Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

EXTRACTION OF DATA FROM PRE-PRINTED FORMS

Suzanne Liebowitz Taylor, Richard Fritzson, and Jon A. Pastor

Paramax Systems Corporation
(A Unisys Company)
Valley Forge Labs Research and Development
70 East Swedesford Road
Paoli, Pennsylvania 19301

Abstract. The widespread use of printed forms for data acquisition makes the ability to automatically read and analyze their contents desirable. The components of a forms analysis system include conversion from paper to an image through scanning, image enhancement, document identification, data extraction, and data interpretation. This paper describes techniques for manipulating electronic images of forms in preparation for data interpretation. A combination feature extraction/model-based approach is used for forms identification, registration, and field extraction. Forms identification is implemented with a neural network. The system is demonstrated on United States Internal Revenue Service forms.

Keywords: Document processing, Neural networks, Image Registration, Region extraction, Forms processing

1 Introduction

Improvements in the availability and affordability of scanning and mass storage devices have resulted in a strong movement away from the paper storage of documents. Information originally received in paper form must be converted to electronic form. In applications where some analysis of the document is required prior to storage, adding automated evaluation of the document's content from its electronic image reduces the need for human intervention in executing simple and repetitive tasks. The challenge of automating such processes arises from the variety and complexity of input documents.

Printed forms make up a significant percentage of submitted paperwork which must be rapidly processed and stored. Although printed forms have a "simple" layout in that they

contain information that is common to every particular type of a form, processing heterogeneous batches of pre-printed forms is still a challenging process which requires identifying the particular type of form, registering the input image with the model of the form, extracting information from the form, and interpreting that information. Interpretation can be achieved by converting the appropriate portions of the image to text using printed or handwritten character recognition (Suen 1980; Nagy 1982; Mantas 1986; Davis and Lyall 1988).

This paper describes a system for manipulating pre-printed forms, specifically when a unique machine-readable identifier, such as a bar code, is not present. We specifically address the processing of pre-printed forms which use printed horizontal and vertical guide lines as field delimiters. One familiar example is the set of United States Internal Revenue Service (IRS) forms. In this case, the algorithms to analyze the documents are less complex than those for more free-form documents (Wong et. al 1982; Akiyama and Hagita 1990; Elliman and Lancaster 1990). However, the large number of different forms and variations due to the printing and scanning mechanism prevent the use of a simple matching scheme. The processes of a forms analysis system, which are illustrated in Figure 1 include:

- *Scanning*: conversion from paper to an electronic image.
- *Enhancement*: processing the image to improve subsequent operations, for example, noise removal, thresholding, skew correction.
- *Form Identification*: determining the particular form type.
- *Region Extraction*: locating and identifying important fields of information on the form.
- *Data interpretation*: converting extracted regions from the image representation to a text representation, such as ASCII (The American Standard Code for Information Interchange).

Currently, our system prepares the form for interpretation, although optical character recognition (OCR) of printed labels was employed for automatic indexing. Indexing determines a unique identifier for the document instance (for example, the IRS form belongs to *J. Taxpayer*).

Our primary test case was a set of IRS tax forms. Superficial analysis of the IRS forms recognition problem might lead to the conclusion that it is appropriately addressed by matching strategies: there is a fixed set of distinct forms that one can distinguish by comparing the input image with models of each form. Implementing such a strategy, however, presents a number of practical problems.

First, one can make few correct assumptions about the layout of any particular IRS form, since forms may be obtained from the IRS, photocopied, or printed by a tax preparation package. Even forms obtained from the IRS are not necessarily identical; there are several "official" printers used by the IRS, and each of these appears to use different artwork. Therefore, different images of the same form can vary not only in size, scale, and position on the page, but also in content and structure. Any recognition strategy must therefore be based on features of the form that are relatively insensitive to these variations.

Second, some IRS forms are similar in structure, and thus require fine comparisons, necessitating high resolution in the scanned images of the forms. High resolution gray-scale images are required by subsequent processing of the form (e.g., character extraction and recognition), once it has been identified. In order to preserve sufficient detail in handwritten characters and small distinguishing features, it is typically necessary to scan forms at 300 ppi (pixels-per-inch). However, for forms identification, we would like to eliminate elements, such as characters that would cause difficulties distinguishing between forms of the same type. For our application, we found it sufficient to use 75 ppi images for the feature extraction necessary for forms recognition and registration. Using our gray-level scanner with 64 gray levels, this results in images that are on the order of 600 by 800 (eight-bit) pixels. Since there are over 160 different IRS forms and schedules, it is clearly impractical to attempt to match on full-size images. Therefore, we developed an approach which combines the data reduction of feature extraction with the precision of model-based approaches.

Form identification is implemented with a neural net as described in Section 2.1. The keystone of forms analysis is the computation of a set of geometric features which is used for both the input representation to the neural network and for image registration with the form model. In Section 2.2 we detail the form model and the feature set. An interactive window-based program developed to assist the user in creating the forms model is described in Section 2.3. Document registration and field extraction are discussed in Section 2.4. Document indexing in the form of label extraction for IRS forms is presented in Section 2.5. Image skew correction, an important image pre-processing step, is detailed in Section 2.6. Results are presented in Section 3.

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

2 System Overview

The shaded area of Figure 2 highlights the functional modules of forms processing that we detail in the following subsections. After the document is scanned and converted to an electronic image, it is transformed to a feature vector representation based on line-crossing features. This representation is used as input to a neural network for identification as well as for image registration to its model after the specific form type is determined. Once the input is registered, the important fields are checked for information content, and if they are not empty, their contents are extracted. In the specific case of IRS forms, document indexing is implemented by locating the pre-printed label on the form, correcting the label image for skew and reading the label with optical character recognition (OCR) software. If no label is present, then the handwritten or typed information is extracted during the field location and extraction stage (Section 2.4). These regions can be used as input to printed or handwritten character recognition modules.

2.1 Form Identification

The large size of the scanned forms suggests that a recognition strategy must include some means of image size reduction by one of: semantic (extraction of a region of the image that provides adequate discrimination among forms), pixel-level (e.g., image compression), or geometric (analysis at a higher level of abstraction than raw pixels, but below the level of semantic features) methods.

As in most neural network applications, proper input representation is the key to successful implementation. We evaluated all three reduction strategies, and experimented with several variants of two of them (Pastor and Taylor 1991) before concluding that a geometric feature representation was the better approach.

Semantic techniques proved impractical as a primary technique, because there is no single distinguishing feature that is present on all IRS forms. Many, but not all, forms have identifying information in the upper-left or lower-right corner, but even among forms that do include identifying information, it is not uniform-sized text, and does not appear in the same position from form to form, thus making it difficult to locate, segment, and recognize. While local high-level semantic features are impractical as a primary means of form recognition, they can be useful given a tentative identification of the form, since features may be sought

in known locations.

Pixel-level reduction was employed by block-averaging the image. We attempted to reduce the dimension of the input and keep enough detail to discern different forms; however, we were unable to find an adequate tradeoff between a sufficient size reduction to make the network tractable and the retention of enough information to make the distinctions between forms.

We examined several types of geometric features before choosing one based on the distribution of line crossings. In a similar spirit, Casey and Ferguson (Casey and Ferguson 1990) use horizontal and vertical lines on the forms themselves for both determining the form's type and registration by matching to a set for form models. However, it was difficult to find a suitable description of line patterns for input to a neural network; we wanted to avoid extensive matching in determining the form type. The line-crossing features are also used for image registration and field description. Once the features have been identified, the location of each occurrence of each of the nine *visible* types of line intersections (Section 2.2, Figure 3) is recorded. We partition the form image into nine equal rectangular bins (three horizontal by three vertical), and count the number of occurrences of each type of *visible* line intersection in each bin, resulting in an 81-element feature vector. The choice of three vertical and three horizontal bins was based on the results of hierarchical cluster analysis on many different partitionings of the image. Partitioning the image captures the spatial distribution of the features, without placing undue emphasis on precise location, thus providing some shift-, scale-, and rotation-invariance (although skew correction of the image, as described in Section 2.6, is performed before calculation of the features). As an added benefit, computing this feature vector has virtually no incremental cost, since the features that it uses are derived via simple arithmetic operations from features that are already being computed for use by another component of our system (Section 2.3).

We employed a simple feed-forward network trained using back-propagation (Rumelhart et. al. 1986). After experimenting with one and two hidden-layer models, we found that the two-hidden layer models produced better results. Floating-point state values were used and the output coding was "one-of-N". This decision scheme chooses the class which corresponds to the highest value of the N output nodes. The values of the output nodes are retained to permit checking of the second- and lower-ranked form types if form registrations fails on the first type chosen.

2.2 Form Model

We developed a robust form model which captures enough information for input image registration and region extraction without excess clutter. The model has two components: a list of corner-point features which are used for data registration and a list of rectangular data fields used for data extraction. The list of corner points provides markers for the input image registration. Each corner feature is formed in the image by the intersection of two straight line segments which define the rectangular fields on the form.

Each rectangular data field is described by its location and type of contents. The location is specified by the coordinates of the upper-left and lower-right corner points relative to the origin at the upper-left-hand corner of the image. The contents are categorized as either a simple binary field (i.e. a "check if yes" field), or a field containing numeric, alphabetic-only, or mixed alphanumeric data.

The particular corner features used for document registration describe the line-crossing patterns of the form. Each possible line crossing on the form is labeled as one of the first nine types illustrated in Figure 3. The tenth "invisible" type describes imaginary corners which are needed in order to complete the description of a field, but are not visible on the form itself (e.g. point A in Figure 4). Such points are either formed from a single line on one side boundary of a field without a perpendicular intersector (as shown in point A in Figure 4) or when a field does not contain any visible line boundaries. Although the registration algorithm is based on matching corner points, it does not depend on matching *all* corner points. Once we have successfully registered the document, we extract all regions using our standard model, whether they are defined by visible or invisible corner points.

Table 3 contains an excerpt from our IRS 1040 field description file. The first field described is the IRS label field which is located at pixel $(88, 69)$ and contains both numeric and alphabetic content. The locations are relative to the upper-left corner in a 150 ppi image of the form. Lines beginning with `POINT` describe corner points in the image and include an index number for each of these points, the pixel location of the corner point, and the type of corner point (Figure 3). Many of these `POINT` entries, for example, lines one and two in Table 3, describe the upper-left and lower-right corner points of the "box" bounding fields on the form. Lines beginning with `FIELD` contain the field name; the field type, *alpha* for fields containing only letters, *numeric* for fields only containing numbers, *alphanumeric* for fields containing both letters and numbers, and *mark* for fields which permit only a check mark; and indices to the two corner points. Extra corner points which are not used for the field descriptions are

listed at the end of the file and are used for form registration.

Each visible corner point is located by correlating a 75 ppi version (reduced by a factor of two from our original scan of 150 ppi) of the form image with four 9×9 pixel templates. These four templates, shown in Figure 5, respond to the presence of the corner types: *cornerLB*, *cornerLT*, *cornerRB*, and *cornerRT*. Notice that the actual corner point is not located at the center of the template, but rather at a variable off-set from the center. For example, using the *cornerRT* template, the actual corner intersection is at a point 3 pixels below and 3 pixels to the right of the center point. The strong response is always noted at the actual corner pixel location, not at the location of the pixel corresponding to the center of the template. These 9×9 templates detect corners formed from lines of at least 7 pixels in length. Since reduction of the image to 75 ppi resulted in one pixel width lines, the templates of Figure 5 were designed to detect 1 pixel width corners. If this method is to be implemented on a different pre-printed form which uses horizontal and vertical guide lines, it may be necessary to modify the templates to search for corners with greater than one pixel thickness.

The other five *visible* types are found by combining multiple high correlation responses for a corner pixel in the image from two or more of the corner templates according to the following rules:

1. if ($cornerLB \wedge cornerRB$) then $bottomT$
2. if ($cornerLT \wedge cornerRT$) then $topT$
3. if ($cornerRB \wedge cornerRT$) then $rightT$
4. if ($cornerLB \wedge cornerLT$) then $leftT$
5. if ($cornerLB \wedge cornerRT$) $\vee$ ($cornerRB \wedge cornerLT$) then $cross$

From the correlation output peaks and the application of the above rules, a list of corner points, their pixel locations, and corner type is completed. This list of points is then used to register the document image to its model.

2.3 Form Model Interactive Program

Manual specification of all the corner types and field descriptions for the form model is a tedious process. To aid the user, we developed a window-based tool Mform written with Motif[1]

[1]trademark of the Open Software Foundation, Inc.

and the X Window System[2]. Mform has both interactive and non-interactive functions.

First, the user runs the non-interactive portion with several images of a particular type of form as input. Each of these images is correlated with the four corner templates. The coordinates of strong output peaks are marked along with the appropriate corner type. The rules of Section 2.2 are applied to corners with multiple peaks in each image; results from each of the correlation outputs from each input image are merged to form the final model with its corners marked and labeled. This final list of corner features is available as input for the interactive portion of Mform.

The results from the non-interactive portion of Mform are superimposed on a form image for the user to examine. The interactive interface is used to complete the corner description and define the form fields. On the right side of the window (Figure 6), there is a "mouse pad". When the cursor appears in this region, a cursor will also appear on the external monitor which displays the blank form image and corner markings. Mouse clicks in this region will indicate corners in the image to be modified. The specific modification is indicated by clicking on appropriate buttons to remove corner markers which were incorrectly placed, add corner markers, and change the descriptions of corner markers (as defined by the field Corner Type at the top left side of the window). The corner type is selected by a pull-down menu. The addition and removal of corners are controlled by the buttons Add Corner and Delete Corner.

In a similar fashion, the two corner points which describe a field are set by selecting the corner points with the mouse and pushing either the button Set Top Left Corner or Set Bottom Right Corner. The field label is typed in, and the field type is selected from a pull-down menu as either mark, text, numeric, or mixed, where text indicates an alphabetic field, and mixed indicates an alphanumeric field. After modification of the corner and field list are complete, the corner point list and the region specifications are saved for image registration and field extraction.

2.4 Document Registration and Field Extraction

The registration procedure is a key component of the field extraction process. It is responsible for aligning the visible points on a newly scanned form with the internal database of points which describe a particular form type.

[2]trademark of the Massachusetts Institute of Technology.

The registration problem is due to the inability to predict the precise location of the form on the scanner bed when the image is formed. Although two scanned points will always be the same distance from each other on different scans (barring any extreme scaling in the image), they may on each scan have different locations with respect to the upper-left-hand corner origin of the page. The registration procedure works with two lists of points, one which describes all the visible points on the "standard" form and one which describes the observed points on the current form instance. Each point is described by its X and Y location on the page relative to the top-left corner and its "type" (as defined in Figure 3).

The result of the registration process is a two dimensional displacement (i.e. a pair of X and Y values) which, when applied to the list of newly observed points, will produce a new list which is precisely aligned with the standard list. The basic procedure is to try all reasonable displacements of the new form and select the one which produces the best results. All "reasonable" displacements of the top form consists of the set of approximately 675 pairs of points (for an IRS form) from (origin-25, origin-25) through (origin$+25$, origin$+25$). For each of these, the program attempts to find a mapping of each of the newly observed points to a corresponding one in the standard set. A point is mapped to a corresponding one if it falls within three pixels horizontally or vertically. Point scores are summed so each successful point match raises the score for that displacement.

The program does not simply select the single displacement with the highest score. Our experience has shown that it is usually possible to select an inaccurate displacement using that technique, since forms with large sets of clearly marked fields (e.g. tables created by vertical and horizontal lines) can achieve high alignment scores if they are registered "off by one column width or row height". To counter this effect, we first find a good average direction for the displacement and then select a best displacement in that direction. Specifically, this means the program first finds the direction of movement most likely to produce a good match by selecting the quadrant with the highest cumulative score. The four quadrants consist of the four sets of displacements whose values are $(+,+)$, $(+,-)$, $(-,+)$ and $(-,-)$. After selecting a quadrant, the program selects the best alignment within that quadrant.

After the proper alignment of the image with the model is complete, the coordinates of each field of interest in the model are displaced by the appropriate amount. In this way, we are able to extract all fields whether or not their boundary points were used for registration. Once the fields have been located in the image, each is checked for contents by comparing the number of text pixels (pixels whose gray level is below a threshold) within the interior of the field region to a threshold. On a 150 ppi image, the thresholds we used were: 10 pixels

for a small region less than 150 pixels in area, 40 pixels for a small region which is less than 500 pixels but greater than 150 pixels in area, and 60 pixels for a region greater than 500 pixels in area. If the field is not empty, the boundary lines are removed and the data are extracted. We search the image space slightly above and below the field for characters which extend outside the boundaries. This commonly occurs in alphabetic text fields containing both upper- and lower-case letters.

If this technique is to be implemented with character recognition, it would be necessary to scan the images at a higher resolution than 150 ppi, shrink the images to 75 ppi for recognition and registration, and then perform region extraction on the 300 ppi version. Because 300 ppi images are cumbersome to work with, we tested the extraction technique using 150 ppi images.

2.5 IRS Label Extraction

For the special case of IRS forms, we developed a method for detecting the presence and location of the pre-printed label. The label is supposed to be placed within a specified region; however, it is not uncommon for the label to extend over the boundaries and to be affixed skewed. Therefore, in addition to locating the pre-printed label, we correct for skew before input to the OCRA (OCRA is the particular character type) optical character recognition software. The label is detected by searching, through template matching, for the asterisk pattern that is common to all pre-printed IRS labels (Figure 7). We search for single asterisks, and if enough are detected, a least-squares fit determines the connecting line. If this line is not horizontal, the label is skewed, and we rotate the region accordingly. The character recognition is implemented with Unisys proprietary OCRA character recognition software. If not enough asterisks are detected, or if the ones detected do not form a straight line, then we know that no pre-printed label exists. In this case, contents of handwritten label information can be extracted during the region extraction using the technique presented in Section 2.4. Each line of the label can be described as a separate field.

2.6 Paper Skew Correction

The aforementioned techniques in Section 2.1– 2.4 for form recognition and form registration are sensitive to document skew. Before any corner features are calculated, we detect any

rotational deviation of the form image and, if necessary, rotate the image before subsequent processing. First, the image is thresholded so that the paper is "light" and the scanner background and ink are "dark".

A bounding box is found for the image (see Figure 8). An edge finding routine selects points along the edge of the bounding box and searches inward until encountering the "lighter" paper background region. This generates a set of points along the edge of the page which are fit to a line (using a least squares fit) and describe the edge of the paper.

The first edge to be detected is the top edge. It is found twice. Once, approximately, using a set of points clustered around the middle of the top side of the bounding box. Then a better sample set, one that is more "spread out" along the line, is generated and the procedure is repeated. The bottom and two side edges are found in a similar fashion. The angle of skew is easily calculated from the angle the "bottom" boundary of the page makes with a horizontal line; the image is rotated counterclockwise by this amount.

Using the detected page boundaries, we can approximate any scaling that has occurred and correct by expanding or reducing the image to fit into the expected paper size.

3 Results

Since there is no public database of filled-in IRS forms, we generated our own training and test data. Each scanned image occupies an average of about 1.4 megabytes of storage; therefore, generation and maintenance of a database have presented a significant challenge. The training and test data images were scanned from a combination of sources, including both original IRS forms and photocopies, and both blank and hand-filled-in forms. Scanning was deliberately haphazard, with shift and rotation introduced casually. Photocopies made at scales ranging from 0.9 to 1.1 were also included in our data set. The training and test sets for our experiments were mutually exclusive and selected randomly from the complete data set.

Images are captured off-line on a 64 gray-level Fujitsu M3191 Image Scanner and stored on optical disk. Image processing is performed using a set of routines written in C, some of which rely on an Androx ICS-400XM9^{TM} signal-processing board in a Sun Microsystems SPARC^{TM} workstation. The neural network is implemented with a dedicated accelerator board (the HNC AnzaPlus^{TM} DP) in a networked Sun 3/160 workstation. In production,

the scanner output would be fed directly to the preprocessor whose output would then be fed to a co-processor hosting the neural network.

3.1 Forms Identification

Our 232-image neural net database contains 11 samples of 21 different forms. Nine samples of each form (189 images) were used for training and two samples of each form were used for testing. Training on the 189-image training data base ceased when the change in the mean-square-error from one presentation of training data to the next was less than 10^{-4}. Initially, we compared networks with one and two hidden-layers, and with various numbers of nodes in each layer. We had the most success using nets with two hidden layers containing 40 nodes in each layer and thus adopted this architecture.

With this architecture, we achieved 98% (41 out of 42) accuracy on the test set. This figure was measured by using the largest response output node. Some "correct" identifications were made with the activation level of the best output node below 0.5, or with the ratio of the highest activation level to the second-highest only slightly better than 1:1. Weak identifications can be resolved in the form registration process by finding the best match to the form models in question. Saving the confidences of the classification allows us to try again with a model of the second or third choice form type if a match is not made in the initial registration stage. At any point, the system can reject the form and designate it for manual processing.

The forms that are misidentified, or identified weakly, are typically either very similar in structure to those for which they are mistaken, or have a structure that is difficult for the line-crossing algorithm to interpret. An example of the former is illustrated in Figure 9: forms 1120 and 1120-A are visually quite similar, and their line-crossing profiles will be almost identical. An example of the latter is form 1040EZ, which uses shaded boxes rather than lines to define input areas and poses a difficult challenge to the line-crossing algorithm (as well as few points for image registration).

3.2 Data Extraction

Data extraction was tested on the first and second pages of IRS form 1040, the first page of IRS form 1040A and a U.S. Department of Defense (DoD) Security Clearance form. The

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

IRS data base consisted of 100 images of 1040 forms and 35 images of 1040A forms manually completed by our colleagues. In order to test the technique on a non-IRS form, we developed a model for a DoD Security Form which was tested on four filled-in versions. All models were created with the assistance of Mform.

We defined 59 different fields for the IRS 1040A form, 69 different fields for the front page IRS 1040 form, 63 fields for the back page of the IRS 1040 form, and 136 fields for the DoD Security Clearance form. It takes approximately 4 minutes to scan and transfer the data for an 8.5×11 form at 150 ppi and six grey levels on our scanner. The total time to process the form (after form type identification) is approximately 10 CPU seconds on a SPARCstation II.

This includes 3.5 seconds to load the image and correct for any skew. The most intensive operations are the correlations to locate the corner features. If the form is an IRS form with a pre-printed label, an additional CPU second is needed to find the label and rotate it. We tested label location successfully on labels rotated up to 40°. The security form, which has more fields, takes slightly over 10 CPU seconds to process. A breakdown and summary of the average CPU times (based on eight arbitrary runs of 1040 IRS forms) to process the forms is given in Table 4. Figure 10 shows a completed IRS form with the extracted regions.

With this technique we were able to achieve document registration for every image in the data base. The total number of regions examined was on the order of 15,000 regions. To estimate error rates in region extraction, we randomly chose 23 forms which had a total of 1587 fields, 381 of which were filled-in. Of these 381 fields only 2 regions were not extracted. We also had 5 empty regions which were extracted as full. These errors occurred in regions above and below filled-in alphanumeric handwritten text fields where ascenders and descenders from the filled-in region spilled into fields above and below. These portions of the characters may be cut off during region extraction of a filled-in field or cause data to be extracted in regions which should be empty. We eliminated some of this problem by searching for characters continuing into neighboring regions above and below (as discussed in Section 2.4).

4 Conclusions

We have demonstrated a prototype system for forms identification, region extraction and registration of pre-printed forms which use a line-crossing signature to define fields. These

steps are an important requirement for the interpretation of the data in the forms through handwritten or machine-printed character recognition.

The line-crossing signature is used to give a new feature representation for both identification of the particular form type and registration of the input image once the form type is known. Identification of the IRS forms is accomplished with a neural network. The neural network is well suited for this problem, since we can use the line-crossing features as an appropriate reduced representation of the data, and we have to distinguish among many different types of forms. With the 1-of-N output coding scheme of the neural net, we may rank the identification results of the forms. If registration of the anticipated form model fails, we can retry with the next choice of forms. Success of the image registration using the line-crossing features guarantees the location of all data fields on the form image, even those which cannot be defined by *visible* markings on the page.

Acknowledgements. This research was funded by Paramax Systems Corporation Independent Research and Development. The authors would like to thank Dan Harrington of Unisys Corporation for providing the OCRA software.

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

References

Akiyama T, Hagita A (1990) Automated entry system for printed documents. Pattern Recognition 23(11):1141–1154

Casey RG, Ferguson DR (1990) Intelligent forms processing. IBM Systems Journal 29(3):435–450.

Davis RH, Lyall J (1988) Recognition of handwritten characters – a review. Image and Vision Computing 4(4):208–218

Elliman DG, Lancaster IT (1990) A review of segmentation and contextual analysis techniques for text recognition. Pattern Recognition 23(3,4):337–346

Nagy G (1982) Optical Character Recognition – Theory and Practice. In: Krishnaiah PR, Kanal LN (eds) Handbook of Statistics, Vol 2, North-Holland Publishing Company, 621–649.

Mantas J (1986) An overview of character recognition methodologies. Pattern Recognition. 19(6):425–430.

Pastor JA, Taylor SL (1991) Recognizing structure forms using neural networks. Proceedings International Joint Conference on Neural Networks.

Rumelhart DE, Hinton GE, Williams RL (1986) Learning internal representations by error propagation. In: Rumelhart DE, McClelland JL (ed) Parallel Distributed Processing: Explorations in the Microstructure of Cognition. MIT Press, Cambridge, Mass.

Suen CY (1980) Automatic recognition of handprinted characters – the state of the art. Proceedings of IEEE 68(4):469–487.

Wong KY, Casey RG, Wahl FM (1982) Document analysis system. IBM Journal Research and Development 26(6):647–656.

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

List of Figures

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

List of Tables

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

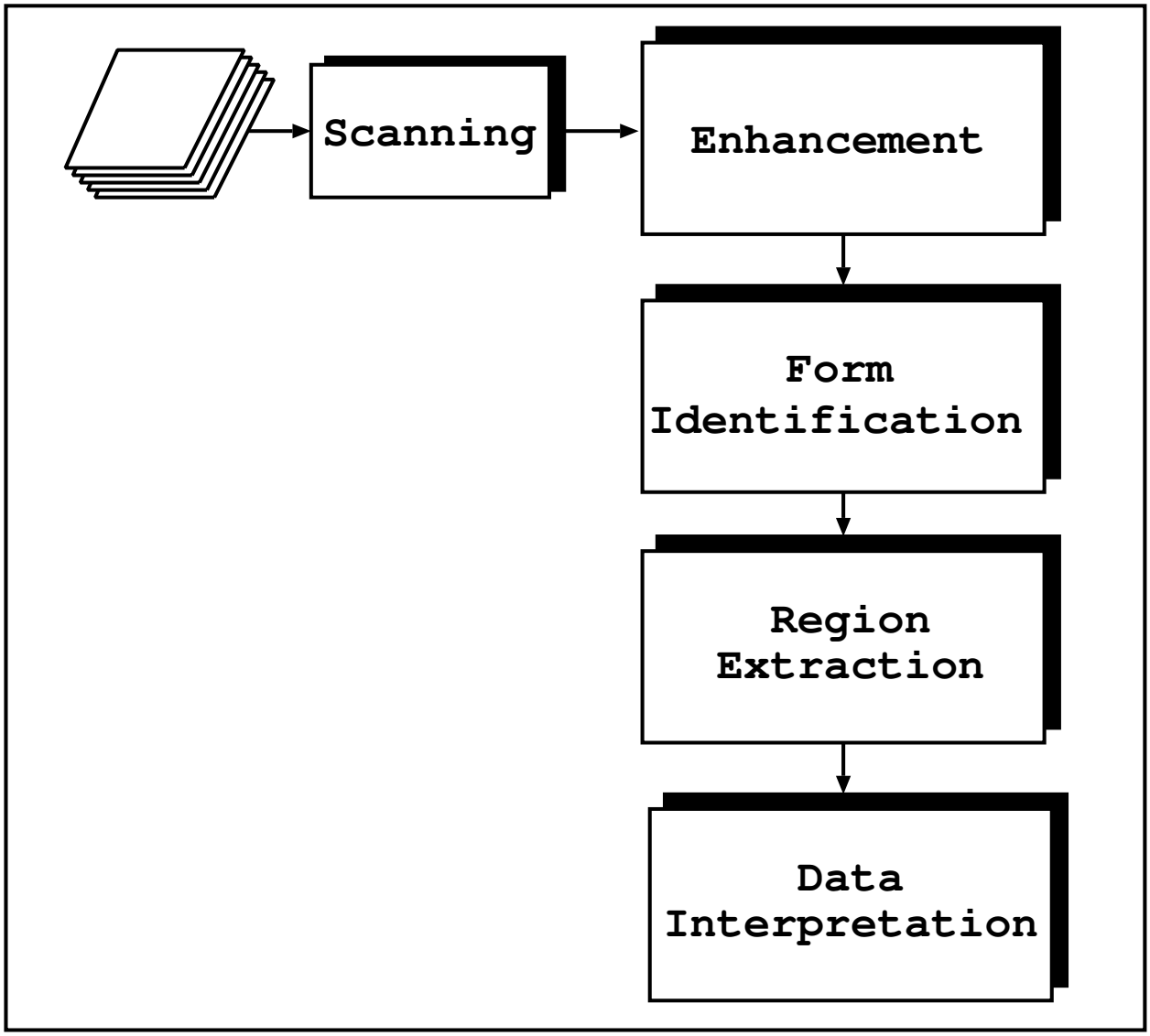


Figure 1: Image processing for forms analysis.

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

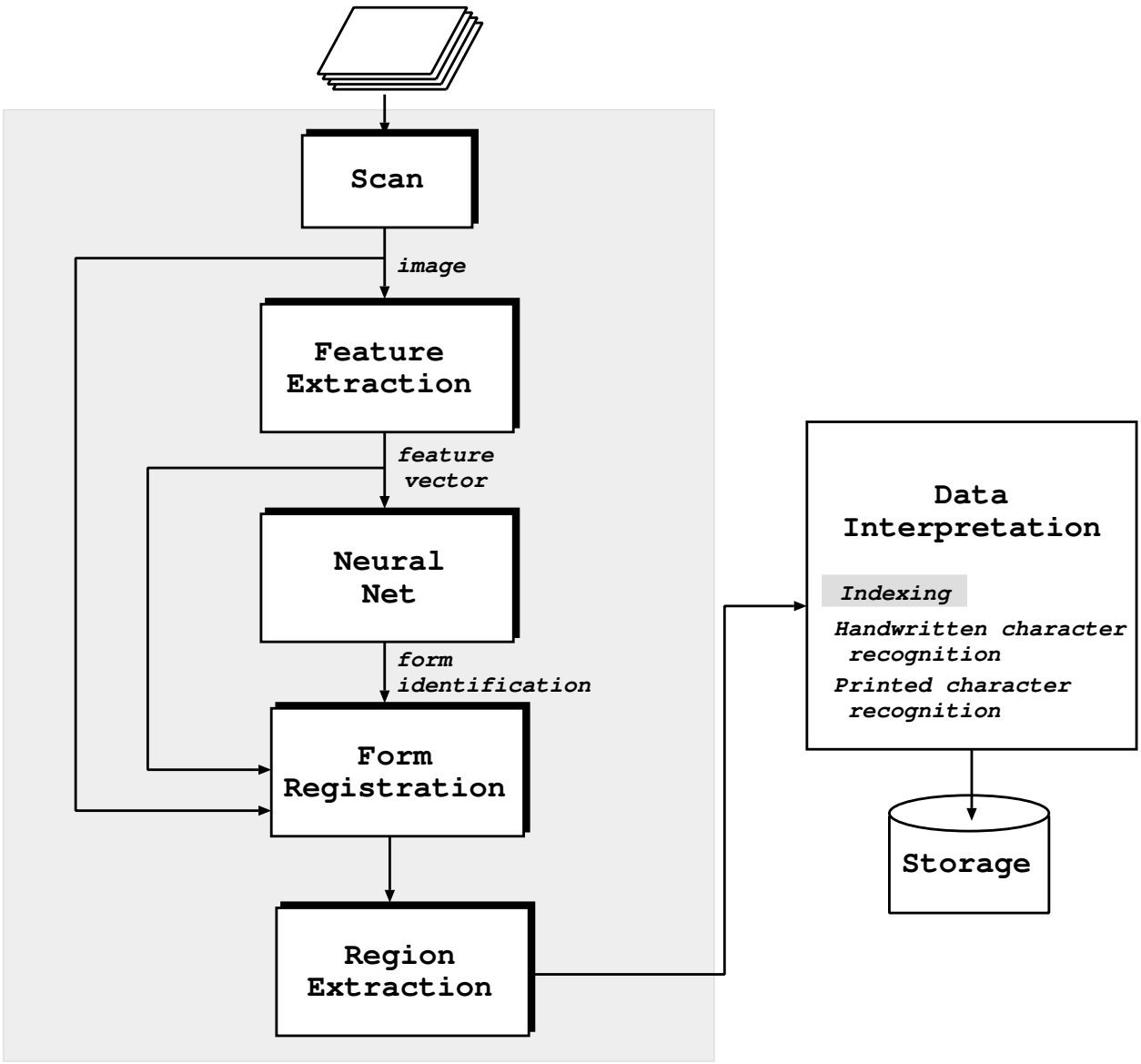
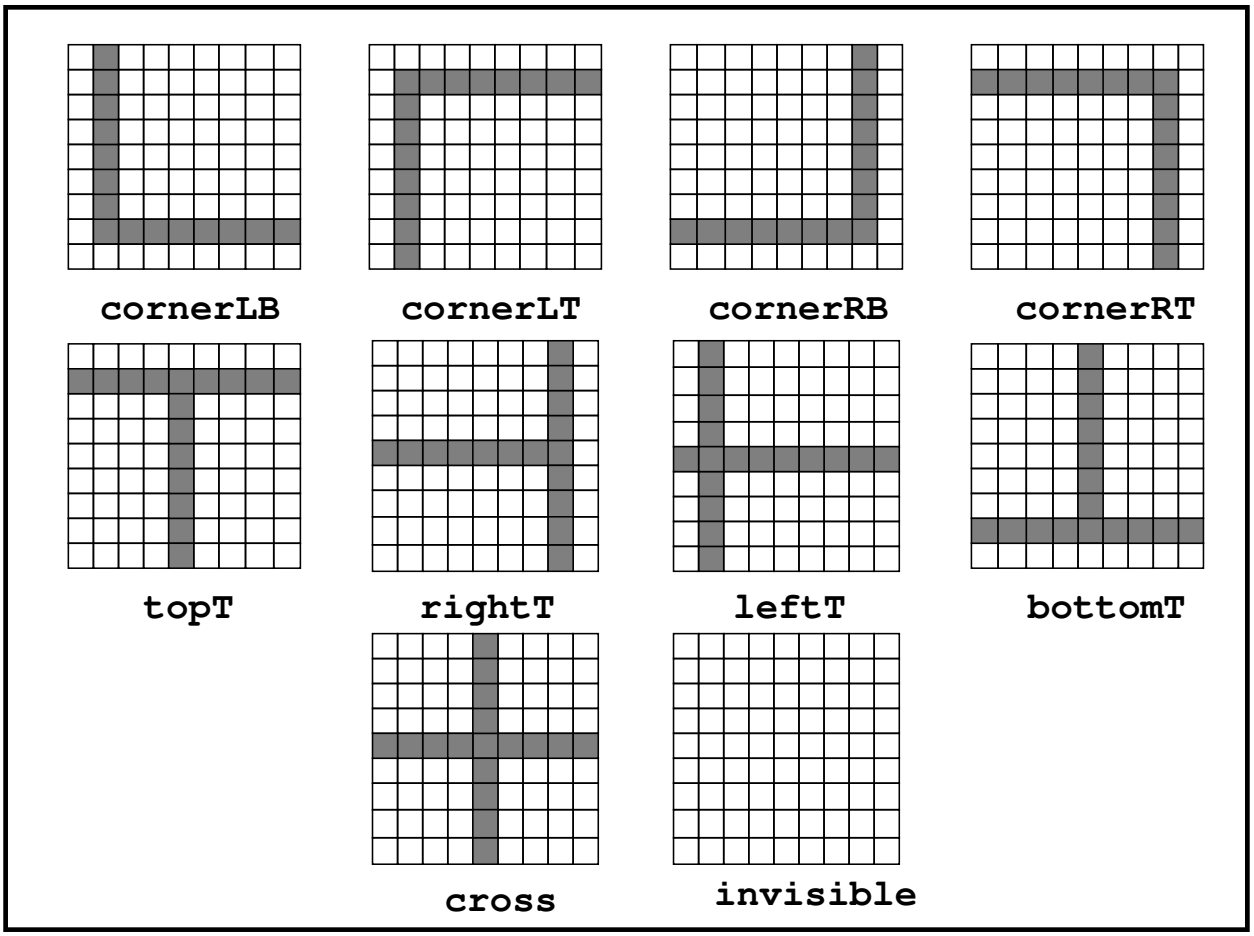
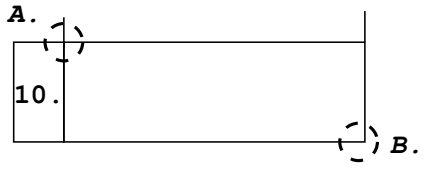


Figure 2: Feature extraction/model-based approach to forms analysis. Shaded areas are detailed in this paper.



Field defined by two corner points.



A.: Cross-junction
B.: Corner-junction
Label: Q10
Data: alpha

Figure 3: Ten line corner junctions used as features for document identification and registration. Example of a field definition using these corner points.

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

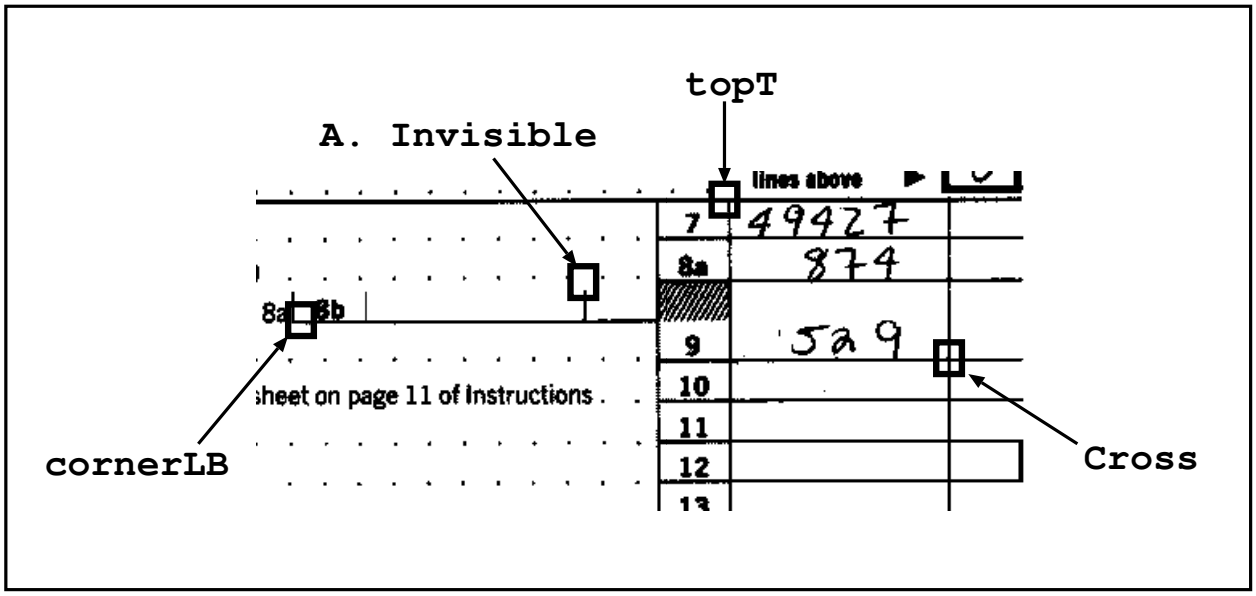


Figure 4: Segment of an IRS 1040 form with sample labeled corner types.

$$\begin{bmatrix} 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 4 & 0 & -2 & -2 & -2 & -2 & -2 & -2 \\ 0 & 2 & 4 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$cornerRT$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ -2 & -2 & -2 & -2 & -2 & -2 & 0 & 4 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 4 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$cornerRB$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & 4 & 2 & 2 & 2 & 2 & 2 & 2 \\ 0 & 4 & 0 & -2 & -2 & -2 & -2 & -2 & -2 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 2 & -2 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$cornerLT$

$$\begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 2 & 2 & 2 & 2 & 2 & 2 & 4 & 2 & 0 \\ -2 & -2 & -2 & -2 & -2 & -2 & 0 & 4 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -2 & 2 & 0 \end{bmatrix}$$

$cornerLB$

Figure 5: Four corner templates used for detecting corner types: cornerRT, cornerRB, cornerLT, cornerLB

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

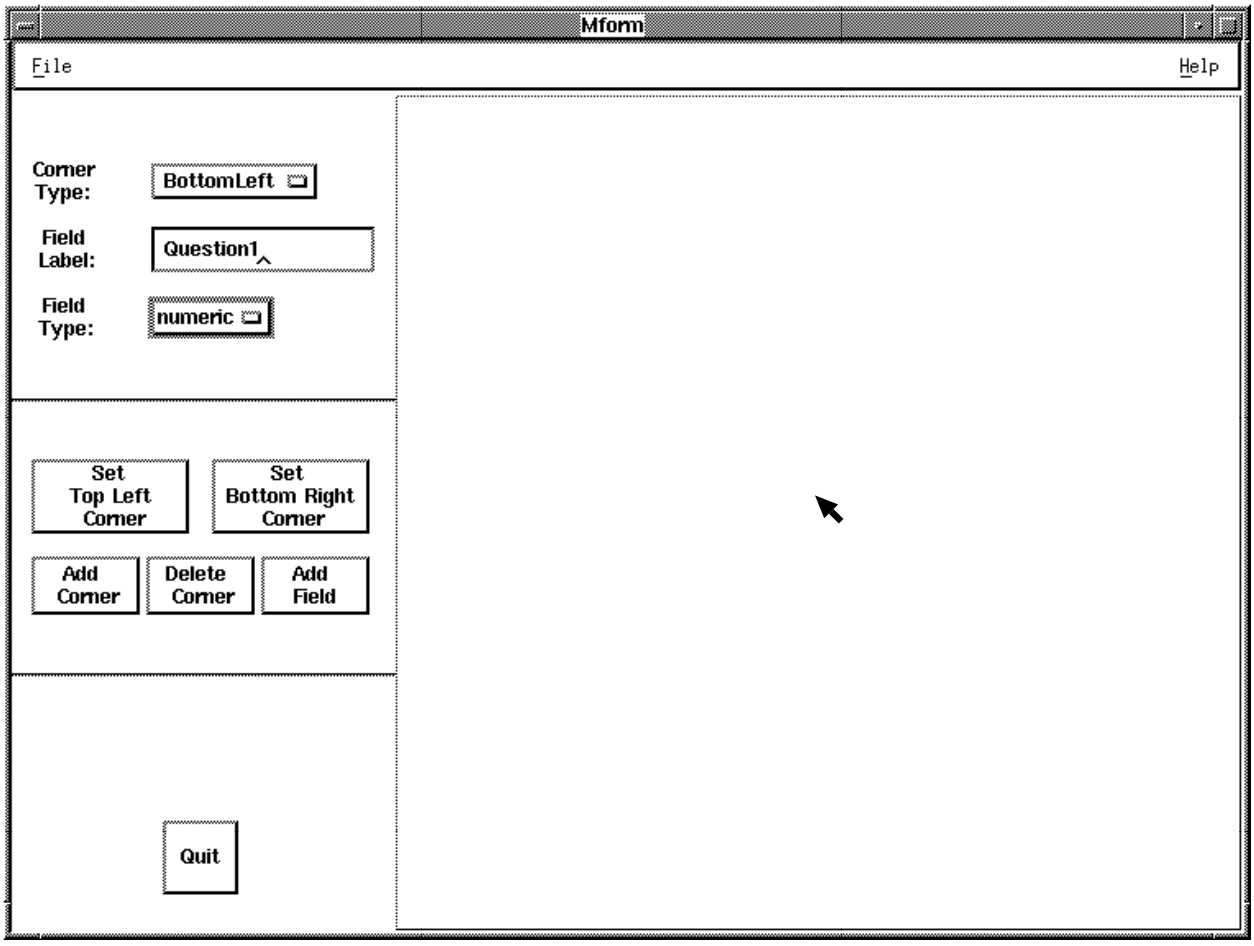
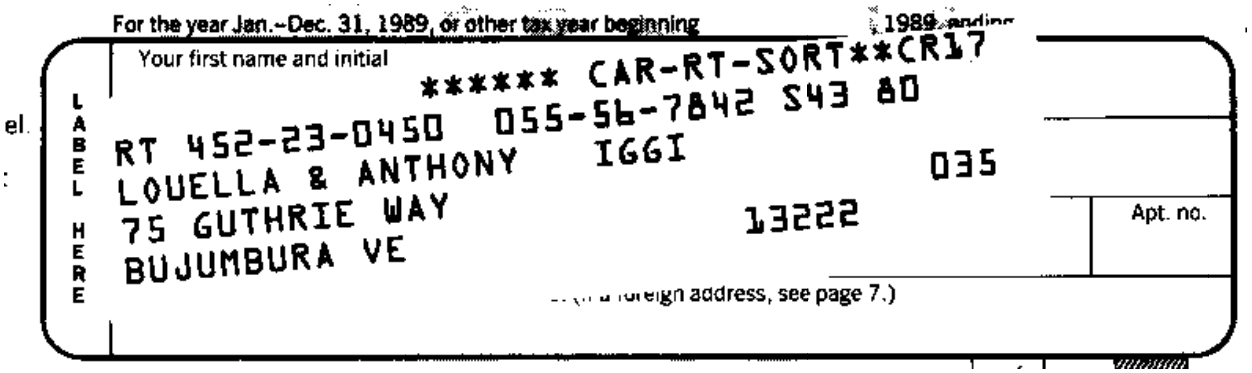
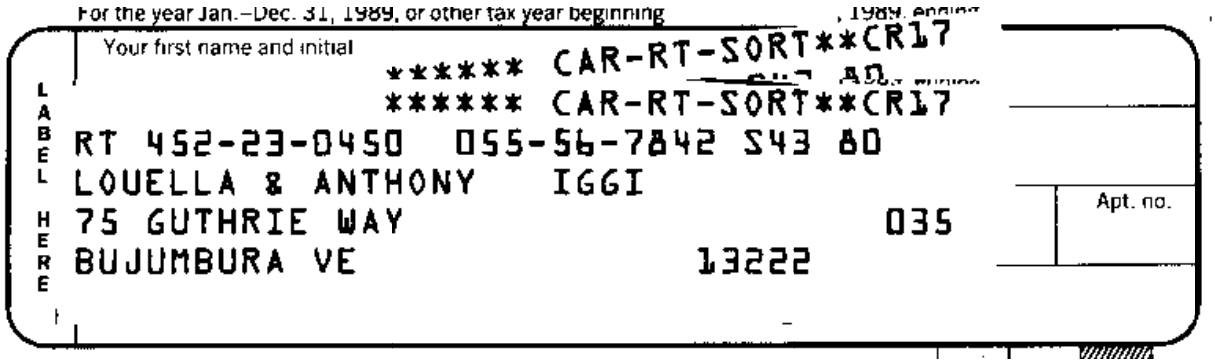


Figure 6: User interface for form model creation.



a.



b.

Figure 7: IRS pre-printed label before skew correction (a) and after skew correction (b). (Any resemblance to actual persons or address is purely coincidental).

Machine Vision and Applications (1992) 5:141-142, ©Springer-Verlag New York, Inc.

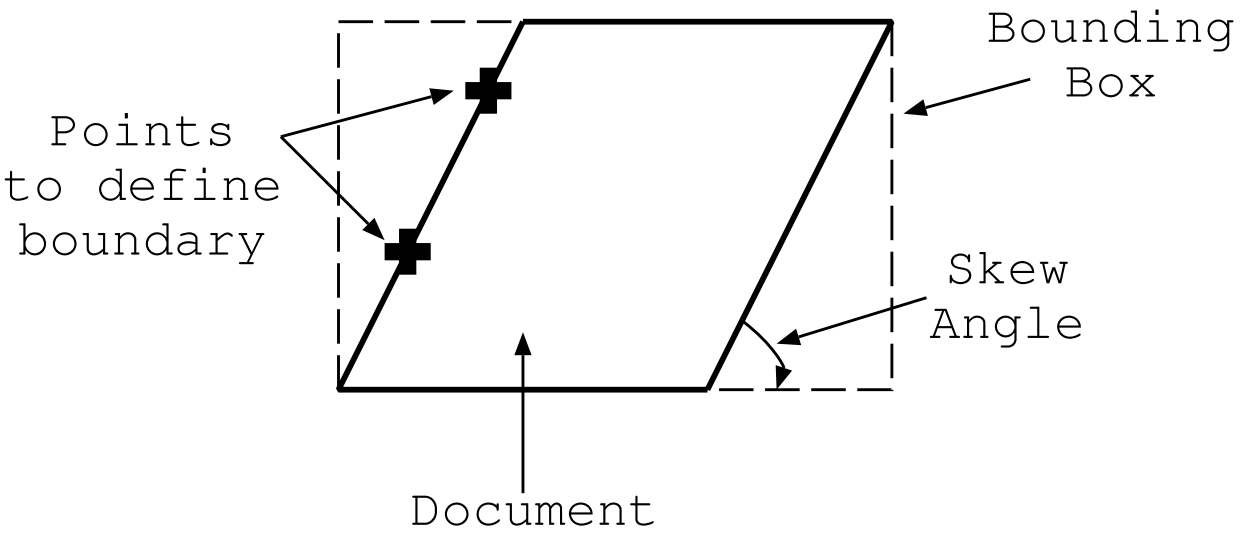


Figure 8: Detection of document image skew from enclosing rectangle.