A Knowledge Based Segmentation Algorithm for Enhanced Recognition of Handwritten Courtesy Amounts

Karim Hussein
Arun Agarwal
Amar Gupta
Patrick Wang

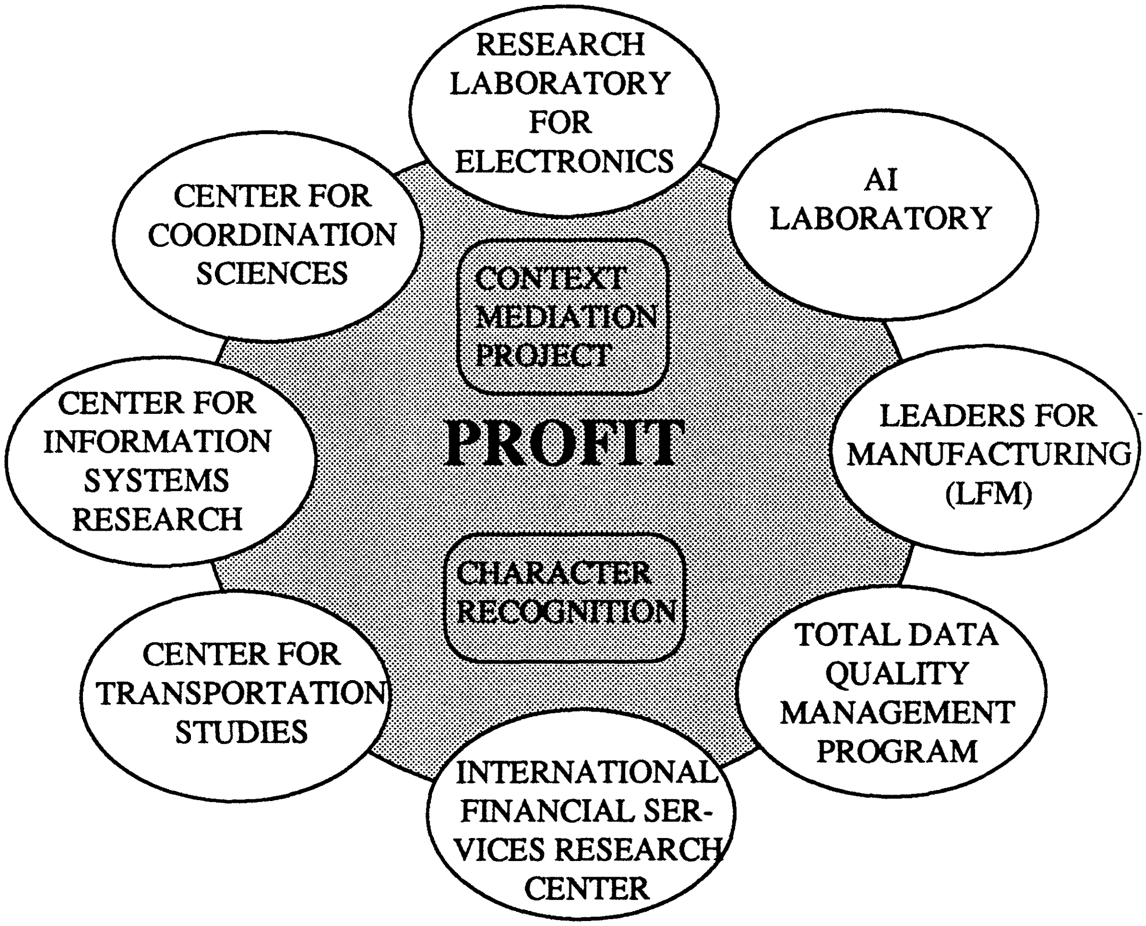
WP #3775 March 1994
PROFIT #94-14

Productivity From Information Technology
"PROFIT" Research Initiative
Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02139 USA
(617)253-8584
Fax: (617)258-7579

 The research described herein has been supported (in whole or in part) by the Productivity From Information Technology (PROFIT) Research Initiative at MIT.

Productivity From Information Technology (PROFIT)

The Productivity From Information Technology (PROFIT) Initiative was established on October 23, 1992 by MIT President Charles Vest and Provost Mark Wrighton "to study the use of information technology in both the private and public sectors and to enhance productivity in areas ranging from finance to transportation, and from manufacturing to telecommunications." At the time of its inception, PROFIT took over the Composite Information Systems Laboratory and Handwritten Character Recognition Laboratory. These two laboratories are now involved in research related to context mediation and imaging respectively.



In addition, PROFIT has undertaken joint efforts with a number of research centers, laboratories, and programs at MIT, and the results of these efforts are documented in Discussion Papers published by PROFIT and/or the collaborating MIT entity.

Correspondence can be addressed to:

The "PROFIT" Initiative
Room E53-310, MIT
50 Memorial Drive
Cambridge, MA 02142-1247
Tel: (617) 253-8584
Fax: (617) 258-7579
E-Mail: profit@mit.edu

EXECUTIVE OVERVIEW

Financial enterprises rely heavily on paper-based documents to conduct various operations; this is true both for external operations involving customers and other financial institutions, as well as internal operations involving various departments.

Researchers at MIT have looked at the possibility of taking information directly from paper documents, especially handwritten documents, to computer-accessible media. Automated reading involves several steps as follows:

(i) Scanning of document;
(ii) Location of area to be "read";
(iii) Decomposing the selected area into separate characters;
(iv) Adjusting size and slant of each character;
(v) Recognizing each character; and
(vi) Testing whether input has been correctly read.

Based on several years of sustained research, the researchers have attained very high "reading" speed and accuracy, even in situations where the quality of the input material is poor. Patent rights for some of the new techniques have been applied for. Sponsor companies are eligible to test the new techniques in their respective environments at no charge.

The work performed so far is described in a number of published paper and working papers. The list of working papers is as follows:

IFSRC # 107-89	Optical Image Scanners and Character Recognition Devices: A Survey and New Taxonomy	Amar Gupta Sanjay Hazarika Maher Kallel Pankaj Srivastava
IFSRC # 123-90R	An Improved Structural Technique for Automated Recognition of Handprinted Symbols Revised October 1990	Patrick S. P. Wang Amar Gupta
IFSRC # 124-90	Integration of Traditional Imaging, Expert Systems, and Neural Network Techniques for Enhanced Recognition of Handwritten Information	Evelyn Roman Amar Gupta John Riordan
IFSRC # 151-91	Handwritten Numeral Recognition Using Dynamic Programming Neural Networks on an Off-Line Basis	Ronjon Nag Alexis Lui Amar Gupta
IFSRC # 162-91R PROFIT 93-03	Algorithms for Thinning and Rethickening Binary Digital Patterns	M. Nagendraprasad Patrick S. Wang Amar Gupta
IFSRC # 173-91	A New Algorithm for Slant Correction of	Vanessa C. Feliberti

	Handwritten Characters	Amar Gupta
IFSRC # 214-92	An Algorithm for Segmenting Handwritten Numeral Strings	Peter L. Sparks M. V. Nagendraprasad Amar Gupta
IFSRC # 215-92	A New Algorithm for Correcting Slant in Handwritten Numerals	M. V. Nagendraprasad Amar Gupta Vanessa Feliberti
IFSRC # 218-92	A System for Automatic Recognition of Totally Unconstrained Handwritten Numerals	M. V. Nagendraprasad
IFSRC # 219-92	A Collection of Papers on Handwritten Numeral Recognition	Amar Gupta
IFSRC # 261-93	An Adaptive Modular Neural Network with Application to Unconstrained Character Recognition	Lik Mui Arun Agarwal P. S. P. Wang
IFSRC # 287-94 PROFIT 93-04	An Integrated Architecture for Recognition of Totally Unconstrained Handwritten Numerals	Amar Gupta M. V. Nagendraprasad A. Liu Amar Gupta S. Ayyadurai
IFSRC # 288-94 PROFIT 93-09	Detection of Courtesy Amount Block on Bank Checks	Arun Agarwal Len M. Granowetter Amar Gupta P. S. P. Wang
IFSRC # 289-94 PROFIT 94-14	A Knowledge Based Segmentation Algorithm For Enhanced Recognition of Handwritten Courtesy Amounts	Karim Hussein Amar Gupta Arun Agarwal Patrick Shen-Pei Wang

The research has been funded by a number of organizations, via the International Financial Services Research Center (IFSRC) and the Productivity from Information Technology (PROFIT) Initiative. Individuals in such sponsor companies should contact their designated contact person at MIT to receive copies of the papers, and the software developed at MIT.

The Principal Investigator for the imaging area is Dr. Amar Gupta, Co-Director, "PROFIT" Initiative, MIT Sloan School of Management, Room E53-311, 77 Massachusetts Avenue, Cambridge, MA 02139, USA; Telephone: (617)253-8906; Fax: (617)258-7579; e-mail: agupta@mit.edu. Your comments and suggestions are encouraged.

A KNOWLEDGE BASED SEGMENTATION ALGORITHM FOR ENHANCED RECOGNITION OF HANDWRITTEN COURTESY AMOUNTS

Karim Hussein
Department of Civil Engineering
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
`hussein@mit.edu`

Arun Agarwal
Department of Computer Science
University of Hyderabad
Hyderabad, India
`aarun@mit.edu`

Amar Gupta
MIT Sloan School of Management
Massachusetts Institute of Technology
Cambridge, MA 02139, USA
`agupta@mit.edu`

Patrick Shen-Pei Wang
College of Computer Science
Northeastern University
Boston, MA 02115, USA
`pwang@ccs.neu.edu`

Abstract

A knowledge based segmentation critic algorithm to enhance recognition of courtesy amounts on bank checks is proposed in this paper. This algorithm extracts the context from the handwritten material and uses a syntax parser based on a *deterministic finite automaton* to provide adequate feedback to enhance recognition. The segmentation critic presented is capable of handling a number of commonly used styles for courtesy amount representation. Both handwritten and machine written numeric strings were utilized to test the efficacy of the preprocessor for the check recognition system described in this paper. The substitution error fell by 1.0% in our early tests.

Keywords: Automata, Check processing, Character recognition, Classification, Knowledge-based, Neural network, Parser, Segmentation, Syntactic.

1. Introduction

Image processing offers the potential for reducing costs involved in processing checks by banks. Traditional check processing involves a series of manual processing steps that impose a heavy financial burden on the banking industry. When a check shown in Figure 1 is deposited for credit into one's account via an automated teller machine (ATM), the depositor's bank is interested in two numerical fields: the account number, which is already written in MICR ink and can be handled using automated

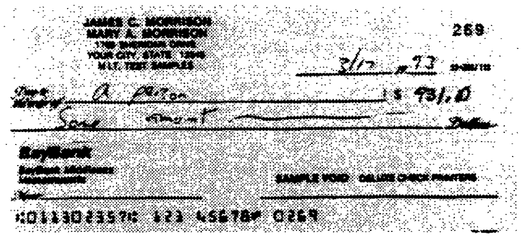


Figure 1: A Typical Bank Check

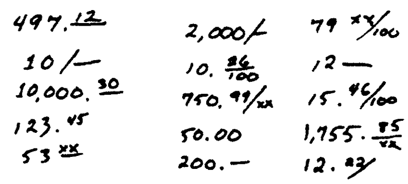


Figure 2: Possible styles for writing courtesy dollar amount

techniques with near-perfect accuracy; and the amount of the check, which is currently read and keyed in by a human operator.

The other fields, such as the name of the recipient, the date, and the signature on the check are largely ignored in routine check processing, unless the transaction is contested or the check is presented for immediate encashment to a bank employee.

The amount of the check is written in two ways: in textual format and in numerical format. The textual format, called the legal amount, was originally intended to be the version used for all transaction related purposes. The numerical format, called the courtesy amount, is the version currently utilized by human operators to key in the amount that appears at the bottom of cancelled checks.

Automated reading of numerical fields has been attempted for a number of application areas. One such area is the reading of postal zipcodes in the addresses written or typed on letters[2–5]. Reading courtesy amounts is more difficult than reading zipcodes due to a number of key differences in the nature of the handwritten material. First, the number of digits in zipcodes is fixed and known a priori, which is not true for courtesy amounts. Second, unlike the case of zipcodes, the courtesy amount consists of two components: the non-fractional component (the dollar portion) and the fractional component (the cents portion). The courtesy component is written in more than a dozen different styles in the U.S., and some of these styles are extremely difficult to handle via automated techniques. Figure 2 shows possible styles. These reasons inhibit the preprocessing of the bank checks on a commercial basis without human intervention.

The focus of this work is to improve the segmentation accuracy of digits by developing a knowledge-based segmentation critic for automated reading of the courtesy amount. Our new algorithm extracts the context available from the typical styles shown in Figure 2 to aid the segmentation and recognition subsystems within the overall check processing system. This paper is comprised of six sections. Section 2

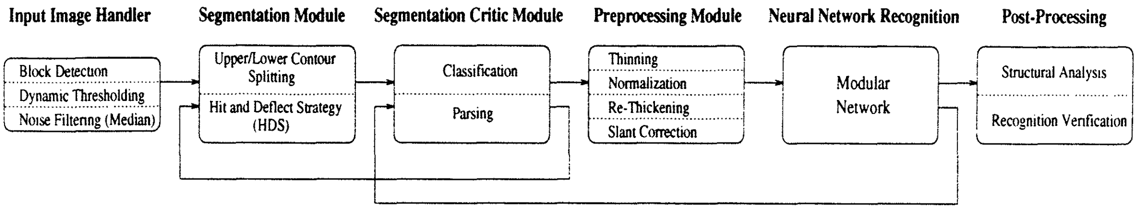


Figure 3: System Architecture

describes the overall check recognition system architecture. Section 3 concentrates on theory and prior work that served as the foundation for our research. Section 4 focuses on the details of the knowledge based segmentation critic module and Section 5 presents results. Section 6 contains the conclusions and discusses areas for future work.

2. Check Recognition System Architecture

Automated recognition can be performed in two environments: on-line and off-line. In the on-line case, recognition is performed as the characters are being written and hence dynamic information like stroke sequence, speed, pressure, and pen-up and/or pen-down positions can be utilized to overcome some of the difficulties created by the non-uniformity and connectivity of unconstrained handwriting[6,7]. In off-line recognition, only static information contained in the image of the numeral strings is available, which makes this enviroment more difficult to tackle. This paper deals with off-line recognition of check courtesy amounts.

The architecture of the prototype system for reading handwritten numerals on checks, shown in Figure 3, consists of six modules and is discussed below.

Image Handler: The position of the courtesy dollar amount on the check varies greatly; hence, a block detection algorithm has been developed to locate and extract the handwritten amount for further processing. Checks in the U.S. are also characterized by wide differences in background textures, writing devices, and colors and shades of the written material. A dynamic thresholding algorithm utilizing the histogram of the input image is used to remove noise and to enhance the clarity of characters. A median filter is then employed to remove any spurious noise left in the image.

Segmentation Module: The binary bitmap of the courtesy amount is passed to the segmentation module which breaks the bitmap into distinct and meaningful pieces in three stages: extraction of connected components, splitting of upper and lower contours, and utilization of the hit and deflect strategy (HDS) for drawing segment cut lines. This segmentation procedure is detailed fully in Sparks[8] and Baird et al.[9]

Segmentation Critic Module: The segmented components are then crudely classified into a list of probable primitives, such as digits and commas. The order of the primitives is analyzed using a parser. Failures in parsing causes the segmentation module to reevaluate its segmentation strategy. This module is discussed in detail in Section 4.

Preprocessing: This stage reduces the variability in the slant and thickness of the various characters. Detailed description can be found in Lam et al[10], Suen and Wang[11], Nagendraprasad et al[12,13].

Neural Network Based Recognizer: The recognition stage consists of an adaptive modular neural network which is trained using a modified back propagation algorithm[14,15].

Postprocessing: When the neural network classifies a character without adequate confidence, the particular digit is passed to a postprocessor, which utilizes traditional pattern recognition techniques to complement the connectionist approach. These techniques include structural analysis described in Duggan[16] and the Freeman Chain Encoding Scheme detailed in Wang and Gupta[17].

All the modules, except the segmentation critic module, were developed previously. This paper focuses on the segmentation critic module which improves the segmentation accuracy using a knowledge-based approach. The details of the module are discussed in Section 4 after some background discussed in the next section.

3. Background Framework

The syntactic and structural approaches for pattern recognition are more popular than conventional statistical methods because they are hierarchical in nature. Moreover, using such syntactic and structural approaches one can directly take advantage of powerful data structures using grammatical rules, trees, and directed labeled graphs, which are widely used in dealing with linguistic problems[18–20]. Further, it has been shown that syntactic and structural approaches can overcome some disadvantages found in the classic decision-theoretic (statistical) approach, which has difficulty in distinguishing between two very similar patterns (characters).

A critical assumption in the pattern recognition techniques described above is that one has provided the specific region or primitive to be recognized. This issue of segmentation or *chunking*, which involves separation of an object from a particular scene or a character from a handwritten string is a non-trivial task. Deciding granularity of segmentation will determine the primitives produced. For example, a given string of connected characters, can be separated using two basic approaches: according to individual characters or according to the strokes that make up these characters. Our approach has been to use a single numeral or delimiter as our base primitive. Several algorithms have been developed to perform this segmentation which are described in Sparks[8]. However the correct segmentation cannot be guaranteed using these bottom-up techniques.

The process of such decomposition (bottom-up) discussed above is inherently narrowly focused. The segmentation and classification processes reach a point where *one cannot see the forest from the trees.* Any successful recognition system must retain a general understanding of the environment within which it operates. Such an environment or context provides additional information concerning the interplay between the classes and the correlations between them. Our aim was to provide such a pattern-directed feedback mechanism to improve our segmentor. In previous work on segmenting numerals, especially work done on postal zip code recognition, the number of digits in a string was known beforehand thereby simplifying the segmentation task. Our algorithm aims to provide an analysis of primitives within a context that has been predetermined (i.e., courtesy amount styles). In reading the courtesy amount on checks, we can only use the fact that certain delimiters are used in particular styles to separate the cents from dollars. This provides a constraint on the cents portion, ie. an indication of the length of the string. The *comma* delimiter also provides an indication of how many numerals are in the dollar amount. These are the constraints that are encoded by our segmentation critic. However, a major difficulty is that the most common delimiters, the comma and period, could also be noise. Therefore, initial noise filtering is critical for the proper functioning of our segmentation critic.

We have used a syntax-based classification approach[23], which permits a large set of complex patterns to be described using small sets of simple pattern primitives and grammatical rules, and successive portions of the input pattern to be classified on a recursive basis. This approach starts with the entire string of input characters and attempts to partition the problem into subgoals (and corresponding subsets of characters) and proceeds until either the last subgoal has been attained or exhausted. This provides a natural way of hypothesizing the global properties of a configuration at an early stage of the recognition procedure.

We have developed and incorporated this paradigm into segmentation critic described below to improve the segmentation process by providing feedback, based on context, to the module regarding errant segmentation of the components.

4. Segmentation Critic Module

As described in section 2 the bitmap image of the courtesy amount is split into probable primitives by the segmentation module. The segmentation critic module takes the list of connected components provided by the segmentation module, validates and orders the courtesy components, and transforms the list into a coarsely classified string of components which permits best recognition by the neural network based recognizer. The segmentation critic module is comprised of three parts: a classifier, a parser, and an evaluator (see Figure 4). The classifier unit matches the input component with one of seven predefined primitives. These primitives include delimiters as well as some special characters (see Figure 5). The parser unit encodes all the context knowledge associated with the seventeen styles shown in Figure 2. The eval-

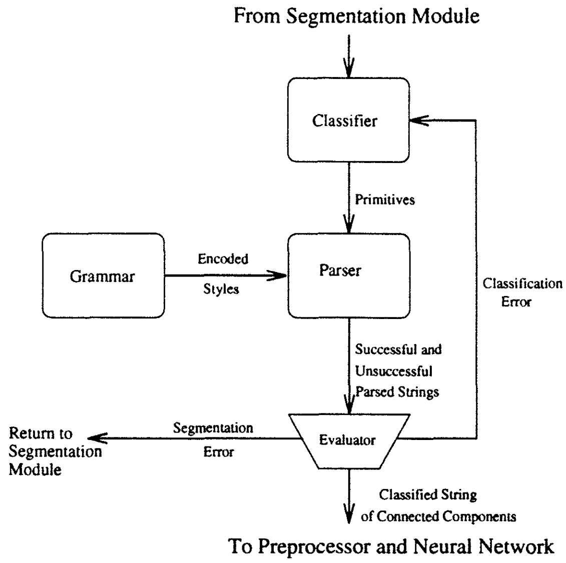


Figure 4: A Block Diagram of the Segmentation Critic Module

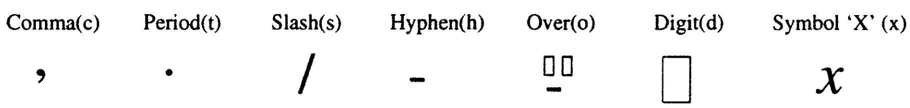


Figure 5: Permitted component primitives and their respective classes

uator unit locates the error, if any, and instructs the classifier unit or segmentation module on the appropriate action to be taken.

4.1. Classifier Unit

As described earlier in this paper, the segmentation module decomposes the input string into connected components using three algorithms geared towards different cases. Each component represents a character, presently of unknown identity, and is encoded as a minimum bounding rectangle (MBR) expressed in terms of each component's end positions in two dimensions denoted by a set of four values: x_{min}, x_{max}, y_{min} and y_{max}. The scale of the coordinate system used to determine the positional information is arbitrary and most syntactic relationships are based on the position of components relative to each other. The permitted component primitives and their respective syntactic categories are shown in Figure 5. A general classification is also given and the position of each character delineates its class.

The following heuristic features are used to assist in the preliminary classification process that is performed by the decision tree shown in Figure 6:

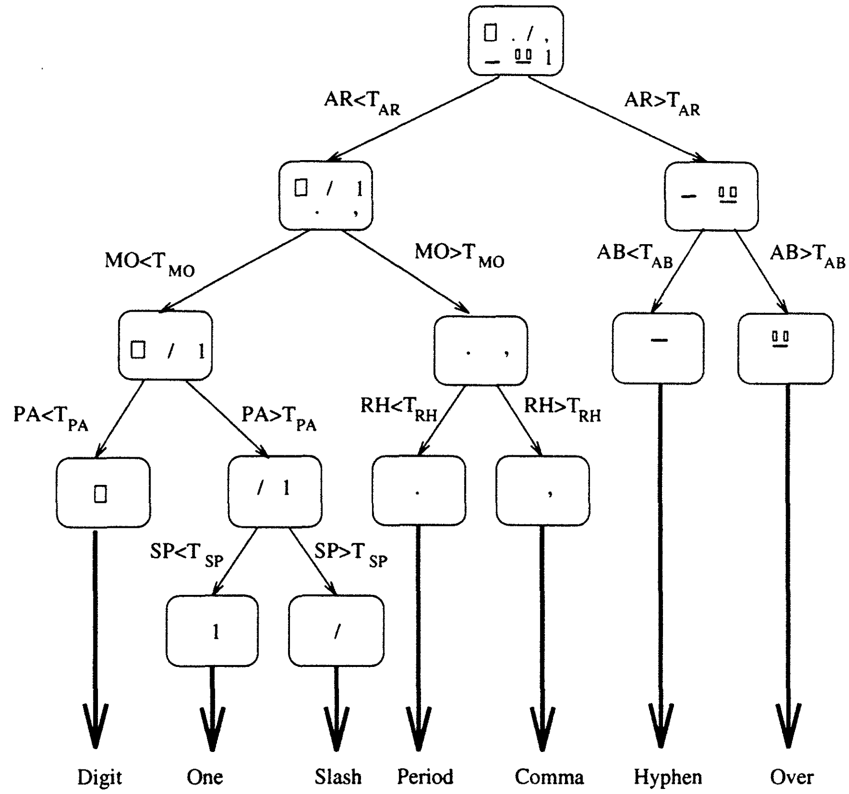


Figure 6: Primitive Classification Tree

Standard Height (*SH*)**:** The standard height is the mean average of the heights of the segmented components. $SH = \frac{1}{n}\sum_{i=1}^{n}(y_{i_{max}} - y_{i_{min}})$

Midline (*M*)**:** The one-half the standard height (*SH*) divided by two. $M = \frac{SH}{2}$

Midline Offset (MO)**:** The distance of the top left corner of the MBR of a component in relation to the midline of the string. This distance is normalized to the standard height (*SH*). $MO = \frac{y_{min} - M}{SH}$

Aspect Ratio (AR)**:** This is the ratio of length to width of the component. $AR = \frac{y_{max} - y_{min}}{x_{max} - x_{min}}$

Relative Height (*RH*)**:** This is the height of a component normalized to the standard height (*SH*). $RH = \frac{y_{max} - y_{min}}{SH}$

Principal Axis Width (*PA*)**:** The principal axis is calculated by determining the axis of least inertia as described in Horn[24]. The principal axis is an approximation of the slant of each component. The width of the component along the principal axis is then determined and is normalized to the standard height(*SH*).

Above (AB)**:** The portion of a previous component that overlaps with the current component is normalized to the standard height. $AB = \frac{x_{(i-1)_{max}} - x_{i_{min}}}{SH}$

Span (SP)**:** This is the height of the current component divided by the height of the last digit encountered. $SP = \frac{y_{i_{max}} - y_{i_{min}}}{y_{(i-1)_{max}} - y_{(i-1)_{min}}}$

The classification tree, shown in Figure 6, tests against some basic heuristics in an attempt to coarsely classify the components into the primitives shown in Figure 5.

Associated with the above list of heuristics is a constant value that indicates the decision boundary in the classification tree. These are indicated by a letter T with a subscript indicating the heuristic on which the threshold is based. The 'X' primitive is not directly determined by this simple classifier. Once a primitive is classified as a digit, the bitmap is sent to a neural network tuned to recognize the special character 'X'. The confidence value returned by the neural net must also pass a predetermined threshold T_X.

For each threshold in the classification tree there are two permissible values: a constrained value and a relaxed value. The classifier initially uses the constrained values until it is instructed otherwise by the evaluator described in Section 4.3.

4.2. Parser Unit

A survey paper[25] discusses several techniques used to store the context for a given recognition application. The efficiency in storage and the speed of retrieval and matching are essential for effective contextual checking of recognized primitives. It would be highly inefficient to store every possible primitive combination that is legal within a certain application. Therefore we have opted for a reasonably compact model for our contextual dictionary. Since our dictionary is not large, we have concentrated on a quick retrieval and matching structure. The finite state automaton described below parses the input sequence according to a grammar that contains the contextual information related to courtesy dollar amounts. This structure is very similar to the TRIE's approach suggested by Knuth[26].

The parser unit validates the connected component sequence generated by earlier segmentation stages. For the discussion of the parser, let us call the input set of labeled connected components simply a **string**. The set of primitives from which the strings are built is called the **alphabet** of the language denoted by Σ. Languages of interest are not comprised of arbitrary sets of strings but rather of strings with specified forms, defining the syntax of the language. For the courtesy dollar amount we have defined a *deterministic finite automata* (DFA), whose operation is determined by the input string as descibed below.

A DFA is a quintuple $M = (V, \Sigma, \delta, s, F)$ where V is a finite set of states, Σ (the alphabet) is a finite set of terminal symbols, $s \in V$ is the initial state, and $F \subseteq V$ is the set of final states, and δ, the transition function, is a function from $V \times \Sigma$ to V.

$$\begin{aligned} \mathbf{V} &= \{\mathbf{q}_0, \mathbf{q}_1, \mathbf{q}_2, \mathbf{q}_3, \mathbf{q}_4, \mathbf{q}_5, \mathbf{q}_6, \mathbf{q}_7, \mathbf{q}_8, \mathbf{q}_9, \mathbf{q}_{10}, \mathbf{q}_{11}, \mathbf{q}_{12}, \mathbf{q}_{13}, \mathbf{q}_{14}, \\ &\quad \mathbf{q}_{15}, \mathbf{q}_{16}, \mathbf{q}_{17}, \mathbf{q}_{18}, \mathbf{q}_{19}, \mathbf{q}_F\} \end{aligned}$$

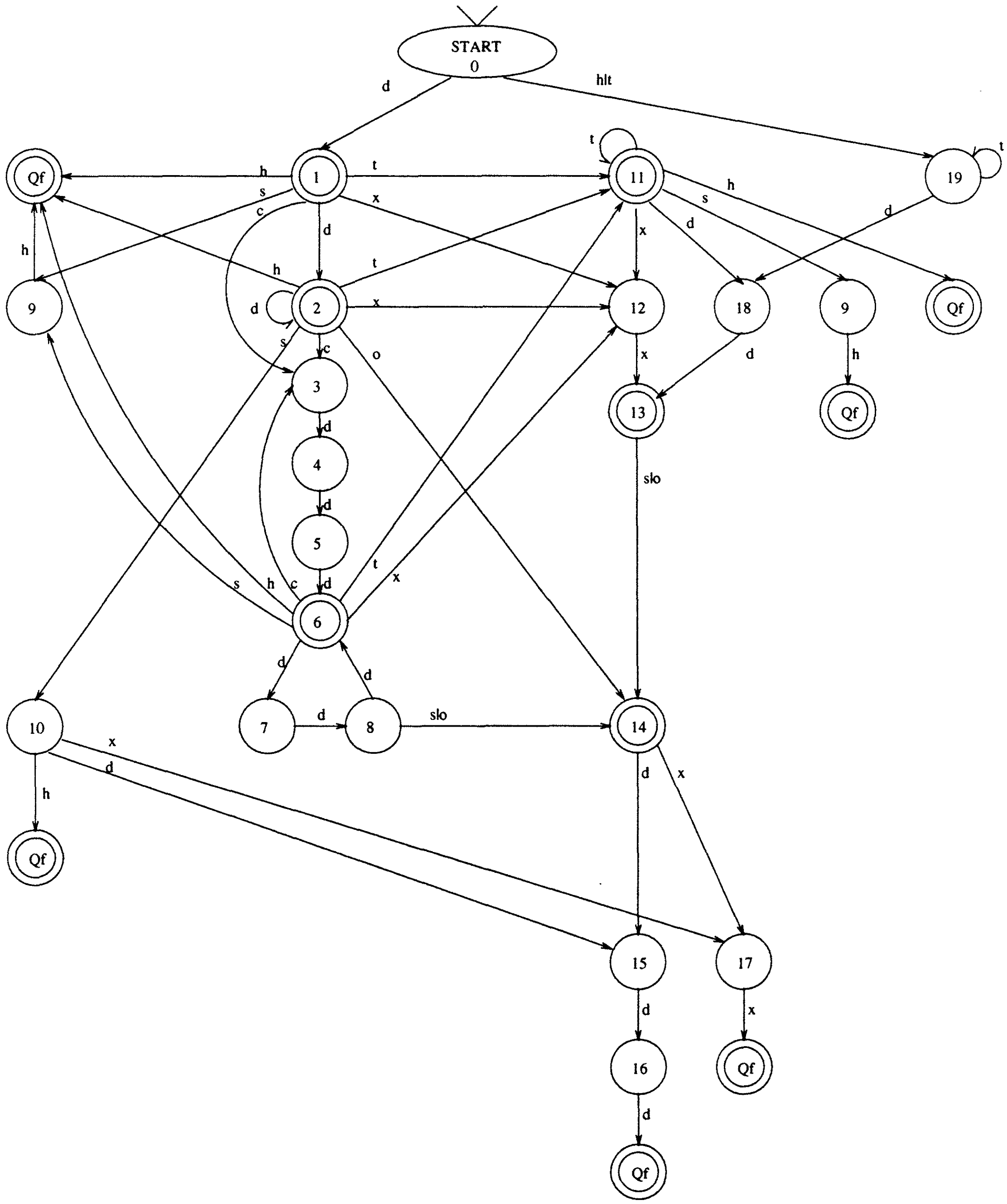


Figure 7: State Diagram of DFA

Table 1: Parser State Transition Matrix

STATE	TYPE	Input Symbol						
		d	c	t	o	x	h	s
Q0	S	Q1		Q19			Q19	
Q1	B	Q2	Q3	Q11		Q12	QF	Q9
Q2	B	Q2	Q3	Q11	Q14	Q12	QF	Q10
Q3	S	Q4						
Q4	S	Q5						
Q5	S	Q6						
Q6	B	Q7	Q3	Q11		Q12	QF	Q9
Q7	S	Q8						
Q8	S	Q6			Q14			Q14
Q9	S						QF	
Q10	S	Q15				Q17	QF	
Q11	B	Q18		Q11		Q12	QF	Q9
Q12	S					Q13		
Q13	B				Q14			Q14
Q14	B	Q15				Q17		
Q15	S	Q16						
Q16	S	QF						
Q17	S					QF		
Q18	S	Q13						
Q19	S	Q18		Q19				
QF	F							

$$\begin{aligned}\mathbf{\Sigma} &= \{d, c, t, s, o, h, x\}\\ \mathbf{s} &= \mathbf{q}_0\\ \mathbf{F} &= \{\mathbf{q}_F\}\end{aligned}$$

The rules according to which the automaton M picks its next state are encoded into the transition function. Thus if M is in state $q \in V$ and the symbol read from the input string is $\sigma \in \Sigma$, then $\delta(q, \sigma) \in V$ is the uniquely determined state to which M passes. The configuration of DFA M is therefore any element of $V \times \Sigma^*$. The transition functions are shown in tabular form in Table 1. The DFA was further optimized using the algorithm in Aho et al.[27]

Successful and failed parses are passed on to the evaluator unit. In case $\delta(q, \sigma)$ is not defined, the parser reacts with certain errors. These errors may stipulate either errors in the classification or errors due to incorrect segmentation. Upon a parsing failure, the state and position of the error are transmitted to the evaluator unit described below to determine the appropriate feedback process.

The DFA encodes two major context constraints, the length of the dollar segment and the length and style of the cents section. States **Q0-Q8** encode the constraints on the dollar section. These states enforce the fact that a comma delimeter must be followed by three digits. States **Q9-Q19** encode the constraints imposed by the 24 styles of writing the cents portion. The transitions from states **Q0-Q9** to states

Q9-Q14 occur upon the detection of a period, slash, over or hyphen delimeter (these are typical delimeters that separate the dollar portion from the cents portion). The remaining states ensure that the cents portion has the appropriate length and type for the particular style that is used. It is clear from the above description that the available context for the cents portion is significantly richer than the dollar section. Therefore errors in segmentation of the cents portion can be detected more often than errors in segmentation of the dollar section. However, this limitation of our system is only due to the lack of contextual constraints associated with the dollar section.

4.3. Evaluator Unit

The evaluator unit performs the final critique of segmentation and dispatches the errors to the appropriate stages for correction. There are essentially two feedback paths provided by the evaluator unit:

Feedback to Classifier Unit: Due to the inherent similarity between some of the primitives, namely the comma and period and the one and slash, the classification tree must be dynamically tuned. Failures in parsing that can be attributed to either of these substitution errors, prompt the evaluator to tune the appropriate thresholds in the classification process.

Feedback to Segmentation Module: The ultimate goal of the segmentation critic is to improve the segmentation process. If parsing failures are not due to the substitution errors described above, the evaluator verifies the segmentation of the connected components generated by the segmentation module and identifies the most likely candidate for further segmentation.

This unit is encoded with knowledge pertaining to the probable errors associated with failed parses. It is a rule based system that attempts several feedback paths until an appropriate parse is produced. The rules are prioritized to provide a deterministic solution to the process. However, these rules do not encompass all the errors that could have occured. After consideration of a larger set of rules, the ones that follow were chosen based on their effectiveness and relative simplicity in implementation. The basic form of all the rules is as follows:

$$
\begin{array}{rcl}
 & Given \quad Failed & Parse \;\; State \;\; \mathcal{R} | \mathcal{R} \in \mathcal{V}: \\
if \;\; < condition > & then & < action > \\
< condition > & \Rightarrow & < condition > AND < subcondition > | \\
 & & < condition > OR < subcondition > | \\
 & & < subcondition > \\
< subcondition > & \Rightarrow & String(n) = P \;\; where \;\; (0 < n < len(String) and (P \in \Sigma). \\
< action > & \Rightarrow & Threshold \;\; Adjustment | \\
 & & Concatenate \;\; last \;\; two \;\; characters | \\
 & & Adjust \;\; segmentation \;\; criteria.
\end{array}
$$

Threshold Adjustment: This indicates replacement of any of the heuristic thresholds in the classifier unit with constrained or relaxed value.

Concatenation: This process involves the 'glueing' of two primitives. It is geared to the special case when the horizontal bar above the number '5' is detached from the body of the numeral. Before concatenation, a hyphen is tested for its vertical position and proximity to the previous primitive to verify that it is part of the body of the previous primitive.

Segmentation Adjustment: There are various thresholds that direct the use of the three segmentation algorithms described earlier. These include thresholds on the aspect ratio, the primitive weight and the number of contours that are split. Relaxed values are used for these thresholds to allow for greater segmentation flexibility upon the detection of a parsing error due to segmentation by the evaluator unit.

The left column of the parser reaction table shown in Table 2 indicates the if condition while the right column indicates appropriate action to be taken. The name of the threshold indicates the module to which the feedback is directed.

5. Results

The segmentation critic system significantly enhanced our overall handwritten numeral recognition system. It successfully flagged all incorrect segmentations in the cents portion. Overall, the segmentation critic was effective at locating substitution errors (i.e., comma/period or slash/one substitution), and locating segmentation errors when there was a comma in the dollar section. The critic removed all comma/period and slash/one substitution errors.

The results are based on the modular neural network developed by our group[14]. For a training set of 5993 digits, the training phase was terminated after a mean square error of 0.25 was reached, with an accuracy of over 99.5 percent on the training set. We subsequently used a sample of 1000 checks obtained from several sponsor banks. The overall substitution error fell by 30% to 40% depending on the characteristics of the tested checks. This result shows that the parser tends to make the segmentation process more informed to get better segments for recognition. The parser algorithm has been very successful at locating the source of the segmentation error (locating 10had occurred in the dollar portion, while most segmentation errors in the cents portion where flagged.

In order to illustrate the effectiveness of the system, consider the initial scanned input image shown in Figure 1. After the courtesy amount block is detected using Agarwal et al[28], the image is dynamically thresholded to seperate the ink object points from the background. The result is shown in Figure 8. The segmentation process splits the image as shown in Figure 9. The components are then passed to the classifier unit of the segmentation critic module resulting in the string *ddscd*. Next the parser generates the following output describing the failure stage:

Parser Reaction Table		
STATE	Condition	Action
Q0	c	relax T_{RH}
	s	relax T_{SP}
	x	constrain T_X
	o	relax T_{AB}
	o AND T_{AB_R}	Concatenate
Q1	o	relax T_{AB}
	o AND T_{AB_R}	Concatenate
Q2	NULL	NULL
Q3	s	relax T_{SP}
	x	constrain T_X
	h,o	Concatenate
Q4	c,t	Check Segmentation
	End	check c at Q2 relax T_{RH}
	s	relax T_{SP}
	x	constrain T_X
	h,o	Concatenate
Q5	c,t	Check Segmentation
	End	check c at Q2 relax T_{RH}
	s	relax T_{SP}
	s AND T_{SP_R}	check c at Q2 relax T_{RH}
	x	constrain T_X
	x AND T_{X_C}	Check Segmentation
Q6	Same as Q1	Same as Q1
Q7	Same as Q4	Same as Q4
Q8	c,t	Check Segmentation
	h	relax T_{AB}
	h AND T_{AB_R}	Concatenate
	x	constrain T_X
	x AND T_{X_C}	Check Segmentation
Q9	c,t,d,x,s	Check s at String(n-1) relax T_{SP}
	o	relax T_{AB}

Table 2: Parser Reaction Table (Continued on next page)

Parser Reaction Table		
STATE	Condition	Action
Q10	c,t,s	Check s at String(n-1) relax T_{SP}
	o	relax T_{AB}
	o AND T_{AB_R}	Concatenate
Q11	c	relax T_{RH}
Q12	d	constrain T_X for string(n-1)
	c,t	Check Segmentation
	h,o	Concatenate (if error Check Segmentation)
	s	relax T_{SP}
	s AND T_{SP_R}	Check Segmentation
Q13	h	Concatenate
	c,t,x	Check Segmentation
	d AND PrevState=18	Check t at S=11 for c
Q14	h	Concatenate
Q15	c,t,h,s,o,x	Check segmentation
Q16	c,t,h,s,o,x	Check segmentation
Q17	d,c,t,h,s,o	Check segmentation
Q18	c,t	Check Segmentation
	End	Check Segmentation
	x	constrain T_X
	s	relax T_{SP}
	h,o	Concatenate (if error Check Segmentation)
Q19		

Table 3: Parser Reaction Table (Continued from previous page)

Figure 8: Thresholded Courtesy Amount

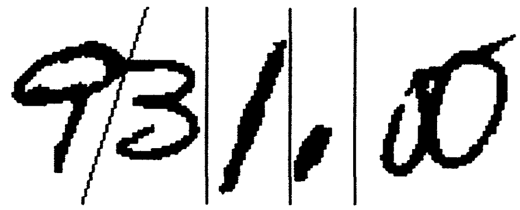


Figure 9: Initial Segmenatation

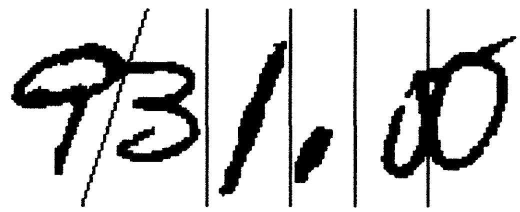


Figure 10: Final Segmenatation

```
Char => d, State => Q0
Char => d, State => Q1
Char => s, State => Q2
Char => c, State => Q10
Failed Parse at State Q10 with Input 'c'
```

As described in Table 2 the evaluator relaxes the spanning threshold T_{SP} resulting in the string *dddcd*. This again fails the parser resulting in a failure at state **Q4**. The T_{RH} threshold is relaxed and the comma becomes a period. The string *dddtd* again fails the parsing process at state **Q18**. The encoded reaction table then prompts the segmentor to relax its aspect ratio threshold resulting in the segmentation shown in Figure 10. Finally, the correct string of components is generated and passed to the later stages of recognition, yielding the required answer of *931.00*.

6. Concluding Remarks

One of the most challenging tasks in the creation of a pattern recognition system for unconstrained handwritten material is the segmentation of the input image into meaningful components. The segmentation critic suggested in this paper aids the segmentation process significantly by encoding the context knowledge associated with handwritten courtesy dollar amounts.

The proposed segmentation critic is also very efficient and adds minimal overhead to the overall check recognition system. The results discussed in Section 5 show significant improvement in the performance of the system to warrant the limited overhead associated with the critic.

References

1. Helm, Sylvia, *Banks Check into Image Processing*, Computers in Banking, v7 p25(7) March 1990.
2. Stern, R., *From Intelligent Character Recognition to Intelligent Document Processing*, Proc. Int. Electronic Imaging Exposition and Conference, Prentice-Hall, pp 236-245, 1987.
3. Cohen, E., J. J. Hull, and S. N. Srihari, *Understanding Handwritten Text in a Structured Environment: Determining ZIP Codes from Addresses*, Character and Handwriting Recognition (eds) P. S. P. Wang, World Scientific, pp 221-264, 1991.
4. Suen, Ching Y, Christine Nadal, Raymond Legault, Tuan A. Mai and Louisa Lan *Computer Recognition of Unconstrained Handwritten Numerals*, Proceedings of the IEEE, Vol. 80, pp 1162-1180, 1992.
5. Matan, O., Henry S. Baird, Jane Bromley, Christopher J.C. Burges, John S. Denker, Lawrence D. Jackel, Yann Le Cun, Edwin P.D. Pednault, William D. Satterfield, Charles E. Stenard and Timothy J. Thompson, *Reading Handwritten Digits: A Zip Code Recognition System*, Computer, Vol. 25, pp 59-64, 1992.
6. Tappert, C. C., C. Y. Suen and T. Wakahara, *On-line Handwriting Recognition–A Survey*, Proc. Int. Conf. on Pattern Recognition, Rome, pp 1123-1132, 1988.
7. Tappert, C. C., *Speed, Accuracy, and Flexibility Trade-Offs in On-line Character Recognition*, Character and Handwriting Recognition (eds) P. S. P. Wang, World Scientific, pp 79-96, 1991.
8. Sparks, P., *A Hybrid Method for Segmenting Numeric Character*, SB Thesis, MIT, 1990.
9. Baird, H., H. Bunke, P. Wang,*Document Analysis and Recognition*, World Scientific Publishing, 1994 (to appear)
10. Lam, L., S. W. Lee and C. Y. Suen, *Thinning Methodologies - A Comprehensive Survey*, IEEE Transactions on Pattern Analysis and Machine Intelligence, Vol. 14, No. 9, pp 869-885, 1992.
11. Suen, C.Y. and P. S. P. Wang (eds), *Advances in Thinning Algorithms*, World Scientific Publishing, 1993 [to appear].
12. Nagendraprasad, M. V., P. S. P. Wang and A. Gupta, *Algorithms for Thinning and Rethickening Digital Patterns*, Journal of Digital Signal Processing, Academic Press, Vol. 3, No. 2, pp 97-102, 1993.
13. Nagendraprasad, M. V., P. S. P. Wang and A. Gupta, *An Improved Algorithm for Thinning Binary Digital Patterns*, Proc. of Eleventh ICPR, The Hague, The Netherlands, August 30-September 3, pp 286-389, 1992.
14. Lik, M., Arun Agarwal, Amar Gupta and P. S. P. Wang, *An Adaptive Modular Neural Network with Application to Unconstrained Character Recognition*, IFSRC No. 261-93, Sloan School of Management, MIT, 1993.
15. Guyon, I., P. Wang(eds), *Advances in Neural Nets and Pattern Recognition*, World Scientific Publishing, 1994 (to appear)
16. Duggan, M., *Enhancing Accuracy of Automated Numerical Recognition*, SB

in Electrical Engineering, MIT, 1992.

17. Wang, P. S. P., and A. Gupta, *An Improved Structural Approach for Automated Recognition of Handprinted Characters*, International Journal of Pattern Recognition and Artificial Intelligence, Vol 5(1 & 2), pp 97-121, 1991.
18. Fu, K. S., *Syntactic Pattern Recognition and Applications*, Prentice-Hall, 1982.
19. Pavlidis, T., *Structural Pattern Recognition*, Springer-Verlag, New York, 1980.
20. Rosenfeld, A., *Picture Languages: Formal Models for Picture Recognition*, Academic Press, New York, 1979.
21. Winston, P. H., *Artificial Intelligence*, Addison-Wesley, 1992.
22. Winston, P. H. and S. Shellard (eds), *Artificial Intelligence at MIT - Expanding Frontiers*, MIT Press, 1991.
23. Rothman, Peter, *Syntactic Pattern Recognition*, AI Expert, Vol. 7, pp 40-51, Oct 1992.
24. Horn, Berthold K. P., *Robot Vision*, MIT Press 1986.
25. Elliman, D. G. and I. T. Lancaster, *A Review of Segmentation and Contextual Analysis Techniques for Text Recognition*, Pattern Recognition, Vol. 23, No. 3/4, pp. 337-346, 1990.
26. Knuth, D. E., Digital Searching, *The Art of Computer Programming*, Vol. 3, pp. 481-499, Addison Wesley, Reading, MA (1973)
27. Aho, A., Sethi R., and J. Ullman *Compilers, principles, techniques and tools*, Addison Wesley, 1986, pp. 141-145.
28. Agarwal A., Len M. Granowetter, Amar Gupta and P. S. P. Wang, *Detection of Courtesy Amount Block on Bank Checks*, in preparation.