D-Pen: A Digital Pen System for Public and Business Enterprises

Naohiro Furukawa*, Hisashi Ikeda*, Yosuke Kato+and Hiroshi Sako*
*Central Research Laboratory, Hitachi, Ltd.
1-280 Higashi-Koigakubo, Kokubunji-shi, Tokyo, 185-8601 Japan
E-mail: {n-furukw, h-ikeda, sakou}@crl.hitachi.co.jp

+Government & Public Corporation Information Systems Division, Hitachi, Ltd.
1-6-27 Shinsuna, Koto-ku, Tokyo, 136-8632 Japan
E-mail: yo-katou@itg.hitachi.co.jp

Abstract

A data entry system named "D-Pen: Digital Pen System" was developed. D-Pen requires a built-in intelligent character recognition (ICR) and can be applied to various data entry applications, such as middleware for paper-based data entry processes. Two important problems concerning the built-in ICR must therefore be solved: (1) support various character fields of business forms; (2) recognize various character-string entries in the fields. To solve these problems, a logical structural expression is defined to represent the logical relation among all the frames for characters, strings, and marks on a form. A method for layout-driven character-string recognition based on a divide-and-conquer method was also developed. Evaluation tests with volunteers showed that the recognition rate for samples of 71 Japanese address name is 87% without any errors.

1. Introduction

Paper is one of the most familiar media to people and has many advantages, such as being easy to read and write on. It is thus still widely used also in today's information society. For example, when an end-user wants to use services at government offices, bank offices, and other kinds of offices, he/she hands an application form to an employee in the appropriate section, and then that employee initiates the application processes according to the information on the forms. We define the sequences of all these processes between filling in forms and launching jobs as "data entry."

Many data entry systems using pen and paper have been applied to business tasks, such as notifications to government offices, remittances at banks, insurance applications, patient-record updates in hospitals, maintenance records for utilities services, and inventory management in warehouses. Typical examples of the processes involved in these systems are shown in Figure 1. An end-user writes entry data on a document and sends it to an operator. The operator captures the document image using an optical scanner or a tablet, for instance. The data entry system can then get the contents of the document by using optical character recognition (OCR) or conventional keypad data entry.

These conventional data entry systems generically have two drawbacks: (1) they take a lot of labor and time to transport and to scan documents into the systems; (2) they require costs for each application service as well as operation costs. Moreover, some of these systems can be used only in a limited number of locations, such as near customer service counters, on PCs, and on tablets.

With this background in mind, the authors developed D-Pen in order to solve the above-mentioned problems. This system enables user-friendly data input by digital pens and specially printed forms, and can translate data into codes by intelligent recognition of handwriting (characters and drawings) captured by the digital pens. In other words, this system can quickly and efficiently transform the handwriting contents written on a document into coded texts, with a user interface in the form of the developed pen and paper. Moreover, this system has a versatile intelligent character recognition (ICR) function that can identify a multitude of characters in a variety of formats, and also has an enterprise pattern look-up service (EPLS) function that can deliver input data to an application service automatically. Using ICR and EPLS, the D-Pen system can be applied to many kinds of data entry service at low cost. It can therefore complete a novel paper-centric process management, which highly contributes to both ease of use in data entry and to effectiveness in terms of time and cost.

2. System overview

The D-Pen system can fully automate data entry. It consists of three main elements (Figure 2): (1) digital pens and papers as input instruments; (2) a control unit for processing communication from pens to application services, digitizing handwriting information, and managing application services; (3) and application services for executing electronic data entry.

D-Pen facilitates entire data entry tasks from writing on paper to launching the designated server-side

Proceedings of the 9th Int'l Workshop on Frontiers in Handwriting Recognition (IWFHR-9 2004)
0-7695-2187-8/04 $20.00 © 2004 IEEE

application services. To automate the task completely, the control unit processes handwriting strokes as entry data in the following way.

(1) When information is written on the paper form with the digital pen, the handwritten information is sent to the control unit.
(2) The control unit identifies the form type and specifies which application is to be processed.
(3) The control unit translates the handwritten information into coded texts by ICR and structures them appropriately.
(4) According to the type of application to be processed, some elements of the structured coded texts are selected and fed into the application service in order to complete the task.

In this way, a whole task composed of paper sorting, paper scanning, paper encoding, and information dispatching to the appropriate application service can be completed by a single D-Pen system. The control unit is composed of two programs: the enterprise paper look-up service (EPLS) and the application service handler (ASH). To enable data entry, the following six steps are processed by hardware components of D-Pen (Figure 3). By virtue of the versatility of D-Pen explained below, it can be easily applied to various kinds of data entry task.

[Step 1: Pen captures pen strokes] The digital pen used in D-Pen can store pen-stroke data easily by using Anoto's functionalities [1]. Aw well as the usual ink cartridge, this pen has a camera, a force sensor, a processor, a memory unit, a battery, and a communication unit. On the paper used in D-Pen, small dots are placed in special patterns. When the user writes characters on the paper by the pen, the built-in camera captures the dotted patterns on the paper, and detects the absolute coordinates of the pen point by analyzing the uniqueness of the dot texture. The pen can also detect the pressure of the pen against the paper and the angle of the pen to the paper.

[Step 2: Pen queries EPLS] The digital pen queries the EPLS to locate the application server that is registered to handle the paper dot pattern. A database in the EPLS relates coordinates of dot patterns with URL addresses in the ASH. When the pen sends the coordinates, the EPLS locates the appropriate URL.

[Step 3: EPLS responds to the pen] The EPLS sends the pen the URL of the ASH, to which the pen then sends the pen-stroke data. The ASH is software that transfers the stroke data from the pen to the application service.

[Step 4: Pen sends collected data to ASH] The pen sends the collected data to the specified ASH over the network. The data contains the pen-stroke information that the writer entered through the paper. Since some handwriting data is extremely confidential information, such as private medical information, the D-Pen system must use encrypted stroke data for the communication from the pen to the application service. The pen sends pen-stroke data to the application service by way of the ASH because the application service cannot decrypt the data directly.

[Step 5: ASH processes the pen-stroke data] When the ASH receives the stroke data from the pen, it must perform two processes—"decryption" and "text encoding"—on the data so it can be used effectively by the application server. When receiving the stroke data, the ASH decrypts the data using its private key. Next, the ASH converts its data to text by using the built-in ICR. Most application systems need to convert handwriting data into text. The ASH is therefore equipped with the built-in ICR as the handwriting recognition.

[Step 6: Transmission terminates] The application service processes the pen-stroke data and notifies the pen when it has finished storing the data.

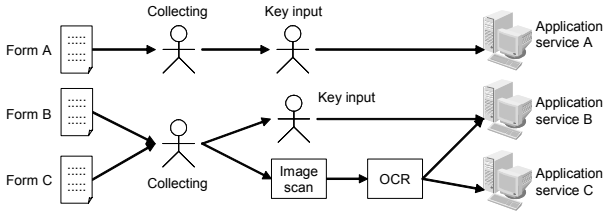


Figure 1. A usual data entry system

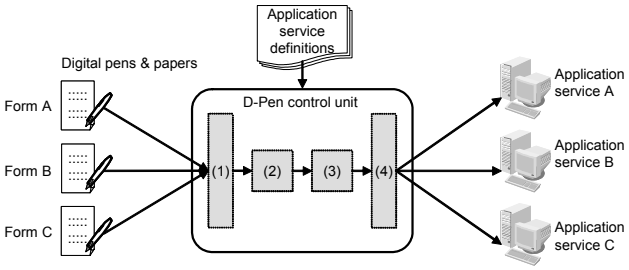


Figure 2. Overview of the D-Pen system

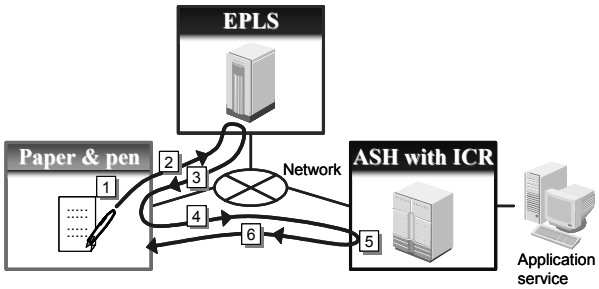
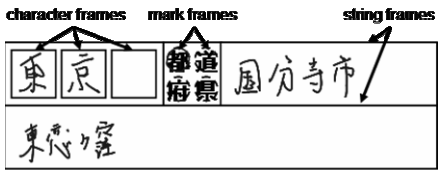


Figure 3. Process flow of the D-Pen system



→ **"東京都国分寺市東恋ヶ窪"**

Figure 4. An example of a reading field

3. Problems to be solved

The built-in ICR must satisfy two important requirements: (1) support various fields of the document form and (2) recognize various entries (see section 3.2.) To cope with these two problems, a new recognition method named "layout-driven character string recognition" was developed. Before this method is described, the reading fields are explained in the next section.

3.1. Definition of reading field

As regards the target to be recognized, many application and business forms contain frames. A character string or a mark is usually written in each of the frames. A character string is written in one frame or a sequence of frames. Such a sequence of frames is defined as a "reading field." For example, the set of frames shown in Figure 4 is a reading field. In this character-string entry, a Japanese address name is entered in the Japanese address field. Japanese address names reflect the primacy of the group in Japan. They proceed from the general to the particular, such as "東京都 (*Tokyo-to*: prefecture name)"-"国分寺市(*Kokubunji-shi*: city name)"-"東恋ヶ窪 (*Higashi-koigakubo*: area name)". The character "都(*to*)" is a suffix like that for prefecture. The mark frames are often used for specification of these suffix-like characters in Japanese forms.

3.2. Problems in recognizing reading field

Several methods for reading frame by frame in a reading field have been devised; however, there are two problems with these conventional methods. One problem is that the program and the knowledge database of the D-Pen system must be changed whenever a new business form is created or the layout is changed, because the recognition algorithm and the knowledge database depend on the form layout. It is therefore very important that the system must preserve independency from the program and the knowledge database from a viewpoint of system maintenance in the case that the system must handle reading fields in different types of forms.

The second problem is that a high recognition rate is required in the case of variation in the entry style for the reading field. Figure 5 shows this entry variation for reading fields in an example of writing a Japanese address. In a test with 132 examinees, each person was asked to write entry strings in the reading fields as shown in Figures 5 (A) and (B) without any directions regarding entry rules. The entry strings were randomly selected from Japanese prefecture and city names. In the results, the entry cases varied as shown in Figures 5 (1) to (6). The case in Figure 5 (1) is our expected entry, which is called a "regular entry". There were five kinds of entry other than the regular entry. These kinds of the entry are referred to as "irregular entries."

(2) Characters only, with no marking: All the characters of "東京都(*Tokyo-to*)" are written in the character frames.

(3) Duplication: The mark frame "都(*to*)" is also marked.

(4) No marking: The mark frame "都(*to*)" is not marked.

(5) Right-alignment: Character frames are filled in with right-alignment.

(6) Correction, etc: Mainly, in the case of irregular entry (3), the last character frame or mark frame is crossed out by a double line.

The occurrence results for each kind of the entry are listed in Table 1. The percentages of regular entries (Figure 5 (A) and (B)) are 62% and 70%, respectively. Those of irregular entries are 30% or more. For the sake of high accuracy, it is thus very important that the system can cope with irregular entries. Especially, it must read irregular entries (2) and (5) at an accuracy equivalent to reading the regular entry, because these irregular entries are not incorrect writing.

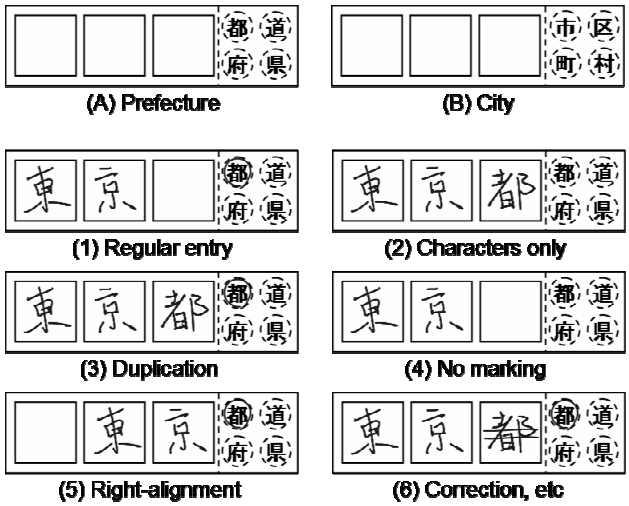


Figure 5. Entry variation

Table 1. Entry variation

		(A) prefecture		(B) city	
(1) regular entry		82	62%	93	70%
irregular entries	(2) characters only	3	3%	18	14%
	(3) duplication	23	23%	8	6%
	(4) no marking	0	0%	3	2%
	(5) right-alignment	7	7%	5	4%
	(6) correction, etc.	17	17%	5	4%
Total		132		132	

4. Layout-driven character string recognition

There are two problems in recognizing reading fields:

(1) recognizing various reading fields without changing the program or the knowledge database;

(2) coping with irregular entries.

At first, to solve problem (1), we define a logical structural expression to logically represent the relation between all of the string frames and the mark frames. Second, to solve problem (2), we propose layout-driven character string recognition based on a divide-and-conquer method.

4.1. Logical structural expression

The minimum unit of a frame is called a "cell". In the case shown in Figure 4, the reading field consists of several parts, namely, the set of letter frames for the prefecture-name entry, the set of mark frames for a suffix-like character for prefecture, the string frames for the city name entry, and the string frames for the street name entry. In addition, the set of character frames for the prefecture-name entry consists of three character frames, cells 'A', 'B', and 'C'. The set of mark frames consists of four mark frames, cells 'D', 'E', 'F', and 'G'. The string frame for city-name entry is cell 'H', and the string frame for street-name entry is cell 'I'. Consequently, the reading field as shown in Figure 4 consists of nine cells: (a) character-frame cells 'A', 'B', and 'C'; (b) mark-frame cells 'D', 'E', 'F', and 'G'; and (c) string-frame cells 'H' and 'I'.

The contents of the whole filled-in string can be obtained by concatenating the recognition results of each cell. The following equation represents the structure of a reading field, which is called a "logical structural expression."

$$AxBxCxDxExFxGxHxI, \quad (1)$$

where the symbol 'x' is a concatenation operator.

In the case of reading a mark frame, the maximum and the minimum number of selections may be specified as conditions for acceptance. For example, if there are two or more marks on the mark frame for prefecture in Figure 4, it is not a valid Japanese address name. Moreover, since the suffix character for a prefecture may be written in the character frame before the mark frame, the minimum number of selections is 0. After all, in the case of Figure 4, it is necessary to specify that the maximum is 1 and the minimum is 0. When it is necessary to extend the logical structural expression in consideration of the case of not only concatenation but also selection, a selection operator '+' is newly defined. A set of a mark cells is put in a bracket and the minimum and the maximum number of selections are put in a square bracket at the end. As a result, equation (1) is extended to equation (2).

$$AxBxCx(D+E+F+G)[0, 1]xHxI, \quad (2)$$

To simplify the recognition procedure, the logical structural expression is beforehand transformed into a tree-structure form as shown in Figure 6. The original expression is divided into subset trees expressing the portion of string cells and the portion of mark cells. In this paper, this tree is called a "logical structural tree," and the subset of cells is called a "cell binder." For example, equation (2) is transformed into equation (3), which is then transformed into the logical structural tree as shown in Figure 6.

$$\{AxBxC\}x\{(D+E+F+G)[0, 1]\}x\{HxI\}, \quad (3)$$

Logical structural trees introduce cell binders into sets of all frame types, so they can always serve as balanced trees with three layers. This is because a balanced tree makes recognition processing simple.

Incidentally, the method of expressing the contents of a reading field by the tree structure or the layered structure has been reported [2][3][4]. Compared with these methods, our proposed method for logical structural expression can:

(1) recognize mark frames and the mixture frames of strings and marks;

(2) describe reading rules such as the order of reading frames.

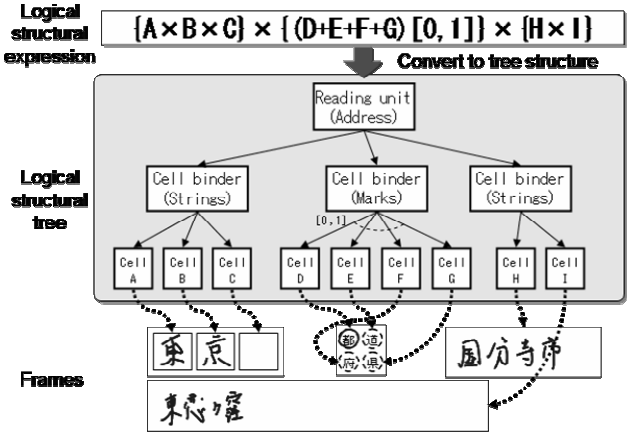


Figure 6. Logical structural expression and tree

4.2. String recognition

The main process for string recognition is based on a divide-and-conquer method. This process consists of four steps: (1) reconstruction of pen-stroke data by each cell; (2) creation of a segmentation candidate network for character segmentation in each cell; (3) unification of all segmentation candidate networks in the reading field; and (4) string matching on these networks.

[Step 1: Reconstructing pen strokes] All the pen strokes on a form are reconstructed in order of the cells within reading fields. If a stroke straddles two or more

cells, the stroke data is distributed to the cell to which the longest part of the stroke belongs.

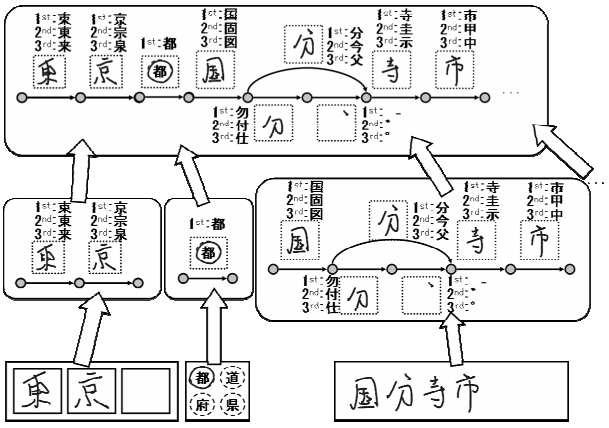


Figure 7. Create and unify segmentation candidate networks

[Step 2: Creating segmentation candidate network] This process depends on the type of cell. In the case of a string cell, at the beginning, a character pattern (a group of stroke candidates for a character) is extracted character by character from all strokes of the cell. Next, the extracted character patterns are classified and the classification result is added to each character pattern. This result includes not only the first-ranked candidate but also n-th-ranked candidates. The results of this character-pattern extraction and character classification are represented as a form of segmentation candidate network. For example, the bottom-right part of Figure 7 shows the case of "国分寺市(*Kokubunji-shi*)." As shown in the figure, some multi paths exist for expressing multi possibilities for a character pattern. If the string cell is the character frame, the segmentation candidate network consists of only one character pattern.

On the other hand, in the mark-cell case, only when it is discriminated that the mark cell is marked is the segmentation candidate network created by the pre-determined data corresponding to the character pre-printed in the mark frame on the form. The mark strokes are only used for the selection of the pre-determined data.

[Step 3: Unification of segmentation candidate networks] According to the logical structural tree, segmentation candidate networks are unified after creating a network in each cell. At first, the candidate network of each cell in a cell binder is unified. The two candidate networks are connected by joining the terminal node of the former network to the root node of the latter network. In addition, if the cell binder is a set of mark cells, the system judges whether the number of marked frames is between the minimum and the maximum numbers. It rejects the entry data in the case that the number of marked frames exceeds the range. Next, the candidate networks of the cell binder in the reading field are unified. As a result, the system obtains one candidate network of the reading field.

[Step 4: String matching] There are many methods for string matching to find predetermined strings over the segmentation candidate network. For example, the methods in [5][6][7] use dynamic programming, the high-speed matching method in [8] uses dynamic programming, and the high-speed matching method in [9] uses the TRIE as lexical knowledge.

In this paper, we use a string matching method using the Recursive Transition Network (RTN) as lexical knowledge [10], because this method can permit character-classification failure and character-segmentation failure. Using this matching, our system can recognize the text code from the segmentation candidate network.

5. Experimental results

To confirm the effectiveness of the proposed method, an evaluation experiment was performed with two sets of reading field samples (Table 2). The first set has already been explained in section 3.2. The reading field of the second set is shown in Figure 8 (A). Examinees were shown randomly selected real address names. We collected samples, without directing the examinees about entry rules. When a person fills in a prefecture name in the reading frame shown in Figure 8 (A), he/she may mark the upper-left mark frame, which indicates the prefecture name, or may fill in the upper-right string frame. Hence, this reading field has highly flexible entry.

The evaluation results are listed in Table 3. Using this result, we confirm whether the proposed method satisfies the two requirements stated in section 3.

The first requirement is to support the various fields of a document form without having to change the D-Pen program. The recognition rates of regular entries are 98.3% and 90.3%, respectively.

The second requirement is to recognize the various entries. Especially, the system should read irregular entries (2) (characters only with no marking) and (5) (right alignment) in accuracy equivalent to regular entry. As a result, the recognition rates of these irregular entries are 100.0% and 94.3%, and total recognition rates are 78.8% and 87.3% without any errors. In regards to these two points, our system satisfies the two requirements.

Examples of correct reading are shown in Figure 8 (B) and (C). Example (B) is a regular entry. Example (C) is a sample in which the examinee canceled the mark frame "府(*fu*)" by a double line after selection and re-chose the correct mark frame, "県(*ken*)." The system can recognize this irregular example correctly, because it uses mark-shape verification. The accuracy of this recognition is, however, not high, so we have to improve it.

On the other hand, an example of recognition failure is shown in Figure 8 (D). This example is rejected by string matching because two duplications exist, and it causes a rejection from the viewpoint of preventing misreading. The rejection must be a correct answer for the duplication irregular entry, since a redundant character is inserted in a duplication irregular entry. However, the percentage of duplication entries is not low, as shown in Tables 1 and 3. The D-Pen system must therefore be able to handle these samples in the future.

6. Conclusion

A new data entry system named "D-Pen" with a built-in ICR was developed and tested. The built-in ICR is able to recognize various reading fields and entries using our proposed method for logical structural expression. By means of a user interface in the form of a specially developed pen and paper, D-Pen can thus quickly and efficiently transform the handwriting contents written on a document into coded texts. Moreover, it can be applied to many kinds of data entry service at low cost.

References

[1] http://www.anoto.com

[2] G. Nagy, and S. Seth, "Hierarchical Representation of Optical Scanned Documents," *Proc. of 7th ICPR*, pp. 347-349, 1984.

[3] H. Naruse, T. Watanabe, Q. Luo, N. Sugie, "A Structure Recognition Method of Table-Form Documents on the Basis of the Information of Line Segments," *IEICE (D-II)*, Vol. J75-D-II, No. 8, pp. 1372-1385, 1992.

[4] K. Kise, K. Momota, M. Yamaoka, J. Sugiyama, N. Babaguchi, and Y. Tezuka, "Model based Understanding of Document Images," *Proc. of MVA'90*, pp. 471-474, 1990.

[5] H. Bunke, "A Fast Algorithm for Finding the Nearest Neighbor of a Word in a Dictionary," *Report of Institut fur Informatik und Angewandte Mathematik*, Universitat Bern, 1993.

[6] F. Kimura, M. Shridhar, and Z. Chen, "Improvements of a Lexicon Directed Algorithm for Recognition of Unconstrained Handwritten Words," *Proc. of 2nd ICDAR*, pp. 18-22, 1993.

[7] N. Furukawa, A. Imaizumi, M. Fujio, H. Sako, "Document Form Identification Using Constellation Matching," *IEICE-PRMU 2001-125*, Vol. 101, No. 421, pp. 85-92, 2001.

[8] K. Marukawa, M. Koga, Y. Shima, H. Fujisawa, "An Error Correction Algorithm for Handwritten Kanji Address Recognition," *Journal of IPSJ*, Vol. 35, No. 6, 1994.

[9] M. Koga, R. Mine, H. Sako, and H. Fujisawa, "Lexical Search Approach for Character-String Recognition," *Proc. of DAS'98*, Nov. 19, pp. 237-251, 1998.

[10] H. Ikeda, N. Furukawa, M. Koga, H. Sako, and H. Fujisawa, "Context-Free Grammar-Based Language Model for String Recognition," *International Journal of Computer Processing of Oriental Languages*, Vol. 15, No. 2, pp. 149-163, 2002.

Table 2. Evaluation sample sets

	reading field	string type	persons	samples
Set I	see Figure 5 (A)(B)	pref./city name	132	264
Set II	see Figure 8 (A)	address name	16	71

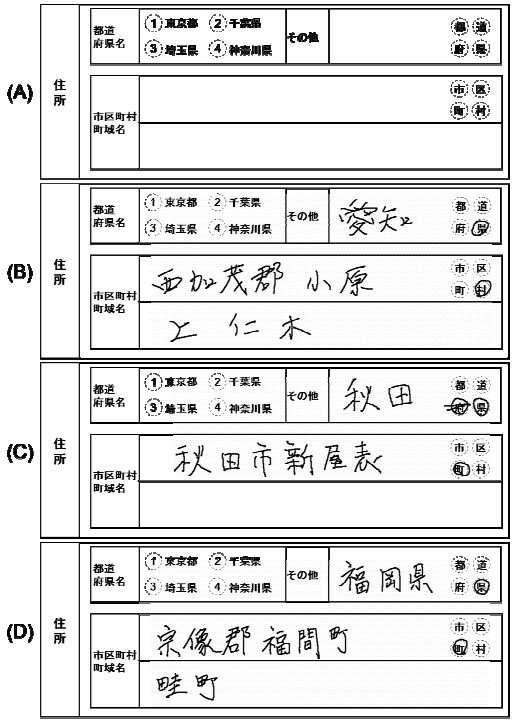


Figure 8. Examples of evaluation samples

Table 3. Evaluation result

entry type		Set I			Set II		
		correct	error	reject	correct	error	reject
(1) regular entry		172 (98.3%)	0 (0.0%)	3 (1.7%)	28 (90.3%)	0 (0.0%)	3 (9.7%)
irregular entries	(2) characters only	21 (100.0%)	0 (0.0%)	0 (0.0%)	33 (94.3%)	0 (0.0%)	2 (5.7%)
	(3) duplication	0 (0.0%)	0 (0.0%)	31 (100.0%)	0 (0.0%)	0 (0.0%)	3 (100.0%)
	(4) no marking	0 (0.0%)	0 (0.0%)	3 (100.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
	(5) right-alignment	12 (100.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)	0 (0.0%)
	(6) correction, etc.	3 (13.6%)	0 (0.0%)	19 (86.4%)	1 (50.0%)	0 (0.0%)	1 (50.0%)
Total		208 (78.8%)	0 (0.0%)	56 (21.2%)	62 (87.3%)	0 (0.0%)	9 (12.7%)