Seventh International Conference on Document Analysis and Recognition (ICDAR), Edinburgh, Scotland, pp921-925 (2003.8)

A Prototype of an Active Form System

Taro SHIMAMURA Bilan ZHU Atsushi MASUDA Motoki ONUMA
Takeshi SAKURADA Masaki NAKAGAWA
Tokyo University of Agriculture and Technology 2-24-16 Naka-cho Koganei-shi, Tokyo, 184-8588 Japan E-mail:shimamura@hands.ei.tuat.ac.jp
Yasushi KURONUMA
Alpha Tech CO., LTD 572-7 Fujisawa Fujisawa City 251-0052, Japan

Abstract

This paper describes prototyping of a form processing system employing dot texture for printing input frames of the form. The dot texture is the texture composed of small dots. It eases the separation of handwritings from the input frames even under monochrome printing/reading environments and makes the system to process the handwritings according to the information embedded in the dot texture of the frames. The embedded information in the form dictates how to process the form so that we call the form "active form" being opposite to the passive form processed by the program stored in a document reader. This method can also be used to embed other information such as attribute of handwriting and so on. This paper presents the design, prototyping and some preliminary evaluation.

1. Introduction

Nowadays many types of forms are used in business transactions. In many form processing systems, dropout color is used in the preprinted guide lines or input frames to distinguish the handwriting from them. Since this requires color printing and color reading, monochrome copiers and monochrome facsimiles cannot be used and it involves extra costs. On the contrary though monochrome forms are cheap and prevalent, they have their own limitations. It is very difficult to separate the overlapping of the frame and handwriting in monochrome environment. Many methods have been proposed to solve this problem but all of them had their own limitations. Additionally many form processing systems which are currently in use, require the form to be registered, so as to understand where the input frames are, what are written there, how to process them, and so on.

In order to solve these problems we devised a method, which uses dot texture in place of the frame. The dot texture is a pattern constructed by a number of small dots or line segments. By this method, we can easily separate the overlap of the frame and handwriting by a simple and fast image processing methods. We can also dispense with the registration of the forms to the reading machines since the attribute of handwriting and how to process it in each input frame can be embedded on the dot texture of the frame by spreading the dots unevenly. Thus, the form itself dictate how to process handwriting such as to recognize the handwriting, to store it to the database, and so on.

This system can be made to act as a reading form without accepting any command from the keyboard or mouse etc. by embedding directions or orders about how to process the form. Therefore, we call the form "active form" being opposite to the passive form processed by the program stored in a document reader.

2. Comparing with conventional method

(1) Separation of overlap

Conventional form processing system uses dropout color on the frame to separate the frame and handwriting and requires color reading/printing environment. Hence, monochrome copiers and monochrome facsimiles cannot be employed in this process. However, our method can separate overlap in monochrome environment by an easy process. We will describe it in detail in next chapter.

(2) Registration of the form

Many conventional forms processing system needs the form to be registered for correct processing of input frames on the form. In our method, however, all the information is embedded in the frames, so that there is no need to explicitly register the form.

(3) Information embedding method

A bar code is the most common and general method used for embedding information on a paper or else. However, there are some inconveniences if it is used for embedding information on the form. A bar code must be placed near a frame or must be linked to it beforehand by the form registration.

Furthermore, printing of bar codes on the form consumes extra space and also gives odd appearance. On the other hand, our method embeds the information in the frame itself, thus association of the information to the frame is trivial and no extra space is required.

Except for the bar code, research is currently going on how to embed information on paper naturally. The DataGlyphs is the method of embedding information using small “\” and “/” [2]. Kise et al. proposed another method of embedding information on the paper in the form of an image or picture in its background by the use of small dots [3].

(4) Comparing to software applications for input data

Nowadays, inputting data into a PC with the help of a mouse or a keyboard is a common practice. However, it is a difficult task for those people who do not have knowledge about the usage of PC, and it certainly needs a PC.

In contrast, as our method only uses paper form and usual pen, it doesn’t need a PC or any other electric input devices for data input. In addition, this method is able to deal important documents (such as some contract documents, lease agreements) involving signature or personal seal. Therefore, the document of our method can also act as a tangible proof and evidence when required.

3. Basic methods

Suppose one writes characters over the frame printed by dot texture as shown in Fig.1 (a). Erosion is applied to the image (Fig.1 (b)). If it is necessary, erosion is carried out several times. If the dots are of lesser thickness than the thickness of handwriting, they disappear before handwriting disappears. Then, dilation is carried out the same number of times as that of erosion. Consequently, the handwriting is restored with almost the same thickness as of before erosion (Fig.1 (c)). Different methods can be applied to separate the overlap. One way is to measure the areas of dots and handwritings by labeling and to use the disparity in those two areas to separate the overlap. By combining these methods and handwriting recognition engine, it is possible to make up a form processing system which can separate the overlap of handwriting and the frame effectively even in monochrome environment. The dot texture can embed information such as attribute of handwriting or how to process the handwriting by varying their pitches, sizes, shapes and so on.

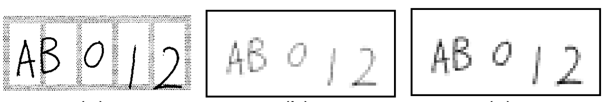


Figure 1. Separation of the handwriting and the frame.

4. System design

4.1.Structure of the system

The following figure (Fig.2) presents the structure of the system. It consists of the form preparation system that makes an active form, the form processing system, that extracts handwritings and embedded information from the active form, and the application software/hardware that acts according to the order embedded in the form.

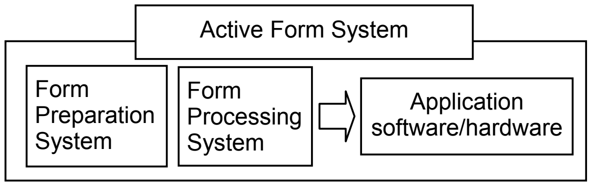


Figure 2. Active Form System.

4.2.Form preparation system

This system comprises of a PC with an active form editor (for preparing an active form) and a printer.

(1) An active form editor

An active form editor is one which can arrange input frames with their titles on a sheet and can embed information like usual form editors. The information to be embedded can be selected by means of menu.

The operating screen is as shown in Fig.3.

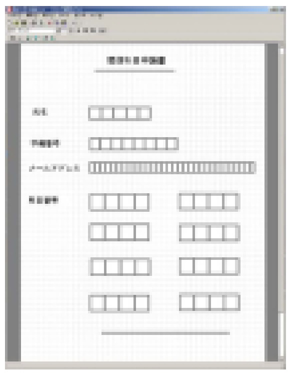


Figure 3. Active form editor.

The form prepared by the active form editor is converted to a PostScript file. We employ PostScript in this prototyping in order to control the dot shapes being independent from specific features of printers.

(2) Information embedding

For embedding information, we make use of a circular dot and a rounded rectangle (similar to a capsule) shape (Fig. 4).

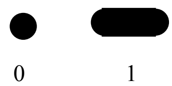


Figure 4. Dot shapes.

The diameter of the circular dot is 0.1mm and the length of the rounded rectangle is 0.25mm and they indicate a value of “0” and “1” respectively.

The information that is embedded in the frame includes

1) Attribute of the item filled in.

2) Character type that would be used in the input frame (e.g. English alphabets, numerals, Hiragana, Katakana etc.)
3) Direction.
4) Title.

The attribute is the classification of the item to be filled in the input frame that is categorized according to their meanings. Sometimes, the same contents can be called by different titles and hence we categorized the field names by their meaning.

This would not only increase the recognition rate of the handwritten characters, but also aid in the process of building a database from the extracted data. In this prototyping, we use 14 attributes. They are “Document”, “Address”, “Personal name”, “Job title”, “Amount of money”, “Number”, “Quantity”, “Date, hour and a day of the week”, ”Seal”, “Free transcript”, “Checking mark”, “Figure etc.”. We collected forms used by the administration office of our university and classified their contents into the above-mentioned attributes. We represent this data in 8 bits.

The character type is the type of characters used in the form and we used an 8 bit flag to denote each type of characters. This also raises the recognition rate. We categorize the characters as shown in Table 1 and allot a unique bit for each of the character type.

Table 1. Character type.

Bit No.	Character type
1	Alphabet
2	Hiragana
3	Katakana
4	JIS level-1 Kanji set
5	JIS level-2 Kanji set
6	Digit
7	Other marks
8	Punctuation

Table 2. Direction.

Bit No.	Direction
1	Recognize
2	Store to database
3	Send by mail
4	Store the image
5	Compare to the DB
6	Unused
7	Unused
8	Unused

The direction is the order or command to process the handwriting and the form itself. We represent it using an 8 bit flag and allocate a unique bit for each direction as shown in Table 2.

The title is the label used in the form describing what should be written in the input frame. The full title might be too long to embed, so that we take the first four characters and represent them in the shift JIS code. The title is followed by the terminal and represented by 8 bits NULL.

For the entire form, we use a horizontal bar printed in dot texture anywhere but possibly under the title of the form to represent the form’s attribute, title and so on.

We make the data set composed of the above-mentioned items as shown in Fig.5 and represent it as a horizontal sequence of dots by the embedding method described in 4.2. We repeat the sequence as far as the input frame allows and also in succeeding lines of dots for a minimum of 3 lines, thus, making the dot texture for the frame.

When repeating the data in multiple lines, we shift the data set 2 dots rightward so that overlapping handwriting does not damage the same information to be extracted (Fig. 6). We describe this in detail in next section. Data embedding is done only in the upper part of the frame (Fig. 7).

Attribute	Character type	Direction	Title	Terminal

Figure 5. Data set.

Figure 6. Arrangement of data set.

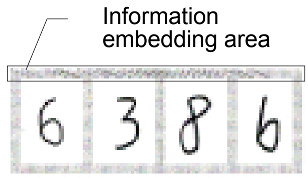


Figure 7. Information embedding area.

4.3. Form processing system

This system consists of a scanner and a PC with the processing software. The screen of the application and scanned image of a form are as shown in Fig. 8 and the processing flow is as shown in Fig. 9. The numbered components are described briefly as follows:

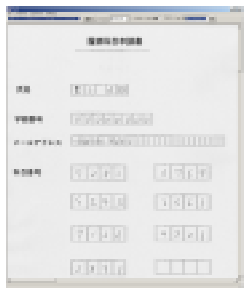


Figure 8. Scanned image of a form.

(1) Form reading

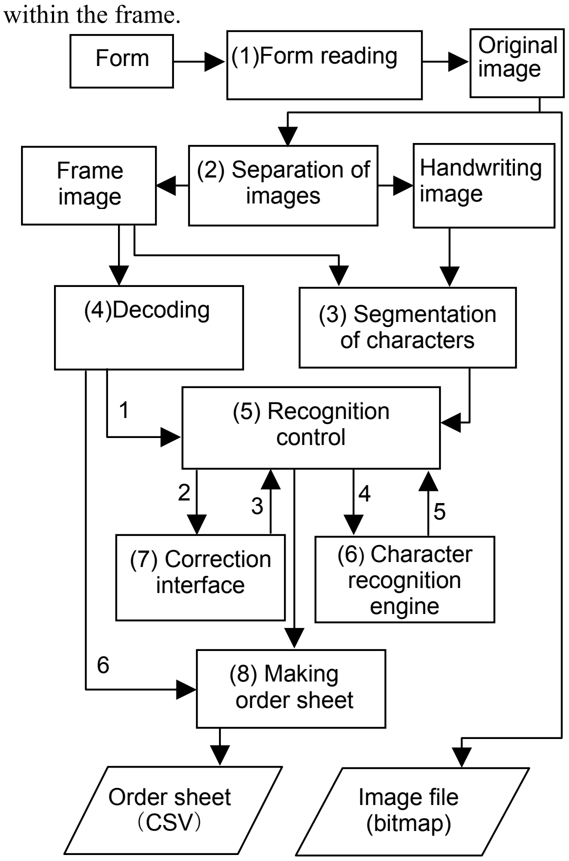
A form is read by an image scanner and a bitmap of the form image is made.

(2) Separation of images

Handwriting image is obtained by the elimination of small dots using labeling method. Though there are various methods for the elimination of dots, taking into consideration the processing speed and effectiveness, we have selected the labeling method for this prototyping. By subtracting it from the original form image, the frame image is obtained.

(3) Segmentation of characters

The handwriting image is segmented according to the locations of the frame and those of character boundaries within the frame.



1: Attribute, Character type.2:Recognition results, Candidates, Location of Handwriting on the form. 3:Corrected handwriting. 4:Handwriting image, Attribute, Character type. 5: Recognition results, Candidates. 6: Title, Directions.

Figure 9. Processing flow.

(4) Decoding

Decoding is applied to the frame image. Here, we must prepare for the case that handwriting overlaps on the frame and damages some area of dot texture. Since a multiple copies of information are embedded, however, they are extracted and decoded by the method of majority vote.

In order to elucidate the process of decoding, we exemplify it with the following example (Fig. 10). Fig.10 (a) represents the overwriting of handwriting on the frames. The gray area is overwriting and damaged dots (These are eliminated in the process (2)).

The data set is shifted 2-dot size every line as described in previous section. Then, the black area and the white area are measured as (b) in Fig. 10, the data set of each line is aligned as (c). Then, the majority vote by white area and black area is done as (d). In (c), some black areas in the rectangle of second, third and fourth line from the top are disappeared by overwriting but in other rectangles they are alive. Thus, by taking the majority vote, the true data is recovered as (d).

The resultant sequence of dots is decoded to that of "0" or "1" according to the length of dots and then these are translated to the information mentioned in the previous section. The information is sent to the recognition control to be described in (5) and the making order sheet to be described in (8).

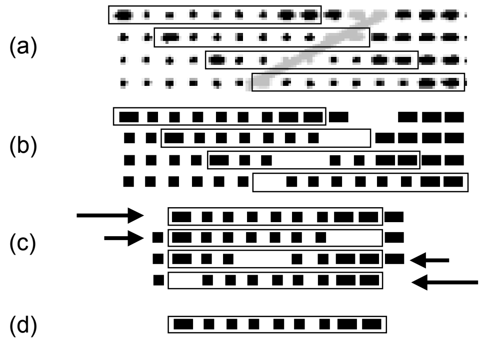


Figure 10. Decision by majority vote.

(5) Recognition control

This function sends the information from (3), (4) to the handwritten character recognition engine (6), and also sends the recognized results to the correction interface of (7).

(6) Character recognition engine

Handwritten character patterns are recognized. To improve the recognition rate, the information of character type and the attribute obtained in (4) is used.

(7) Correction interface

The recognized characters are displayed over the handwritten characters in the frame.

The correction screen is shown in Fig.11. If there is any error in recognition, it is corrected by menu selection, retyping or rewriting on the screen by pen or mouse.

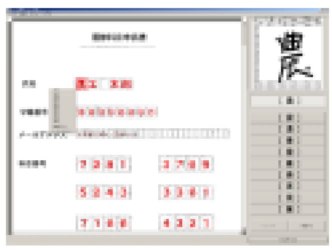


Figure 11. Correction

(8)Making order sheet

The data made by the processes (1) to (7) are bundled together into a CSV format file.

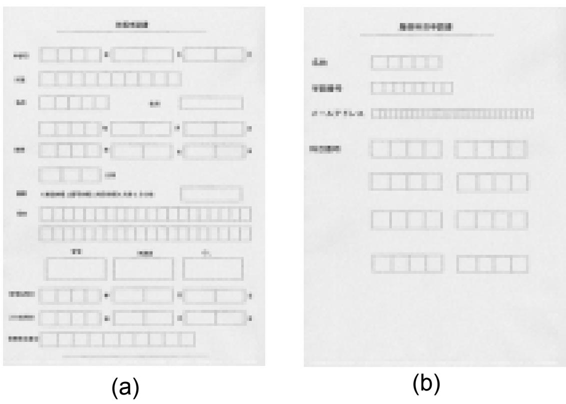
4.4 Application

An application reads the order sheets made by the form processing system and reacts according to them. As a concrete example for applications, we made a course entry system for our university. At first, when an applicant writes his/her name, e-mail address, the course numbers etc. in the form, the application scans the form, process it as described. Then, the courses taken are stored into the database and an e-mail is sent to the applicant for verification. Besides this application, anything that needs to act according to the information extracted from the form is a good candidate to apply the method.

5. Preliminary evaluation

For preliminary evaluation, we did the following experiment. First, we made two different forms: one with many input frames (Fig.12 (a)) and the other with relatively fewer frames (Fig.12 (b)) and printed out these forms by a laser beam printer at 1200dpi. Second, we asked 10 members of our laboratory to fill them up by their own handwriting. Third, we scanned them by a scanner at 600dpi. The Specification of the PC used for this experiment is Pentium(R)4 2.26GHz CPU, 1.00GB memory, and Windows XP Pro. Operating System.

We examined the success rate of data decoding, recognition rate of handwritten characters by taking a total of 20 forms and the average processing time of form (a) and form (b) by taking each 10 forms. The results of the experiment are shown in table3.



(a) (b)

Figure 12. Sample forms.

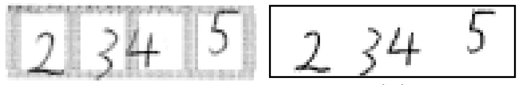
Table 3. Results of the experiment.

Success rate of decoding embedded data (%)		98.46
Recognition rates with and without using embedded data (%)	Used	93.72
	Unused	59.74
Average processing time (sec.)	Form (a)	3.8
	Form (b)	1.8

6. Consideration

As described in chapter 5, almost all the data in the frames were decoded correctly except for those frames where there were too many times of overwriting on them. The recognition rate was improved from 60% to more than 90% by employing embedding data of attribute and character type. The average processing time for form (a) and form (b), differed in the numbers of the frames is acceptable for office form processing. In almost all samples, handwritings are not overwritten on the frames, especially on the upper part of these. Japanese characters are written from left to right and from top to down, so that they seldom overlap on the left side or the topside of the frames.

Fig. 13(a) is the sample of handwriting overwritten on the frame purposefully. Fig. 13(b) is the result after the separation. The shape of every character is almost the same and all of these characters were been recognized correctly by the recognition engine.



(a) (b)

Figure 13. The sample of separation.

7.Summary

We developed, designed and prototyped the methods for embedding information in the input frames, separating handwritings from the input frames, decoding the information and process the form according to the decoded information even in monochrome environments. Elaboration of the method which is more robust against noises and which can embed more information is remaining for our future research. This research is done under the sponsorship of Ministry of Economy, Trade and Industry, Japan.

References

[1] Yoji Maeda, Masaki Nakagawa: Design of paper based user interface for editing document, Proc. SPIE, Vol.4307, pp184-192 (2001).

[2] http://www.parc.com/solutions/dataglyphs/

[3] Koichi Kise, Yasuo Miki, Keinosuke Matsumoto: Backgrounds as Information Carriers for Printed Documents, Proc.15th ICPR'2000, Vol.4, pp380-384 (2000.9).