington 98195
rrogers@cs.washington.edu

Abstract

This paper describes the Document Image Understanding Toolbox currently under development at the University of Washington's Intelligent Systems Laboratory. The Toolbox provides a common data structure and a variety of document image analysis and understanding algorithms from which Toolbox users can construct document image processing systems. An algorithms for font attribute recognition based on the image analysis techniques available in the toolbox ISL DIU Toolbox is also presented.

1 Introduction

The goal of document image analysis is to transform document images into a hierarchical representation of their structure and content. Since document image analysis techniques will be moving to the consumer market, they have to perform nearly perfectly. This means that they have to be proved out on significant sized data sets and there must be suitable performance metrics for each kind of information a document analysis technique infers. Many of the current systems have a sequence of modules and corresponding knowledge for a specific type of document, and hardly any show how to choose appropriate algorithms for different kinds of documents with various format, content and condition.

In the Intelligent Systems Laboratory (ISL) at the University of Washington, we are developing a document image analysis toolbox, including a collection of data structures and algorithms to support a variety of applications. An experimental environment has been built to allow developers to develop, evaluate and optimize their algorithms. The appropriate and quantitative performance metrics and evaluation protocol have been developed. The architecture allows for convenient experimentation to evaluate the performance of different algorithms and sequences of modules. A series of document image databases have been created for this purpose. We have constructed a prototype of the system and demonstrated its flexibility and functionality on different applications.

This paper is organized as follows. In Section 2, a mathematical model of the document structure is discussed and the problem statement of document recognition is formulated based on the model. The experimental environment is described in Section 3. The algorithms in the toolbox and the corresponding performance measures are discussed in Section 4. The specific architectures constructed for different applications are presented in Section 5.

2 Document Structure Model

We define a hierarchical structure, called a *Polygonal Spatial Structure,* to capture all of the information about a document image. A polygon and the divider around it is called a Polygonal Spatial Structure (PSS). A basic Polygonal Spatial Structure, which is not further divided, carries a content, and the nature of the divider. A composite Polygonal Spatial Structure consists of one or more non-overlapping Polygonal Spatial Structures which are either basic or composite Polygonal Spatial Structures. We denote by Σ the alphabet consisting of symbols and images which can be the content of PSS. We denote by $\mathcal{D}$ the set of dividers (spacing, ruling, etc.).

The Polygonal Spatial Structure is built-up from the following basic sets and mappings.

Content Type

We denote by Θ the set of physical types (text-block, text-line, word, table, equation, drawing, halftone, handwriting, etc.). We denote by Γ the set of functional types (section, paragraph, word, title, heading, caption, abstract, author, list item, footnote, page number, etc.). We denote by $\mathcal{C} = \Theta \times \Gamma$ the set of all possible content types. Each content type has an associated physical and functional type.

Polygonal Area

We denote by $\mathcal{A}$ the set of non-overlapping homogeneous polygonal areas on document image. Each polygonal area $A \in \mathcal{A}$ consists of an ordered pair (θ, I), where $\theta \in \Theta$ specifies the physical type of content and I is the area. A polygon is homogeneous if all its area is of one physical type and there is a standard reading order for the content within the area. Two polygons are physical *adjacent* if each has a significant length of a side which near parallel nearby, and are separated by a divider.

Content

$O : \mathcal{A} \rightarrow \Sigma$ associates polygonal areas of the PSS with their contents. $M : \mathcal{A} \rightarrow \Gamma$ associates polygonal areas of the PSS with their functional types of content.

Format Attribute

We denote by $\mathcal{F}$ the set of format attributes (font type, font size, font style, justification, indentation, etc.). $S : \mathcal{C} \rightarrow \mathcal{F}$ specifies the format attributes for each type of content.

Location

We denote by $\mathcal{L}$ the set of qualitative locations (top left, bottom, middle, etc.). $\mathcal{P} \subseteq \mathcal{C} \times \mathcal{L}$ specifies the "preferred" locations of different types of content.

Spatial Relation

$T : \mathcal{C} \times \mathcal{C} \rightarrow \mathcal{D}$ specifies the divider used between different types of content, i.e. inter-paragraph spacing, inter-line spacing, inter-column spacing, the spacing between a figure and its caption, etc.

Reading Order

Let $\mathcal{A} = \{A_1, \cdots, A_K\}$ be the set of non-overlapping polygonal areas. The reading order R is a tuple $(r_1, \cdots, r_K)$ which is a permutation of $(A_1, \cdots, A_K)$.

2.1 Document Structure and Document Analysis Problem

The document structure consists of layout structure, logical structure, style and content. The document analysis problem is to generate the "most likely" document structure from a document image.

Layout Structure

A layout structure of a document image is a specification of the geometry of the polygons, the content types of the polygons, and the spatial relations of these polygons. Formally, a layout structure is $\Phi = (\mathcal{A}, \mathcal{D})$, where $\mathcal{A}$ is a set of homogeneous polygonal areas, and $\mathcal{D}$ is a set of dividers.

Logical Structure

Logical Structure extraction involves assigning functional labels to each polygon of the page, and ordering the polygons according to their read order. Formally, a logical structure is $\Psi = (M, R)$, where M associates polygonal areas of the PSS with their types of content, and R is the reading order.

Content

The content is O which associates polygonal areas of PSS with their contents.

Style

The page style is $\Omega = (S, T, P)$, where S specifies the format attributes for each type of content, T specifies the divider used between different types of content, and $\mathcal{P}$ specifies the "preferred" locations of different types of content. The page style describes the formatting properties for polygons with different content types.

3 Experimental Environment

It is clear that much of the early work on document analysis system provided illustrative results and hardly any had their techniques tested on significant sized data sets and to measure quantitative performance [5]. The main reasons are the lack of accurate document ground truth to train and test the algorithms and the lack of appropriate and quantitative performance metrics and evaluation protocol. A standard interchange document structure representation is necessary for developers to exchange the training and test images and share the algorithms. The experimental environment should be flexible enough to support the performance characterization of different modules and the different specific system architectures, such as top-down, bottom-up or hybrid.

3.1 Data Representation

Our document analysis toolbox uses the Document Attribute Format Specification (DAFS) to represent the document structure. DAFS [13] has been developed as a document interchange format to encode decomposed documents and to allow representation of both the physical and logical information contained within a document image. The DAFS provides a format for breaking down documents into standardized entities. DAFS entities are conveniently defined objects within a document such as a paragraph or word. An entity can have content, which might be the text it encompasses; and properties, such as bounding box, font and point size. DAFS permits the creation of parent, child and sibling relationships between entities, providing easy representation of the hierarchical structures of a document. Thus DAFS is a suitable structure for representing a Polygonal Spatial Structure (See Section 2). Each polygonal area of PSS is represented by an entity of DAFS. Through DAFS, developers will be able to exchange training and test documents and work together on shared problems.

3.2 Meta-architecture

The toolbox consists of a collection of document recognition routines. Each routine uses DAFS entities for input and output. The user can create a configuration file to specify where to look for information regarding a given entity, and which recognition routine to use. Parameters which may be needed during recognition are passed as properties of the entity or the entity type (See Figure 3.2). The user can write custom recognition routines if the ones provided do not suit the needs, then use the configuration file to associate the new recognition routines with entity types, read in parameters and set properties. Specific document recognition architectures, such as top-down, bottom-up, hybrid, or iterative, can be defined in the configuration file. This allows for convenient experimentation to determine the best algorithm for each step and the best sequence of modules for an application, and to estimate the algorithm parameters given the training data.

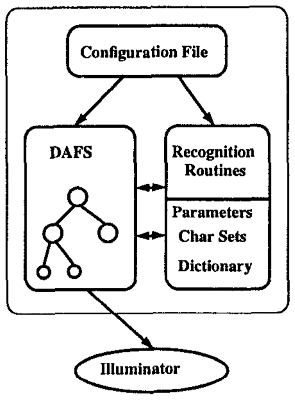

Figure 1. illustrates the environment architecture. Illuminator is the editor of DAFS files.

The configure file lists the specific algorithm and the type of entity the algorithm works on for each step. The following is an example of configuration file that specifies a sequence of processes for the analysis of journal article document:

Document	Sequence	
<Step# >	<Entity-Type>	<Recognizer-Name>
step1	Page	ZoneSegment
step2	Zone	ZoneClassify
step3	Zone	LineSegment
step4	Line	WordSegment
step5	Word	TextRecognizer
step6	Line	BlockSegment
step7	Page	Markup

3.3 Performance Evaluation of Document Analysis Algorithms

A performance evaluation needs a performance metric, ground-truth data, and an algorithm to match the output representation of document analysis algorithms with the ground-truth representation (See Figure 2). Statistical and display tools are also needed to help users categorize errors and analyze the cause of errors [6].

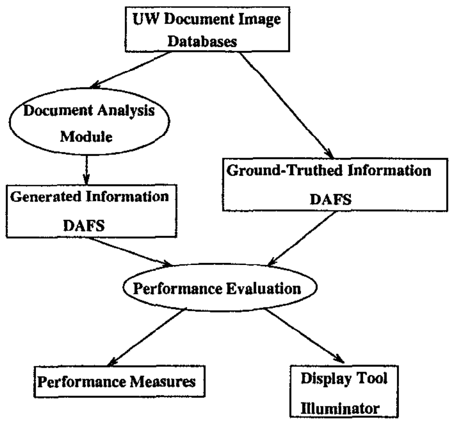

Figure 2. illustrates the process to compare the detected information to the ground truth.

For each structure that we use to describe a document, there is an associated metric that measures the difference between a structure that is automatically produced and the ground-truth structure. DAFS structure representation is chosen as the standard representation of the ground truth data and the results of document analysis algorithms.

UW-III [12] is the third in a series of UW document image databases [11]. It contains a total of 1600 English document images randomly selected from scientific and technical journals. The documents consist of accurately ground-truthed layout and logical structure, style, and content. Each page contains a hierarchy of manually verified page, zone, text-line, and word entity with bounding box and in the correct reading order. The text ground-truth and style attributes are tagged to each zone entity. This database can be utilized by the OCR and document understanding community as a common platform to develop, test and evaluate their systems. Based on the ground-truth data, we can evaluate the performance of document analysis algorithms and build statistical models to characterize various types of document structures. A set of appropriate and quantitative performance metrics are discussed in the following sections.

4 Algorithms in the Toolbox and the Performance Measures

The document recognition toolbox consists of the following routine libraries: layout analysis, logical structure analysis, style detection, text recognition (OCR), markup, image processing, graphical model (Bayesian network) facilities, and other utilities.

4.1 Layout Analysis

The layout analysis discovers various objects of interest $\mathcal{A}$ in an input document image. An object is a homogeneous region in a document image that corresponds to one type: character, word, text line, text block, text or non-text zone. The algorithms for the sub-problems of layout analysis are described in the following sections. The papers that discuss the details of the methods can be found in the references. Each algorithm in the layout analysis toolbox has its advantage and limitations for documents with different layout and conditions. Therefore it is necessary to characterize these algorithms by evaluating their performance on different kinds of document. From the characteristics of algorithms and document images, we should be able to determine the algorithms that work best for a given document set and a given task. The free parameters of the algorithms can be estimated for the different documents if the ground truth is provided. Currently, we are using the default parameter values that are generated by heuristics or from a small set of training samples [10].

4.1.1 Performance Evaluation of Layout Analysis

To evaluate the performance of the layout analysis, the detected entities must be compared with the ground truth entities.

Evaluation of Segmentation

Suppose we are given two sets $\mathcal{G} = \{G_1, G_2, \cdots, G_M\}$ for
ground-truthed entity boxes and $\mathcal{D} = \{D_1, D_2, \cdots, D_N\}$ for detected entity boxes, comparison of $\mathcal{G}$ and $\mathcal{D}$ can be made in terms of the following two kinds of measures:

$$\sigma_{ij} = \frac{\text{Area}(G_i \cap D_j)}{\text{Area}(G_i)} \; and \; \tau_{ij} = \frac{\text{Area}(G_i \cap D_j)}{\text{Area}(D_j)}$$

where $1 \leq i \leq M$, $1 \leq j \leq N$, and Area(A) represents the area of A. The measures in the above equation constitute two matrices $\Sigma = (\sigma_{ij})$ and $T = (\tau_{ij})$. Notice that σ_{ij} indicates how much portion of G_i is occupied by D_j, and τ_{ij} indicates how much portion of D_j is occupied by G_i. The possible errors of misdetection, false alarm, splitting, merging are detected for each entity by analyzing these matrices. The DAFS files which include the correctly detected entities and the differences between ground-truthed and detected data are generated respectively.

Evaluation of Classification

The classification module classifies each extracted entity into one of the predefined categories according to its physical content type. The output of the classification process is compared with the labels from the ground truth in order to evaluate the performance of the algorithm. A contingency table is computed to indicate the number of entities of a particular class label that are identified as members of another class. The mis-classification rate can be computed from these numbers. Let $P(t, a)$ be the probability of observing a unit whose true category is t, and whose assigned category is a. The mis-classification rate is defined as,

$$P(mis - classification) = \sum_{t \in \Theta} \sum_{a \in \Theta,\; a \neq t} P(t, a),$$

where Θ is the set of content types.

4.1.2 Algorithms in the Layout Analysis Toolbox

Zone Segmentation

A zone entity is a rectangular area that consists of homogeneous data (only one physical content type). A text zone is constrained to a single column of text and there is only one reading order for the content within the zone. A document image may be segmented using the recursive *X–Y* cut based on bounding boxes of connected components [4]. This method is tested on UW-III database with a total of 1600 pages. Table 1 illustrates the percentage of correct, splitting, merging, miss, and spurious detections with respect to the ground truth. Of the 24,243 ground truth zones, 87.18% of them are correctly detected.

Table 1. Performance of zone segmentation with respect to the ground truth.

	Correct	Splitting	Merging	Miss	Spurious
X-Y cut	87.18%	3.62%	5.96%	1.08%	2.15%

Zone Classification

Given a homogeneous zone entity, zone classification module classifies it according to its content. We have developed a method using feature vector generation and classification to classify each given scientific and technical document zone into one of the eight labels: text of font size 8-12, text of font size 13-18, text of font size 19-36, display math, table, halftone, line drawing, and ruling [9]. We have tested our method on UW-I document image data set with 979 pages and a total of 13,726 zones. The mis-classification rate of the algorithm for all zone types is 5%. The mis-classification rate for text and non-text distinction is 3%.

Text-line Segmentation
The layout analysis toolbox includes three different methods to extract text-line entities: the extracted word entities are grouped into lines based on a Probability Linear Displacement Model [2]; the text zones are segmented into lines by cutting the projection profile of connected component bounding boxes [4]; the connected components are grouped into lines by merging and splitting the connected component bounding boxes [8]. The algorithms are tested on UW-III database with a total of 105,439 text-lines. The percentage of correct, splitting, merging, miss, and spurious detections for three algorithms are shown in Table 2.

Table 2. Performance of text line segmentation with respect to the ground truth.

	Correct	Splitting	Merging	Miss	Spurious
PLDM	96.49%	1.41%	2.03%	0.01%	0.07%
Projection	94.78%	0.12%	4.78%	0.28%	0.04%
Group c.c.	97.56%	0.53%	1.26%	0.60%	0.06%

Word Segmentation
Two different word segmentation methods are provided in our tool box: the extracted text-lines are segmented into words based on the vertical projection profile of connected component bounding boxes within the text-line [4]; the black pixels are merged into word using recursive morphological closing transform [2]. These methods are tested on UW-III database with a total of 828,201 words. The percentage of correct, splitting, merging, miss, and spurious detections for two algorithms are shown in Table 3.

Table 3. Performance of word segmentation with respect to the ground truth.

	Correct	Splitting	Merging	Miss	Spurious
Projection	97.83%	0.63%	1.36%	0.16%	0.03%
Morphology	79.22%	2.30%	12.18%	5.68%	0.63%

Text-block Extraction
A text-block is a text entity that can be assigned a functional label (paragraphs, section heading, captions, etc.). Given a set of text-line entities, text-block extraction module is the process to merge text-lines into text-blocks. The current algorithms to extract text-blocks are: grouping text-lines and making text-blocks by analyzing the alignment of neighboring text-line bounding boxes; characterizing the text-block structure based on the augmented Probabilistic Linear Displacement Model [2]. These methods are tested on UW-III database with a total of 21,738 text blocks. Table 4 illustrates the percentage of correct, splitting, merging, miss, and spurious detections for two algorithms.

Table 4. Performance of text block segmentation with respect to the ground truth.

	Correct	Splitting	Merging	Miss	Spurious
Alignment	75.64%	7.42%	15.10%	0.00%	1.83%
APLDM	72.96%	6.50%	15.24%	0.04%	5.26%

4.2 Style Detection

The font style analysis problem is to determine the font attributes (size, angle, weight, serif type, spacing, etc.) of each glyph image present in a document. The document analysis toolbox contains an algorithm for estimating a glyph image's weight and angle. The algorithm uses a Bayesian network implemented the toolbox's prototype graphical model facilities to classify a glyph image as bold or normal weight and upright or oblique angle based on features computed from the glyph's stem.

The stems are extracted by recursively opening the glyph image by a vertical line structuring element. The opening is then thresholded to obtain the stem subimage. The threshold value is determined as follows:

- Let h be the height of the glyph image.
- Let v_k be the image of the tallest connected component of the thresholded opnened image using a threshold of k.
- Let h_{v_k} and w_{v_k} be the height and width of v_k, respectively.
- Let n_{v_k} be the number of horizontal runs in v_k.

The threshold is chosen as the minimum k such that

$$\frac{n_{v_k}}{h} < \theta_1,\ \frac{h_{v_k}}{h} > \theta_2,\ and\ \frac{h_{v_k}}{w_{v_k}} > \theta_3$$

$\theta_1 \approx 1$ ensures that v_k contains a single stroke. $\theta_2 \approx 0.8$ ensures that v_k is tall relative to the glyph. $\theta_3 \approx 2.5$ ensures that the stroke is fairly narrow.

Once the stem image is extracted from the glyph, the following features are computed:

- d is the number of black pixels in the stem image divided by the area of the stem image bounding box (in pixels) rounted to a single decimal place and multiplied by 10.
- r is the aspect ratio of the stem image bounding box rounded to the nearest integer. Values above 10 are clipped to 10.
- t is the number of black pixels in the stem image divided by sn, where s is is the font size in points and n

is the number of horizontal r runs in the stem image. t is rounded to a single decimal place and multiplied by 10. Values above 12 are clipped to 12.

- c is the glyph's stem class, as defined in Table 5.

The stem class groups characters with stems of approximately the same shape and size. Classes 1 and 2 are distinct in order to handle fonts in which upper-case vertical strokes are wider than those of lower-case letters.

Table 5. Character stem classes.

Class	Members
1	BDEFIJKLMNPRTU
2	bdfhjklpq
3	aimnrut
4	AVW
5	vw
6	CGOQ
7	ceo
8	SXYZgsxyz

The angle and weight estimation problem is to find a glyph image's angle a and weight w to maximize $\Pr(a, w|d, t, r, c)$. The joint distribution of these variables is represented by a Bayesian network learned from 4732 ideal glyph images generated from TEX's Computer Modern fonts. The glyph image's weight and angle are estimated by entering d, t, r, and c as evidence and performing a max calibration on the network. The algorithm correctly determines the glyph image's weight for 93.7% of the glyphs, and correctly determines the angle for 87.2% of the glyphs.

4.3 Logical Structure Analysis

The logical structure analysis consists of the following sub-problems: text entity labeling (paragraph, title, section heading, caption, etc.); and reading order determination.

The logical labeling module assigns each entity a functional label. For each correctly extracted entity, we compare the automatically assigned label with the ground-truth label. A contingency table is computed to indicate the number of entities of a particular class label that are identified as members of another class. The mis-classification rate is used as the performance measure of logical labeling.

Let $\mathcal{A} = \{A_1, \cdots, A_K\}$ be the set of entities which have been correctly identified. The detected reading order $\hat{R}$ is a tuple $(r_1, \cdots, r_K)$ which is a permutation of true reading order $R = (A_1, \cdots, A_K)$. The problem of evaluating reading order determination algorithm can be reduced to computing the "edit distance" between true reading order R and the output of the algorithm $\hat{R}$. The edit distance is the minimum number of editing operations, such as copy, cut, paste, etc., required to change $\hat{R}$ to R.

4.4 Text Recognition

The extracted text entities are sent to OCR engine to recognize the text content. The problem of evaluating the OCR algorithm can be reduced to the string matching problem between true symbol strings $<T_1, T_2, \cdots, T_x>$ and the output of the OCR algorithm $<O_1, O_2, \cdots, O_y>$. The OCR Performance Evaluation (OPE) Software [3] is provided in UW-I document image database. The program compares the ground-truth and OCR output and output a file containing: single character contingency table, error substring contingency table, statistics of line insertions, deletions and substitutions, and statistics of symbol insertions, deletions and substitutions.

4.5 Automatic Markup

Given a document structure and content, and a specification of formatting attributes (style), markup module converts the document structure into a formatted document in a desired file format. The file format depends on the application, i.e., SGML for document interchange, RTF for editing and document reconstruction, HTML for hyperlinking, and PDF for document archiving.

5 Example Applications

For different applications, specific document recognition architectures can be constructed by specifying the algorithm to use on the hierarchy at each step. The toolbox contains several different algorithms for each document analysis task. Different algorithms may work best for documents with different characteristics. The tool box configurability allows users to create a document analysis system by choosing algorithms appropriate to the documents they work with. The following are three specific architectures which demonstrate the flexibility of the document analysis tool box.

Document Reconstruction

This architecture (Figure 5) converts scientific journal pages to Rich Text Format (RTF) [14] files, which can be edited by word processors. The layout analysis produces a set of text and non-text entities. The logical structure analysis assigns each text entity a functional label and detects the reading order. An OCR engine is called to recognize the content of text entities. The DAFS to RTF conversion converts the DAFS file to an RTF file. The user can define the page style by editing the configuration parameter file or the generated RTF file. One can use Microsoft^{TM} Word^{TM} to reconstruct or edit the document from the generated RTF file. This allows access to the original document contents in a form that is unconstrained by the original physical structure.

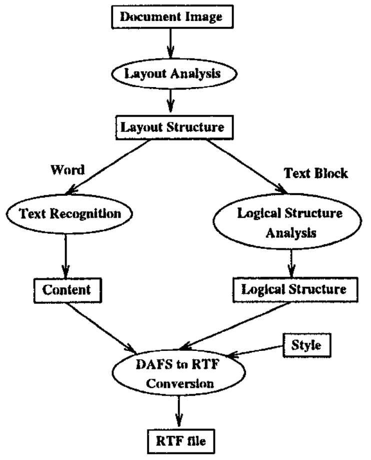

Figure 3. illustrates the architecture for journal article reconstruction.

Document Archiving

This architecture (Figure 5) converts the document pages to Portable Document Format (PDF) [1] files. It is very similar to the function of Adobe^{TM} Acrobat^{TM} Capture^{TM} product. A word segmentation routine is called to extract

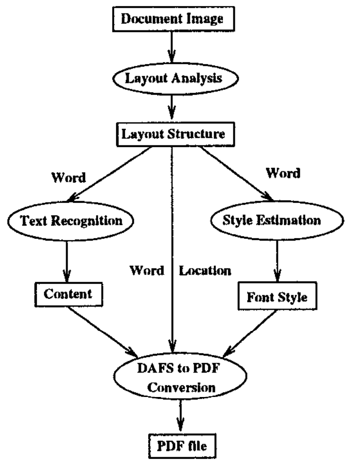

Figure 4. illustrates the architecture for document archiving.

word entities from the input document image. Then an OCR engine is used to recognize the words, and the font style of each word is estimated by a font detection routine. DAFS to PDF conversion module converts the generated DAFS file into a PDF file. The recognized content and font attribute of text entities are used to build the PDF objects which retain the exact location of these text entities. The non-text entities and unrecognizable text entities are output as PDF image objects. PDF enables users to easily and reliably exchange and view the electronic document independent of the environment in which they are created.

Mail Piece Address Recognition

This architecture (Figure 5) selects the destination address from a scanned mail piece to facilitate automated mail routing.

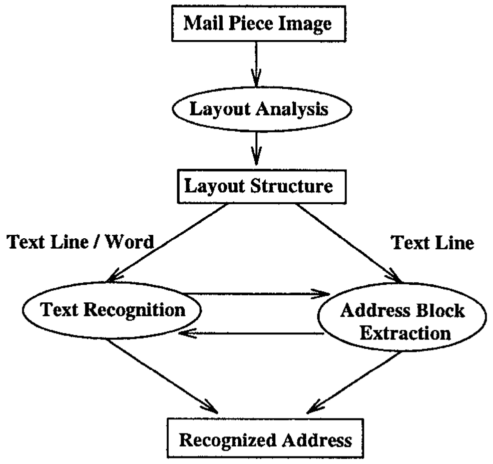

Figure 5. illustrates the architecture for address recognition on the mail pieces.

The text-line segmentation finds text lines in input image. Then the text lines are grouped into zones and the address block location module selects the zone containing address. The text recognition module is called on the extracted address block. The context information, such as the city, state and zip code, can be used for both the address recognition and location.

6 Summary

The objective of our research is to develop an experimental environment to support the design and evaluation of document analysis algorithms and systems. A document analysis toolbox is being developed to provide a collection of algorithms to support scanned document recognition. We have implemented the basic functionality of each module described above. A system prototype has been constructed and demonstrates the flexibility and functionality of the toolbox. We are currently working on the parameter tuning and performance characterization of the algorithms provided in the toolbox. The toolbox will provide graphical model facilities for developing application specific analysis and decision algorithms. The new recognition and control algorithms are being developed to improve the accuracy, efficiency, and robustness of the system on a broad range of documents.

References

[1] Adobe. *Portable Document Format.* 1995.

[2] S. Chen. *Document Layout Analysis Using Recursive Morphological Transforms.* Ph.D. thesis, Univ. of Washington, 1995.

[3] S. Chen. *OCR Performance Evaluation Software User's Manual.* ISL Report, E.E. Dept., U. of Washington.

[4] J. Ha, R.M. Haralick, and I.T. Phillips. *Document Page Decomposition using Bounding Boxes of Connected Components of Black Pixels.* Document Recognition II, SPIE Proceedings, vol. 2422, pp 140–151, San Jose, February 1995.

[5] R.M. Haralick. *Document Image Understanding: Geometric and Logical Layout.* Proceedings 1994 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, pp. 385-90, 21-23 June 1994.

[6] J. Kanai, T.A. Nartker, S V. Rice, and G. Nagy. *Performance Metrics for Document Understanding Systems.* Proc. 2nd ICDAR, pp 424–427, Japan, 1993.

[7] T. Kanungo and R.M. Haralick. *Automatic Generation of Character Groundtruth for Scanned Documents: A Closed-Loop Approach.* Proceedings of ICPR'96, pp 669-675, 1996.

[8] J. Liang, J. Ha, R. Rogers, B. Chanda, I.T. Phillips, and R.M. Haralick. *The Prototype of A Complete Document Understanding System.* Proc. IAPR Workshop on Document Analysis Systems, pp 130-154, 1996.

[9] J. Liang, I.T. Phillips, J. Ha, R.M. Haralick. *Document Zone Classification Using the Sizes of Connected Components.* Proceedings of the SPIE, Vol 2660, Document Recognition III, pp 150–157, San Jose, 1996.

[10] J. Liang, R.M. Haralick, and I.T. Phillips. *Performance Evaluation of Algorithms in ISL Document Layout Analysis Toolbox.* ISL Technical Report, University of Washington, 1996.

[11] I.T. Phillips, S. Chen and R.M. Haralick. *CD-ROM English Document Database Standard.* Proc. 2nd Int. Conf. on Document Analysis and Recognition, pp 478–483, Japan, 1993.

[12] I.T. Phillips. *User's Reference Manual for the UW English/Technical Document Image Database III.* UW-III English/Technical Document Image Database Manual, 1996.

[13] RAF Technology, Inc., *DAFS: Document Attribute Format Specification.* 1995.

[14] Microsoft, *Rich Text Format (RTF) Specification.* 1994.