Name and Address Block Reader System for Tax Form Processing

Sargur N. Srihari, Yong-Chul Shin, Vemulapati Ramanaprasad and Dar-Shyang Lee

Center of Excellence for Document Analysis and Recognition
State University of New York at Buffalo, Buffalo, NY 14260, U.S.A.
{srihari,ycshin,raman-v,dslee}@cedar.buffalo.edu

Abstract

The reading of names and addresses is one of the most complex tasks in automated forms processing. This paper describes an integrated real-time system to read names and addresses on tax forms of the Internal Revenue Service of the United States. The Name and Address Block Reader (NABR) system accepts both machine-printed and hand-printed address block images as input. The application software has two major steps: document analysis (connected component analysis, address block extraction, label detection, hand-print/machine-print discrimination) and document recognition. Document recognition has two non-identical streams for machine-print and hand-print; key steps are: address parsing, character recognition, word recognition and postal database lookup (ZIP+4 and City-State-ZIP files). Real-time throughput (8,500 forms per hour) is achieved by employing a loosely-coupled multiprocessing architecture. The functional architecture, software design, system architecture and the hardware implementation are described. Performance evaluation on machine-printed and handwritten addresses are presented.

1 Introduction

The Internal Revenue Service (IRS) of the United States receives around 200 million tax forms per year. Only ten percent of these returns are filed electronically. Although efforts are under way to increase the use of electronic filing, IRS still has to contend with paper returns for years to come. In an effort to convert the paper returns into electronic form, IRS plans to use a new, intelligent optical character recognition system called *Service Center Recognition/Image processing System (SCRIPS)*.

In this paper, we describe the design of a subsystem called *Name and Address Block Reader (NABR)*, which is one of the components in the character recognition and identification subsystem of SCRIPS. NABR is an integrated system which has been developed at the *Center of Excellence for Document Analysis and Recognition (CEDAR)*, State University of New York at Buffalo. It combines postal address recognition techniques developed at CEDAR over the past decade and current computing power available in the market.

The primary objective of NABR is to recognize each character within the address on a tax form. Recognition is aided by contextual information in the form of various databases, such as city-state-ZIP Code database, street-name database, and personal-name database.

The software and hardware design of NABR posed some special challenges, principal of which were: a) wide range of quality of machine-printed text, b) lack of reliable segmentation and recognition techniques for hand-printed characters, c) demands on processing speed and recognition accuracy, d) size of databases involved (about 1.2 gigabytes compressed), and e) need to provide a cost-effective solution.

The input to the NABR system consists of a name and address block image, and other pertinent information, such as form type and field locations, constructed externally to the NABR. Depending on tax forms, a maximum of three name and address blocks can be put in an input packet. The input image is a bit-packed binary image in TIFF captured at 200 ppi. The address can be either hand-printed or machine-printed. A machine-printed address can be a labeled address (a labeled address is one in which the address is pre-printed on a label and later pasted on top of the tax form) or a typewriter printed address. Some tax forms may have drop-out ink to distinguish between pre-printed text and the actual address. The current NABR system is designed to read 15 different IRS tax forms including 1040EZ, 1040PC, 941 and IRPS forms. Examples of hand-printed and machine-printed address blocks from several of the tax forms are shown in Figure 1. In the label image of Figure 1(b), each of the five lines are handled as follows. The top line, carrier sort line (********* 5-DIGIT 14260) is to be ignored. The next four lines are to be read. Each of the lines is referred to as follows: the first line is the scan line (HZ 123-45-6789 S08 B1), followed by the name line (JOHN J DOE), street address line (123 MAIN STREET), and the city-state-ZIP line (ANYTOWN NY 14260). An output packet contains processed ASCII information including character recognition results, database access status, and other corrected information associated with a set of confidence values.

2 Address Recognition Software

The overall control flow of the NABR system is shown in Figure 2. The detailed design of the modules resembles the design used in postal address reading systems developed at CEDAR, for reading either handwritten or machine-printed addresses [1].

2.1 Document Analysis

This involves recognizing the type of address in the input image (hand/machine print, label/no-label) and then segmenting it into individual lines. Figure 3 shows the subtasks in document analysis. If the image is that of a label then the task is to extract that portion of the image corresponding to the label. If the tax form is not printed with drop-out ink then the task is to remove all the pre-printed text from the image.

0-8186-7128-9/95 $4.00

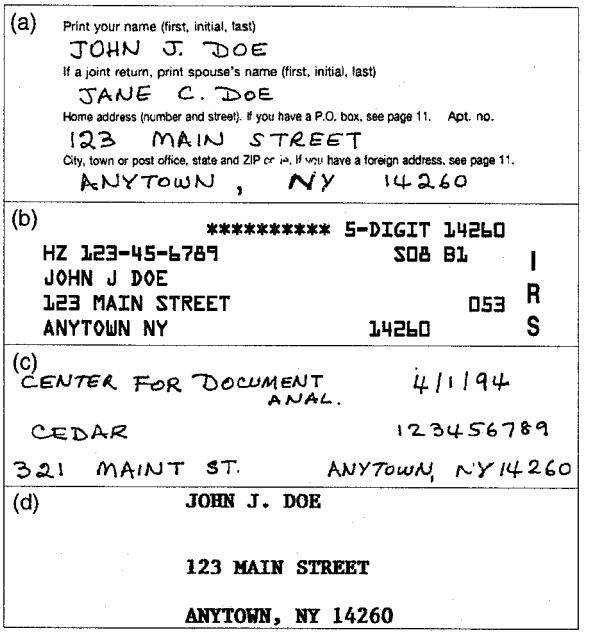

Figure 1: Examples of images input to NABR: (a) 1040EZ hand-printed with non-drop-out ink, (b) 1040EZ label with OCR-A Font, (c) 941 hand-printed with drop-out ink and (d) 1096 machine-printed with drop-out ink.

2.1.1 Connected Component Analysis

Document analysis operations can be speeded-up by performing the operations at the connected component level, rather than at the image level. Thus the first step is to generate connected components of the entire input image. There are several algorithms to generate connected components. We use the Line Adjacency Graph (LAG) method to generate and represent connected components.

2.1.2 Address Block Extraction/Label Detection

If a tax form is not pre-printed in drop-out ink (this depends on the type of form being processed), then certain pre-determined registration marks are used to delete the pre-printed text. These registration marks correspond to areas of high pixel density in the address block. By examining regions around these registration marks, we determine whether there is pre-printed text to be deleted or whether a label has been pasted on top of the tax form.

In case of tax forms that have drop-out ink, the spacing between lines is used to determine the presence of a label. In case of a labeled image the spacing is much smaller than in the case of a non-labeled image. The detection of the presence of a label facilitates parsing the lines into name line, street line and ZIP Code line because of the fact that non-labeled images have fixed parsing.

2.1.3 Hand-print/Machine-print Discrimination

It is necessary to determine whether the address is hand-printed or machine-printed, since different character segmentation and recognition algorithms are used for each script type. The hand/machine print discrimination is performed using six symbolic features extracted from the connected components: standard deviation of connected component widths, standard deviation of connected component heights, average component density, aspect ratio, distinct different heights, and distinct different widths.

The classifier employed is a Fisher's linear discriminant. The result is a fast, high-performance discriminator: on a test set of 800 postal addresses, the correct discrimination rate was 95%.

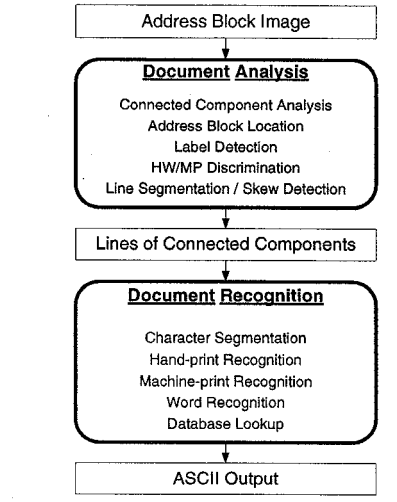

Figure 2: Top-level Modular Organization

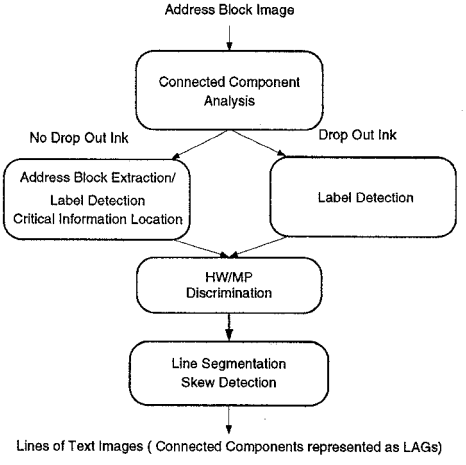

Figure 3: Document Analysis.

2.1.4 Line Segmentation/Skew Detection

One approach to line segmentation is to use a vertical histogram. This method can be very slow as it is pixel-based. Also histogram method will fail badly if the input image is skewed. To overcome these two problems, a clustering algorithm that works directly on the connected components of the image, is used. This decreases the processing time for line segmentation. Also vertical overlap between adjacent components is used to cluster the components into lines. The skew is computed as the clusters are determined. The clustering criterion (vertical overlap) makes the algorithm more robust to skew in the input image.

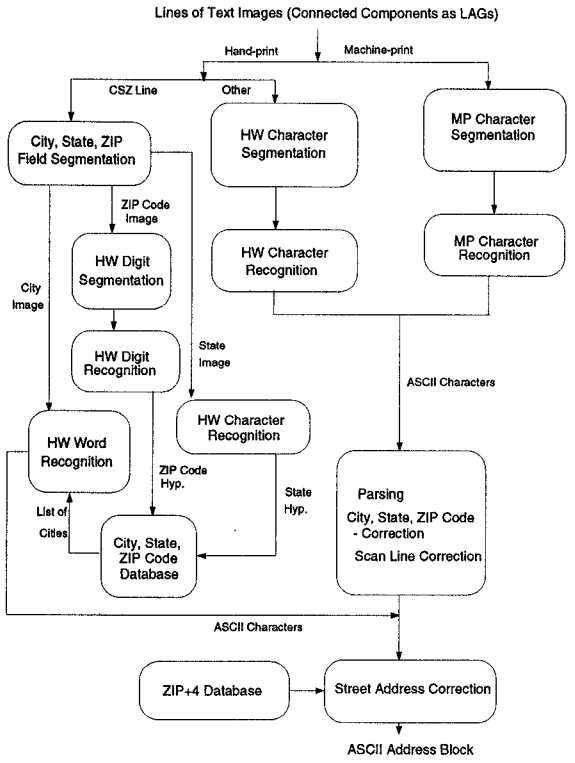

Figure 4: Document Recognition

2.2 Document Recognition

After preprocessing, we have lines of connected components. We also have an indication whether the image is hand-printed (HP) or machine-printed (MP). Depending on the type of the image (HP or MP), different segmentation algorithms are used. The process flow for character segmentation and character recognition is shown in Figure 4.

2.2.1 Machine-Print Segmentation

It was found that most of the machine-printed addresses on the tax forms have fixed-pitch font. This information is very useful in breaking touching characters as it makes the segmentation task trivial. The fixed-pitch font determination is done by examining the spacing between the connected components that have similar width and height as that of normal characters and also by looking at the word gaps.

2.2.2 Hand-Print Segmentation

The hand-print segmentation algorithm is based on an iterative row-partitioning of the image, where each partition consists of a set of contiguous rows. The number of rows in a partition decreases for each iteration. At each iteration, overlapping partitions are grouped into a blob. Each of these blobs is tested with a character recognizer. If the recognition yields a high confidence character, then that blob is removed in the next iteration. After the last iteration (when the number of rows per partition is one) each blob corresponds to a connected component. At this stage each of these blob can contain one or more character. An estimation is made about the approximate number of characters (n) in each of the blobs remaining. Character splitting techniques are used to split the blob into either $n-1$, n or $n+1$ characters. The decision to choose either $n-1$, n or $n+1$ characters is made based on the combined character confidences.

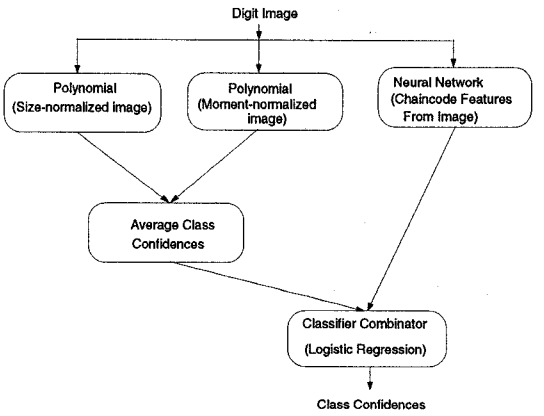

Figure 5: Hand-printed digit recognizer

2.2.3 Character Recognition

Three basic OCR engines are used for character recognition, each OCR engine specifically trained for a different purpose (machine-print, hand-printed alphabets, and handprinted digits) [2]. All three engines use an artificial neural network classifier consisting of one hidden layer, trained using backpropagation. A special OCR-A engine is based on a two-layer neural network. The OCR-A and handprint digit modules consist of a combination of different algorithms.

The *training dataset* for machine-print (98,308 images) was from tax forms and postal envelopes. The data sets for handprinted alphabets (113,694 images) were from various sources, including NIST data sets. The OCR-A engine was trained on 3,620 images. The data set for handprinted digits was from a database created at CEDAR from postal envelopes.

OCR Modules. The basic features are contour (or gradient) features. The contour direction for each edge pixel is calculated by indexing to a lookup table with its 3x3 neighborhood. The basic algorithm calculates the chaincode direction for each pixel by convolving a kernel over the image; the actual chaincode sequence is not constructed. The feature vector is of length 64. Architectures of the network vary in different modules: with 64 input units, 150 hidden units and 71 output units for machine-print, and 64-100-71 for hand-print.

Hand-printed Digit Recognition. Since recognition of handprinted addresses relies heavily on accurately reading the ZIP Codes, a different algorithm is used for handwritten digit recognition. A combination of two algorithms are used (Figure 5). The first one is based on chaincode features. Directions and curvatures at each point on the digit contour are first calculated. Then a histogram similar to the one described above is constructed for each image partition to form a feature vector of 296 dimensions. A multilayer perceptron is used for classification. The neural network contains one hidden layer (296-100-10 units).

Polynomial Classification. The second recognizer uses the polynomials of pixels in a normalized image as features.

Classifier Combination. To take advantage of the orthogonalities between these different classification ap-

proaches, a combinator is trained to combine individual recognition results to improve performance. The combination method used is a logisitc regression model.

2.2.4 Parsing and Contextual Post-Processing

At this stage, we have lines of words/characters with OCR results. The objective of the postprocessing module is to correct OCR results using contextual information. Parsing involves classifying each line into City, State, ZIP line or Address Line or Name Line and also identifying the most frequently used words in a postal address (street, south etc.). Contextual constraints imposed by postal directories, such as the City-State-ZIP Code database and the ZIP+4 database are useful to correct OCR results.

2.2.5 Parsing/Word Category Assignment

Before contextual postprocessing can be performed, the OCR results have to be parsed. There are two levels of parsing. One is *vertical parsing*. Vertical parsing involves classifying each line as name line, street line, CSZ (city, state, ZIP Code) line, etc. Vertical parsing is done by using spatial constraints and the location of each line with respect to the others. *Horizontal parsing* involves labeling each word in a line with appropriate tags. Tags are used to denote special types of words that occur in postal addresses (e.g., street number, street name, ZIP Code, street suffix, etc.). Horizontal parsing is accomplished by comparing each word in a line with a set of keywords that occur very frequently in postal addresses (e.g., South, North, Mr., Street, etc.). Each keyword has a set of tags associated with it (e.g., North can be a personal name or a street prefix). When a word in a line matches with a keyword then that word is labeled with the tags associated with the matched keyword. From this list of tags, an appropriate tag is chosen based on the vertical parsing information (i.e., NORTH in name line will be a family name and in address line it will be labeled as street prefix).

Once an unambiguous horizontal parsing has been done, the databases are used to correct OCR results. The ZIP Code database is used to correct city names and the city name database to correct ZIP Codes and state names. A database of all street names in the entire country is used to correct street names, assuming that the ZIP Code and street number are recognized correctly.

2.2.6 City, State and ZIP Code Postprocessing

City-State-ZIP database is used to correct the OCR results of the CSZ line. This database is a comparatively small database (about 5 MB) and resides entirely in main memory. It supports three types of queries: a) Given a ZIP Code, return all (City, State) pairs. This is the fastest of the three queries b) Given a city name substring, return all city names which contain *exactly* those characters in the substring. This query is slightly slower than the first one, and c) Given a city name substring, return all city names which *contain* the substring. This is the slowest of all queries.

2.2.7 Street Address Postprocessing

The street address database, also called ZIP+4 database, contains street name information of the entire country. ZIP Code and street number are used as indices to get a list of street names from this database. Each of these street names is compared against the OCR results of the recognized street names. If there is a good match then the OCR results are appropriately corrected. The uncompressed ZIP+4 database occupies 4GB of memory space. In order to reduce the storage requirements, the database is compressed, at the same time keeping the database access reasonably fast [3].

Only one type of query is provided for database access: given a street number and a ZIP Code, return all the corresponding street names. The query takes place as follows: a) obtain pointer to global information from ZIP Code table (basically a disk block offset and length) b) fetch global information from the disk, and use street number to calculate the block which contains the specific information about the street number, and c) fetch the block containing the street number data, and decompress the block to find street name information corresponding to the specified street number.

2.2.8 Word Recognition

We use a lexicon-based word recognizer which generates a set of segmentation points for each handwritten word such that a character contains atmost four segments and no segment contains more than one character. The recognizer uses the lexicon and a character recognizer to group the individual segments into characters. This lexicon-based approach is possible in case of CSZ and street line. In case of the CSZ line, the ZIP Code is recognized first. Using this ZIP Code, a set of possible cities are extracted from the database. This set of cities is passed to the word recognizer as a lexicon along with the image corresponding to the city (Figure 4). Using a word-model word recognizer we obtained an average of 55% of field correct rate on the CSZ line as shown in Table 1. Similarly one can use ZIP Code and street number to get a set of possible street names. These names can then be used as a lexicon to recognize street names. However, extracting the street name image from the street line is nontrivial, as this involves accurate recognition of street suffix, street prefix, etc.

3 Multiprocessing System Architecture

In order to meet the required processing performance and speed goals, a multiprocessing scheme was chosen as the NABR system architecture [4][5]. In this architecture, each of several processors executes identical copies of the address recognition software. A system controller (SC) distributes each successive input image to an available processor. The architecture is illustrated in Figure 6.

The entire system is designed to be scalable for any application. Additional computing power can be employed to meet demanding application by simply adding more ARPs in the chassis, without rewriting the control structure.

3.1 System Controller

Overall system activities are monitored and coordinated by the SC utilizing a shared memory in the SC's memory space. Other components of the system notify their status through the shared memory. Also, the SC can probe each subsystem to detect any failure. The SC can reboot any subsystem selectively based on the probing status. The SC is responsible for distribution of input images to the address recognition processors. If no available processor is found when a new input image is ready, the SC should be able to find a candidate processor to kill a stale job.

3.2 Address Recognition Processors

The main processing power of the system is based on four address recognition processors (ARPs) which form a

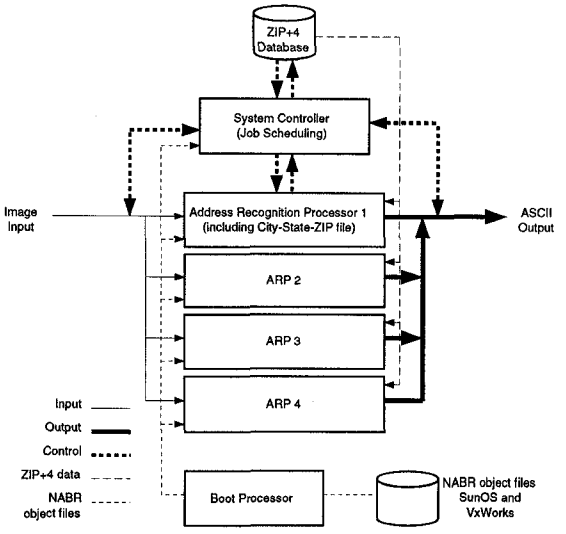

Figure 6: The architecture of NABR system

loosely-coupled multiprocessor system. There are two levels of tasks in each ARP. The upper level task is activated when there is no input image to process, and hence the processor is in an idle state, waiting for a start command from the system controller. Once an input image is transferred and stored in a shared memory space on an ARP, the SC issues a start command to the ARP. The start command is recognized by the upper level task and a lower level task is spawned accordingly. The lower level task performs the reading of name and address. Once the lower level task is spawned successfully, the upper level task is put in a sleep mode, monitoring minimal status flags. The upper level task is activated again after the lower level task has been completed.

While the lower level task is activated, a task-delete command can be issued by the SC based on scheduling. The lower level task is deleted upon receiving a task-delete command and the processor becomes free for the next input image.

3.3 Hardware Implementation

The NABR system was implemented using commercial off-the-shelf boards on a VME backplane. The system consists of six SPARC engine boards, three Transputers, one SCSI host adapter, two database disks, one system disk and one tape drive.

Among the six SPARC engine boards, four are microSPARC based CPU boards with 47.3 MIPS of computing power for ARPs, two are SPARC2 based CPU boards with 28.5 MIPS – one is for SC, and another for boot processor. The SC and the ARPs are running real-time operating systems VxWorks 5.0.2 and 5.1, respectively. The SPARC2 based SC can map its entire memory to VME memory space, whereas the microSPARC based ARPs can only map maximum one megabyte to VME. The one megabyte mapping window is used for input and output. SC maps each of the one megabyte windows (corresponding to each ARP) into its own address space but at different addresses so that it can differentiate each ARP by looking at the corresponding memory space. The boot processor is running Solaris 1.1 (equivalent to SunOS 4.1.3). The ARPs run the same version of the embedded recognition software. For program and local database memory, each board has 64 Mbytes of RAM. The boot processor facilitates system maintenance and provides versatile communication channels with other system components.

Three Transputers are used to provide a versatile interface between the boot processor and the outside SCSI interface for input and output. One is for the host system interface, the next one is for handling input and output buffer, and the last one for SCSI interface. Differential SCSI interface is employed for reliability in noisy environment.

The ZIP+4 database is stored in two disks and accessed through a SCSI host adapter. The tape drive is used to install a new database, to upgrade recognition software, and to backup the system.

When a new input image is fetched and stored in a buffer, this is notified by the SC and a job scheduling is started. Once an available ARP is found as a target, the address of the shared memory space of a target processor is given to the Transputers and the image transfer is initiated. Image transfer operation is performed through the boot processor over the VME interface. The SC is also responsible for arranging the database access.

3.4 Address Databases

The compressed database is stored in one disk (2.5 GB). Another disk is provided to hold backup data. The database is accessed on the fly through a SCSI host adapter. Efficient database query is made to enable a fast access to the database. The queries requested by the ARPs are fetched by the SC to provide a centralized access to a single database. The access initiated by the SC is fetched by the SCSI host adapter and the corresponding disk blocks are read and the retrieved information is directly transferred to the shared memory space of the corresponding ARP.

3.5 I/O Interface

The differential SCSI interface is used for data transfer between the NABR and the SCRIPS' system control and data management subsystem. The differential SCSI signals received by the NABR are converted to the single ended protocol, then read in by a transputer module with a single ended SCSI interface which runs a code necessary for handling SCSI target transactions, so that the SCSI transputer module looks like a tape drive unit to the outside world for all intents and purposes. Another transputer module interacts with the SCSI transputer module so that the data packet is transferred to and received from the SCSI transputer module. The transputer module also provides extra links so that external transputers can access the NABR system. The transputer module is responsible for queuing images and returning result data. When an address recognition processor is available in the system the next available image in the transputer module is transferred through a HSI/Sbus over the VME bus to the appropriate ARP. The data transfer between the HSI/Sbus and a ARP is performed via programmed I/O.

In addition to the data I/O, the NABR system also provides an Ethernet LAN to SCRIPS' system control and data management subsystem, so that the overall activities of the NABR can be monitored remotely.

4 Performance

A set of test images was collected for measuring the performance of the current NABR system. The test set con-

Table 1: Field Correct Rates

	Hand Printed		Machine Printed	
Filed Name	Correct Rate	Total Fields	Correct Rate	Total Fields
SL	n/a	n/a	78.53%	191
QT	n/a	n/a	40.00%	5
EIN	n/a	n/a	100.00%	5
SS	n/a	n/a	97.53%	81
NL	8.89%	90	67.18%	710
AD	15.00%	80	73.95%	645
CT	46.25%	80	92.30%	649
ST	58.75%	80	97.38%	648
ZP	57.50%	80	95.53%	649

sists of 72 hand-printed images and 628 machine-printed images. Each image has a set of filling areas called fields. For the hand-printed images in the test set, there are five fields; name line (NL), street address line (AD), city (CT), state (ST) and ZIP Code (ZP). Some images have multiple NL fields. For machine-printed images, there are additional fields such as scan line (SL), quarter date (QT), employer identification number (EIN), and social security number (SS). Field correctness is defined such that all the recognized character should match to the truth of the field. Any mis-recognized, missing or additional character in the output constitutes an incorrect field. Performance on each field has been measured and summarized in Table 1. Performance on those fields (CT, ST,ZP), where contextual information is used, is much higher than on those fields where contextual information is absent. The performance on street address line is not as high as City, State and ZIP fields because for the street address line to be recognized accurately both the ZIP Code and the street number have to be recognized accurately. Also the same street addresse line can be written in many different variations (like abbreviating (or expanding) the suffix (or prefix), dropping of suffix (or prefix) etc.

Character read rate (CRR) is a performance measure obtained using an edit distance between a set of recognized fields and the corresponding set of truth records. The character read rate can be used for obtaining an estimate of the system performance with respect to any human effort to correct the mis-recognized output. However, the character read rate may not reflect the true character recognition performance of the built-in recognizers, since it might include errors from other tools in the system. In other words, the character read rate can be affected and reduced due to segmentation or parsing errors. For example, a 5-character word may be segmented into a 6-character word and hence the resulting recognition cannot be a success. If the word were segmented correctly, it would get the right recognition. Therefore, another performance measure called *OCR correct rate* is defined by considering only the recognition results with same string length in the truth records. Let P be a list of field pairs, $P_j = (R_j, T_j)$, where $\text{strlen}(R_j) = \text{strlen}(T_j)$ for all j, and $1 \leq j \leq p$. Now the OCR correct rate is defined by $1-(\sum_{j=1}^{p} E_j)/(\sum_{j=1}^{p} \text{strlen}(T_j))$. Table 2 shows overall performance, including the character read rate and

Table 2: Overall Performance of NABR system

	Hand-Printed	Machine-Printed
Average field correct rate	36.59%	84.85%
All-field correct rate	2.50%	47.15%
Character read rate	61.85%	95.08%
OCR correct rate	89.47%	97.99%

the OCR correct rate. The two significant measures in assessing NABR system performance are image correct rate and field correct rate. The image correct rate reflects the number of images that need to go for operator entry. Since the operator entry is on a field basis, the number of fields requiring correction is also a useful measure. The images that go to operator entry are those that do not match a stored database of addresses and social security numbers.

5 Conclusion

We have described an integrated system for document processing, specifically for reading text contained in address blocks of tax forms. The system combines a number of software modules that CEDAR has developed over the past decade, as well as new techniques developed specifically for tax form processing. The NABR system achieves a throughput of 8,500 forms per hour. It has been implemented using a cost effective real-time processing architecture. This architecture, derived from other real-time architectures developed at CEDAR for address-block location and handwritten address interpretation, was the first system that CEDAR deployed in the field. The design can be easily migrated into other similar document analysis applications. Both hardware and software design details have been presented and the performance of the system was discussed. More research on hand-printed image analysis and recognition is necessary for the development of a higher performing system.

References

[1] S. N. Srihari, "High Performance Reading Machines," *Proceedings of the IEEE, 80(7)*, pp. 1120–1132, 1992.

[2] S. N. Srihari, "Recognition of Handwritten and Machine-Printed Text for Postal Address Interpretation," *Pattern Recognition Letters*, vol. 14, pp. 291–302, 1993.

[3] P. Filipski, J. Hull, and S. N. Srihari, "Compression for Fast Read-Only Access of a Large Database," in *Proceedings, the United States Postal Service Advanced Technology Conference*, (Washington, D.C.), pp. 1113–1127, Nov-Dec 1992.

[4] P. W. Palumbo, S. N. Srihari, J. Soh, R. Sridhar, and V. Demjanenko, "Postal address block location in real time," *IEEE Computer*, pp. 34–42, 1992.

[5] J. J. Hull, V. Demjanenko, R. Sridhar, and S. N. Srihari, "Architectural design for address interpretation in a Laboratory Address Recognition Unit (LARU)," in *Proceedings, USPS Advanced Technology Conference*, pp. 1313–1323, November 1992.