PERGAMON

Pattern Recognition 34 (2001) 187–201

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

www.elsevier.com/locate/patcog

Hierarchical random graph representation of handwritten characters and its application to Hangul recognition

Ho-Yon Kim*, Jin H. Kim

Department of Computer Science & Center for AI Research, KAIST 373-1 Kusong-dong Yusong-Gu, Taejon, 305-701 South Korea

Received 7 January 1999; received in revised form 10 June 1999; accepted 21 September 1999

Abstract

A hierarchical random graph (HRG) representation for handwritten character modeling is presented. Based on the HRG, a Hangul, Korean scripts, recognition system also has been developed. In the HRG, the bottom layer is constructed with extended random graphs to describe various strokes, while the next upper layers are constructed with random graphs (Wong and Ghahraman, IEEE Trans. Pattern Anal. Mach. Intell. 2(4) (1980) 341) to model spatial and structural relationships between strokes and between sub-characters. As the proposed HRG is a stochastic model, the recognition is formulated into the problem that chooses a model producing maximum probability given an input data. In this context, a matching score is acquired not by any heuristic similarity function, but by a probabilistic measure. The recognition process starts from converting an input character image into an attributed graph through the preprocessing and the graph representation. Matching between an attributed graph and the hierarchical graph model is performed bottom-up. Since the hierarchical structure in an attributed graph is decided after the recognition ends depending on the best interpretation of the graph matching, we can avoid incorrect sub-character segmentation. Model parameters of the hierarchical graph have been estimated automatically from the training data by EM algorithm (Dempster et al., J. Roy. Stat. Soc. 39 (1977) 1) and embedded training. The recognition experiments conducted with unconstrained handwritten Hangul characters show the usefulness and the effectiveness of the proposed HRG. © 2000 Pattern Recognition Society. Published by Elsevier Science Ltd. All rights reserved.

Keywords: Character recognition; Random graph; Pattern recognition; Hangul recognition; Grapheme-based approach; Stochastic approach; Hierarchical graph representation; Automatic estimation

1. Introduction

A handwritten character is generated by a series of trajectories of a writing instrument. This implies that the shapes of trajectories and the spatial relationship among them contain all the information needed to decode a handwritten character. The others varying the handwritten instances can be regarded as noises added on them during the information transmission process, caused by writers, writing devices, input devices, etc. In this sense, modeling the handwritten characters in trajectories and their spatial relationship seems to be fit for their origin, and also effective to remove unnecessary variations from width of a pen and other noises. Most structural systems for the handwritten character recognition have been developed under the assumption that all the information necessary to identify a handwritten character can be obtained from the trajectories, or strokes.

The main obstacle in the trajectory- or stroke-based methods is that it is difficult to extract strokes robustly. This is because the touching and the crossing of trajectories give rise to the distortion of stroke shapes and produce an ambiguous connection of strokes. As a consequence, it often becomes unable to recover original

* Corresponding author. Tel.: +82-42-869-5557; fax: +82-42-869-3510.

E-mail address: hykim@ai.kaist.ac.kr (H.-Y. Kim).

PII: S0031-3203(99)00222-8

strokes from a character image. In fact, most structural systems have suffered from stroke extraction errors, which result from fixing the ambiguous strokes with their local shapes.

The stroke extraction errors can be reduced if strokes are extracted in a global view. Rocha and Pavlidis [1], for example, proposed a method with emphasis on identification of structural descriptions of character shapes. They postpone the decision whether a pixel is black or white until they try forming a stroke. Similarly, strokes are not interpreted as straight or curved until the final matching, and also a number of shape transformations and a gap-handling procedure are introduced so that strokes of a handwritten character can be interpreted in a global view. However, because of the restriction that an input character should be a connected component, the method cannot be applied to more complex character sets such as Korean characters and Chinese characters. In addition, it may not work for some alphabets having more than one connected component such as 'i' and 'j'. This infers that only stroke modeling is not enough to describe handwritten characters, but we also need another modeling mechanism with hierarchical representation. Even though Rocha and Pavlidis made an extension of their method to the unsegmented word recognition [2], it handles only one-dimensional relationships of character positions.

There are many studies on the hierarchical representation of handwritten characters. In the Chen and Lieh's method [3], simple descriptions of strokes and their relationships are adopted to make hierarchical graphs. They construct a 2-layer attributed graph to represent a handwritten Chinese character. One layer is for the strokes and the other is for the components of a character. By synthesizing 2-layer attributed graphs, a 2-layer random graph is constructed as a reference model. Another method [4] based on the structural representation, called the hierarchical attributed graph representation (HAGR), was introduced. In this method, the strokes are represented by attributed sets, and then grouped into branches to construct the HAGR.

Although the hierarchical models reflect hierarchical nature of handwritten characters, especially of Chinese characters, stroke descriptions in the systems are too simple to specify various strokes. Such stroke descriptions may be enough to specify Chinese characters, which are mainly composed of straight lines, but, in general, more sophisticated representation mechanism is essential to distinguish similar characters. Furthermore, as Rocha and Pavlidis [1] pointed out, no effort has been made in these studies to address group features that have been divided by spurious points. Instead, strokes are merged and grouped to make a hierarchical graph without regard to global correspondence of them, and then matched against features of reference models by one-to-one mapping. This kind of methods often generates grouping errors that cannot be recovered at matching step.

To tackle these problems, a new hierarchical random graph (HRG) representation for handwritten characters is proposed in this paper. Since there are two different information sources of pen-down movement and pen-up movement, we introduced two different modeling mechanisms: one is for trajectory modeling, and the other is for relationship modeling between trajectories. The former is focused on describing trajectories approximated with a stroke or connected strokes, while the latter is focused on representing their relationships on 2D plane. These two modeling mechanisms are incorporated in a hierarchical graph representation.

At the bottom layer of the HRG, the extended random graphs called chain graphs are introduced so that they can model strokes stochastically. An arc of a chain graph represents a chain of points, or a chain of features, and a vertex represents an ending point or a contact point of chains. In the chain graph, multiple-to-one mapping of features from input strokes to an arc of a chain graph is possible according to model parameters. The correspondence between features of input strokes and model arcs is not determined until the best interpretation of the mapping is found. Not imposing any constraints, all the points in strokes may be a segmentation point. In other words, a vertex of a chain graph can be matched with any point in strokes of an input character.

The upper layers of the HRG, representing relationships between strokes, are constructed with random graphs introduced by Wong et al. [5]. For the character sets with simple structure such as numerals and English alphabet, one layer is enough to represent the relations of strokes. But, to represent the character sets having more complex structure, composed of sub-characters such as Chinese characters and Korean characters, more than two layers are used to represent their hierarchical structure.

Based on the hierarchical representation, the handwritten Korean character (Hangul) recognition system has been developed. One of the principal advantages of the system is that it is a hybrid system combining advantages of both structural and statistical pattern systems. In general, structural pattern recognition systems are easy to represent structural information of characters and find distinctive features well, but they are sensitive to noise and hard to train. So, adjustment of the system to different writing styles is difficult. On the other hand, statistical systems are trainable and less sensitive to noise, but they incline to miss distinctive local features, which results in confusing similar categories. We have attempted to make a stochastic model that has the advantages of both systems. In the proposed system, structural aspects of strokes are represented with the structure of graphs, and their variations are modeled statistically with probability distributions of random variables in graphs. Model parameters of the hierarchical graph have been

estimated automatically from the training data by EM algorithm [6] and embedded training technique.

In the proposed system, as in many stochastic methods, the recognition is formulated into a problem to find a model that produces maximum probability given an input data. To get a matching probability, similarity measures and cost functions, which seem to be somewhat arbitrarily defined, are not used. Instead, a matching probability estimated by using probability distributions of features in a model is used. In summary, the recognition is carried out in a probabilistic framework, which is a salient characteristic of our system compared with the previous systems.

The rest of this paper is organized as follows. Section 2 presents the hierarchical graph representation. A hierarchical Hangul model, an application of the hierarchical graph, is explained in Section 3. In Section 4, the procedure to make an attributed graph from an input character image is described, including stroke extraction, feature encoding, gap filling to handle broken characters, and attributed graph generation. The Hangul recognition system and its parameter estimation method are explained in Section 5 and in Section 6, respectively. Some experimental results are given in Section 7, followed by concluding remarks in Section 8.

2. Hierarchical graph representation of handwritten characters

As mentioned in the previous section, all the information necessary to interpret a handwritten character is originated from the trajectories of a writing instrument and their relationships on 2D space. Thus, a handwritten character can be effectively represented with the hierarchical random graph (HRG) representation layered by stroke models for trajectory modeling and relation models for their relationship modeling. In the proposed HRG, stroke models are made of chain graphs, while relation models are made of random graphs [5]. In this section, the HRG will be introduced. As a first step, the definition of attributed graphs is given, followed by the definition of random graphs and chain graphs.

2.1. Attributed graph

An attributed graph [4], which is considered as an outcome graph of a hierarchical random graph, consists of a set of attributed vertices and attributed arcs. The definition of an attributed graph is following.

Definition 1. An attributed graph is a graph $G = \langle N, E \rangle$, where

$N = \{v_i \mid v_i \text{ is an attributed vertex, and } 1 \leqslant i \leqslant n\}$

$E = \{a_i(j,k) | a_i(j,k) \text{ connects vertices } v_j \text{ and } v_k \text{ with an attributed relation.}\}$

2.2. Random graph

A random graph consists of a set of random vertices and a set of random arcs, all of which are random variables. The definition of random graphs [7,3] is following.

Definition 2. A *random graph* is a pair of tuples $R = (W, B)$ such that

1. W, representing the random vertex set, is an n-tuple $(\alpha_1, \alpha_2, \ldots, \alpha_n)$ where each α_i, called a random vertex, is a random variable;
2. B, called the random arc family, is an m-tuple $(\beta_1, \beta_2, \ldots, \beta m)$ where each β_j, called a random arc, is also a random variable; and
3. associated with each possible outcome graph $G = (N, E)$ of R and a graph monomorphism (or subgraph isomorphism) $M : G \to R$, there is a probability $p(G, M) = pr(R = G, Q = M)$, Q being the family of graph morphisms, which satisfies

$$p(G, M) \geqslant 0 \quad \text{for all } G \in \Gamma,$$

$$\sum_{\Gamma} p(G, M) = 1,$$

where Γ is the range of R.

Let $G = (N, E)$ be a possible outcome graph of a random graph R with a monomorphism $M = (\mu, \delta)$. Then, the outcome probability of G from R is expressed as the product of outcome probabilities of vertices and arcs in G. That is,

$$P_R(G, M) = \prod_{\alpha_i \in W} \Pr(a = \mu'(\alpha_i)) \times \prod_{\beta_i \in B} \Pr(b = \delta'(\beta_i) | a = \mu'(\alpha_k)),$$

where, $a \in N$, $b \in E$, μ' and δ' are the inverses of μ and δ, respectively, and $M = (\mu, \delta)$ is a monomorphism. $\mu : N \to R_N$ and $\delta : E \to R_E$ are functions.

2.3. Chain graph

A chain graph, composed of a set of random vertices and a set of random arcs, has been devised to enable multiple-to-one mapping of features, which gives stroke matching more flexibility. The definition of a chain graph is the same as that of a random graph except that a random arc has a length probability distribution specifying the number of features to be matched.

The notion of the length probability distribution is imported from a variable duration hidden Markov model (HMM) [8] in order to model the length of strokes. In a variable duration HMM, duration density describes the length of a series of observations to be consumed in a state. In chain graphs, the length probability distribution specifies statistically the length of observations to be matched with an arc. Applied to handwritten character modeling, it is interpreted as how many points or how long stroke should be matched to an arc. Consequently, random arcs are used for modeling successive homogeneous parts in strokes with their length together, while random vertices are used for modeling their connected shapes.

2.4. Hierarchical random graph representation

After the concept of random graphs was introduced by Wong et al. [5,7] to deal with both the probabilistic and the structural aspects of relational data, it has been extended to a 2-layer [3] and a multi-layer random graph [9] for the description of complex objects with hierarchy. Since hierarchical random graph representation can reduce a complex object to manageable size by decomposing it into sub-objects containing lower complexity, it has been applied to the recognition of complex character sets with many strokes and a lot of characters such as Chinese characters.

At the first layer of the HRG proposed in this paper, character models composed of sub-character models are located. This layer is optional according to target character sets, for it is unnecessary to divide simple characters such as digit and English alphabet into sub-characters. In the next layer, sub-character models composed of primitive stroke models are located. In the bottom layer, primitive strokes are represented with a set of connected points.

The upper two layers are constructed with random graphs, representing relationship between sub-characters and between primitive strokes, respectively. In the first layer, a vertex of a character model corresponds to a sub-character, and an arc corresponds to the relation between the linked sub-characters. In the second layer, a vertex of the sub-character model represents a primitive stroke, and an arc represents the relationship between primitive strokes.

Modeling mechanism and the features used in the first and the second layers are same. In order to describe simply but effectively the relationship between two objects at each arc, we introduce reference-point-based (RP) description, which describes the relation between two objects simply by means of the relation between points. Center point is an exemplar referred frequently to specify positional relationship. To describe the relationships more precisely, we define nine types of reference points (RP-type) as in Fig. 1: center, left-center, right-center,

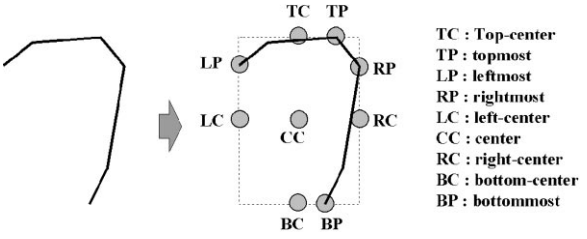


Fig. 1. Nine types of reference points.

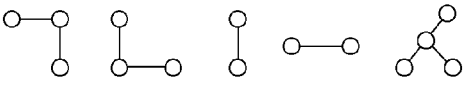


Fig. 2. Examples of primitive stroke models.

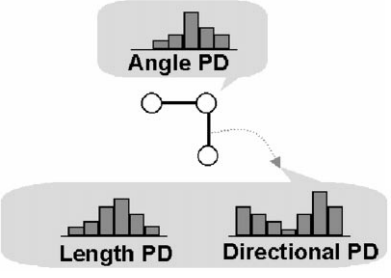


Fig. 3. Probability distributions of features in a primitive stroke model.

top-center, bottom-center, leftmost, rightmost, topmost, and bottommost. A pair of the RP-types is assigned to every arc as its attribute, and modeled with a probability distribution for the direction and length between two RP-types. Since automatic selection of the RP-type at each arc has not been studied, the RP-types of arcs is assigned manually.

A primitive stroke defined as a basic unit that constitutes a handwritten character should be a connected component. This is because primitive stroke models, which are made of chain graphs, have been designed to describe only a trajectory or connected trajectories of a writing instrument. Disconnected strokes varying with writing habits are to be managed in the upper layers, but broken strokes by noises are to be managed in this layer by means of gap filling. A vertex of a primitive stroke model corresponds to a point of input strokes, while an arc corresponds to a sequence of points. Therefore, in a word, a primitive stroke model describes a part of a connected component of a character. Examples of the primitive stroke models are given in Fig. 2. As shown in Fig. 3, arcs of a primitive stroke model have probability distributions of the predefined features, which include the length of strokes

First Consonant (FC)	ㄱ, ㄴ, ㄷ, ㄹ, ㅁ, ㅂ, ㅅ, ㅇ, ㅈ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ, ㄲ, ㄸ, ㅃ, ㅆ, ㅉ
Last Consonant (LC)	ㄱ, ㄴ, ㄷ, ㄹ, ㅁ, ㅂ, ㅅ, ㅇ, ㅈ, ㅊ, ㅋ, ㅌ, ㅍ, ㅎ, ㄲ, ㅆ, ㄳ, ㄵ, ㄶ, ㄺ, ㄻ, ㄼ, ㄽ, ㄾ, ㄿ, ㅀ, ㅄ
Vertical Vowel (VV)	ㅏ, ㅐ, ㅑ, ㅒ, ㅓ, ㅔ, ㅕ, ㅖ, ㅣ
Horizontal Vowel (HV)	ㅗ, ㅛ, ㅜ, ㅠ, ㅡ

Fig. 4. Groups of graphemes in Hangul.

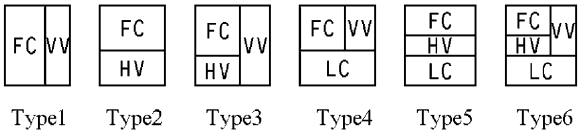


Fig. 5. The combining structure of graphemes in Hangul: six types.

and the direction of strokes. The vertices also have probability distributions for their features. We used the angle between two connected arcs as a feature of vertices.

3. Hierarchical Hangul model

As explained in many previous studies [10,11], a character in Hangul, which we call a syllable, is composed of two or more graphemes on 2D plane. (see Fig. 4). Even though the number of basic graphemes is small, which is 24, the number of syllables generated by the combination of these graphemes is in the order of 10,000. As shown in Fig. 5, all the syllables are classified into six types according to the construction rule of graphemes, and each grapheme can be divided into the primitive strokes. Thus, whole set of Hangul characters can be represented with a three layered hierarchy as shown in Fig. 6: primitive strokes, graphemes, and syllables for each layer. Simple descriptions about each layer and model will be given in the following subsections.

3.1. Primitive stroke model

The primitive stroke models to constitute grapheme models are placed in the bottom layer of the hierarchical Hangul model. Analyzing the shape of handwritten Hangul, we constructed 47 primitive stroke models representing horizontal shape, vertical shape, and somewhat complex shape of graphemes. The primitive strokes with similar shape, though derived from different graphemes, are modeled in one primitive stroke model. That is, the primitive stroke models with similar shape are structurally tied in the hierarchical Hangul model so that we can reduce the number of primitive stroke models.

3.2. Grapheme model

The number of graphemes necessary to constitute all the syllables is 44, where compound graphemes are included. In fact, much more grapheme models are required if we set apart the graphemes by their position in syllables. For instance, a grapheme 'ㄱ' can be found many times in the internal nodes of the hierarchy in Fig. 6. However, similar to the case of primitive stroke models, we reduce the number of models by tying the models of a label structurally. On the other side, to deal with variation of graphemes due to different writing styles, multiple grapheme models are designed for a grapheme according to its variation.

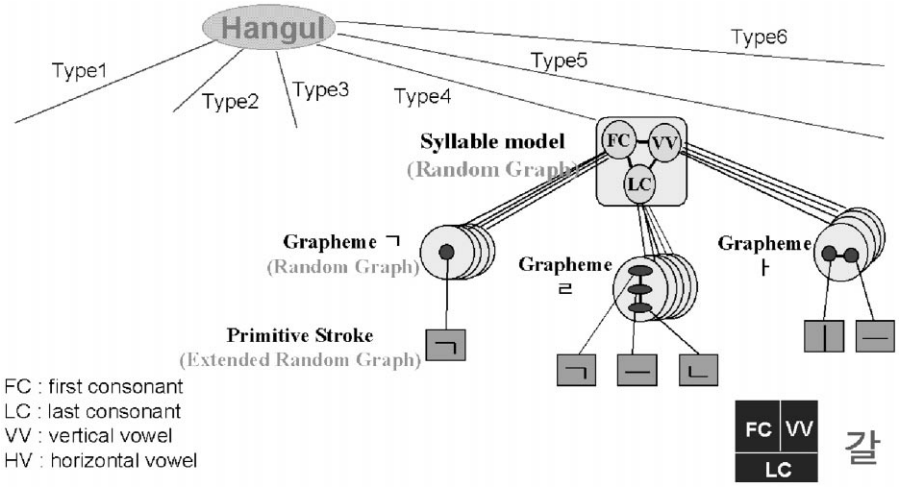
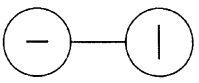


Fig. 6. Hierarchical graph representation of Hangul.



(a)

Type		Attributes
Arc	1	Relation types : RP – CC
Vertex	1	Primitive stroke '–'
	2	Primitive stroke 'l'

(b)

Fig. 7. Grapheme model '┤'. (a) Grapheme model '┤'. (b) Attributes of grapheme model '┤'.

A vertex of a grapheme model corresponds to a primitive stroke, and an arc represents relationship between primitive strokes at reference points indicated by the RP-types. Some of the grapheme models consist of only one vertex. An example of a grapheme model is given in Fig. 7, where '┤' model has two vertices and one arc: One vertex is for a vertical stroke and the other is for a horizontal stroke, and the arc is for the relation between the two vertices.

3.3. Syllable model

In a syllable model of the first layer, a vertex describes graphemes, and an arc describes the relationship between graphemes. Since a syllable model stands for all the syllables following the same grapheme combination rule drawn in Fig. 5, a vertex of a syllable model should be matched with a grapheme included in the grapheme set specified by the attribute of the vertex. The relationship between graphemes is represented with discrete probability distributions for the direction and length between them in arcs. In Fig. 6, a syllable model corresponding to the 4th type of Hangul is illustrated. It has three vertices with their attributes of first consonant, vertical vowel, and last consonant, respectively, and also has three arcs expressing the relation between the graphemes.

4. Construction of attributed graph

In this section, the procedure making an attributed graph from a handwritten character image is explained, including stroke extraction, feature encoding, gap filling, and attributed graph generation. Apart from the previous studies, where a hierarchy of an attributed graph is constructed by grouping strokes before recognition, our system does not make any decision about the hierarchy of an attributed graph, which is just a by-product obtained by matching with the hierarchical model.

4.1. Stroke extraction

Since an attributed graph is constructed with the strokes extracted from a character image, preprocessing applied here is focused on getting consistent strokes tracing the original trajectories of writing instruments. First, to remove small noises on images Gaussian smoothing algorithm is applied. Then, a thinning algorithm is applied to the images. Particularly, we used gray-level thinning algorithm to get better skeletons, from which strokes are extracted.

4.2. Feature encoding

The extracted strokes are encoded into predefined features, with which attributes of arcs and vertices in attributed graphs are assigned. As in the case of online character modeling [12], we assumed that strokes could be approximated with piecewise line segments. The length of the segments, a sampling unit of strokes, is relatively given with a 12th of a character height, and each of the segment is encoded into 16-direction angle code. After the encoding, we can get chains of features as in Fig. 8. To match the features efficiently, consecutive segments with a same direction code are grouped into one segment specified by both direction code and the length code of it. In the two end points of a segment, joint angles between connected segments are represented with 16-angle code. All of these discrete-valued features are to

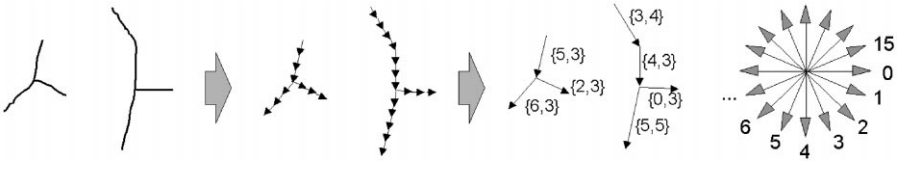


Fig. 8. Feature extraction from strokes.

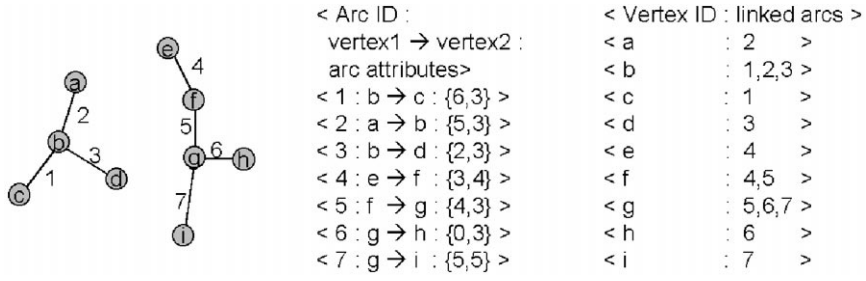


Fig. 9. An example of an attributed graph.

be regarded as outputs of discrete probability distributions in a model.

4.3. Gap filling

Since only a connected component of input strokes can be matched with a primitive stroke model, broken parts of primitive strokes should be connected before matching. However, it is very difficult to decide whether two separated strokes should be connected only with the local analysis of them. Thus, in this step, the decision for the connection of broken parts is not determined, but the candidates for the connection are generated so that they may be confirmed later. Let us call the candidates white-segments, which are added on where the distance between the end points of two strokes is less than a given threshold. It should be noted that whether a white-segment is a part of a character is not determined until the recognition is finished. The final matching result contains only minimum length of white-segments required to interpret an input character. This is because penalties are imposed on the matching probabilities in proportion to the length of the white-segments when white-segments are used for a matching. The other unused white-segments are ignored and have no effects on the results. This approach can reduce errors caused by a hard decision at the preprocessing step. Same idea was applied to the recognition of multifont printed characters [1].

4.4. Attributed graph generation

Basically, a line segment and two end points of it correspond to an arc and vertices of an attributed graph, respectively. Every arc has two attributes. One is a directional code and the other is a length code of a line segment. On the other hand, attributes of a vertex are angle codes between adjacent arcs and the number of linked arcs. An important role of vertices is that they are preserving the structure of the graph. Especially, since the strokes branched from a point are also placed on 2D plane, the relative order of them in clockwise or counter-clockwise should be preserved. By imposing this restriction, we can reduce the matching complexity (see Fig. 9).

5. Recognition system

The recognition in statistical approaches can be formulated into the problem that finds a model producing maximum probability given an input data. Let M_i be a model and X be an input graph. Then, the recognition means finding a model $\hat{M}$ to maximize the a posteriori probability given the observation graph X, i.e.,

$$\hat{M} = \arg\max_{M_i} P(M_i|X).$$

Using the Bayes' Rule, the a posteriori probability $P(M_i|X)$ can be written as

$$P(M_i|X) = \frac{P(X|M_i)P(M_i)}{P(X)}.$$

Since $P(X)$ is independent of M_i, the maximum a posteriori(MAP) decoding rule can be rewritten as

$$\hat{M} = \arg\max_{M_i} P(X|M_i)P(M_i),$$

where $P(X|M_i)$ is the probability that the model generates output X, and $P(M_i)$, the second term, is the a priori probability of the model. With the assumption that $P(M_i)$ is known, we have to seek nothing but $P(X|M_i)$ in order to get $\hat{M}$. $P(X|M_i)$ can be obtained from the matching of the input graph X with a model M_i.

The recognition process is depicted in Fig. 10. As shown in the figure, it is divided into two main parts: attributed graph construction and hierarchical graph matching. Attributed graph construction includes stroke extraction, feature encoding, gap filling, and graph generation, while hierarchical graph matching includes primitive stroke matching, grapheme matching, and syllable matching. Note that the matching process is adjusted to Hangul recognition. Brief description about the recognition process is following. First, an input character

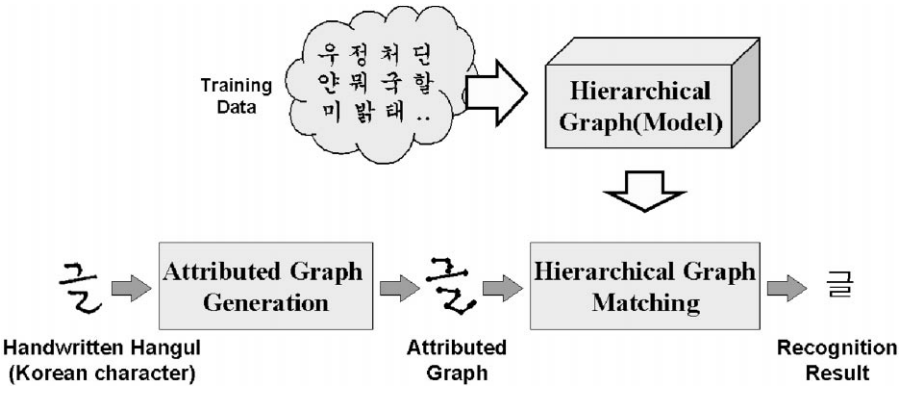


Fig. 10. Recognition process.

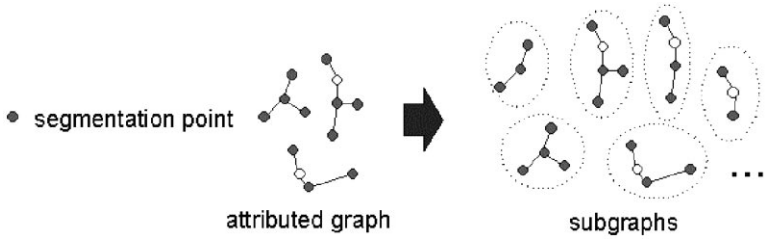


Fig. 11. Primitive stroke spotting.

image is converted into an attributed graph. Next, primitive strokes constructing graphemes are spotted from the attributed graph. That is, primitive stroke models are matched with subgraphs of the attributed graph. At this time, the system produces many plausible primitive stroke candidates that may be used later to construct graphemes. Then, grapheme matching is executed to generate grapheme candidates. Finally, by combining the grapheme candidates with a corresponding syllable model, a syllable producing the best matching probability is selected as a recognition result. In a word, the matching between an attributed graph and the hierarchical graph model is performed bottom-up. Other details will be explained in following subsections.

5.1. Primitive stroke model matching

The matching of an attributed graph with a primitive stroke model is multiple-to-one arc matching and subgraph matching as well (see Fig. 11). Thus, conventional algorithms of graph isomorphism are not applicable to this problem. The matching algorithm is recursive, and starts with a mapping from a vertex of a model to a vertex of an attributed graph. When a mapping found from all vertices and arcs of a model to vertices and arcs of an attributed graph, a matching result is obtained. At this time, one arc of a model can be matched with multiple arcs in an attributed graph. To reduce the complexity, the vertex of an attributed graph to be matched with an end vertex of a model is restricted to the segmentation points selected among the vertices of an attributed graph. As a consequence, not all the subgraphs of an attributed graph are considered, but only the subgraphs that can be partitioned by the segmentation points. Segmentation points include branch points, end points, and the corner points with sharp angle. In the matching, the configuration of arcs on 2D plane is maintained as mentioned in previous section. The primitive strokes spotted in this step will be used as candidates of a vertex in grapheme models with their matching probabilities.

5.2. Grapheme model matching

To generate the grapheme candidates, which are to be used for syllable matching, the grapheme model matching is performed (see Fig. 12). A grapheme candidate is generated together with its matching probability when the mapping from vertices of a grapheme model to the primitive stroke candidates is found satisfying some constraints that no stroke should be doubly used. The primitive stroke candidates are fetched from the pool of primitive strokes that have been made in the previous matching step. After the grapheme matching, all plausible grapheme candidates are extracted with their

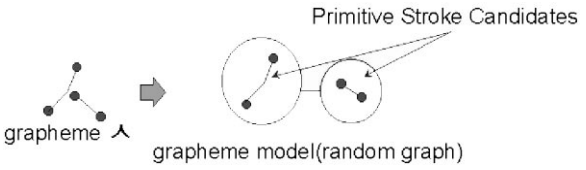


Fig. 12. Grapheme matching.

matching probabilities. The matching probability of a grapheme candidate is computed by multiplying all the matching probabilities of arcs and vertices in the grapheme model. The arc matching probabilities can be obtained, depending on the relationship of the vertices linked to the arc, from probability distributions in the model, and the matching probabilities of vertices are obtained from the matched primitive strokes.

To reduce the matching complexity, the concept of neighborhood systems is introduced. We define that the two strokes have neighborhood relationship if they satisfy one of the following conditions.

1. There is no stroke on the line connecting center points of the strokes.
2. End points of the two strokes are sufficiently close.

By extending the definition, we also define that two primitive strokes are neighbor if they contain the strokes with neighborhood relationship. Then, imposing the restriction that only the primitive stroke candidates having neighborhood relationship can be chosen for the matching of the two vertices linked by a random arc of a grapheme model, we can eliminate unnecessary calculation.

5.3. Syllable model matching

Syllable model matching is aimed at generating syllable candidates as recognition results of an input character so that the syllable with best matching can be selected as a final result (see Fig. 13). To generate the candidates, all possible combinations of grapheme candidates are considered in syllable models. Since a vertex of a syllable model indicates a group of graphemes, any grapheme candidate included in the grapheme group can be matched to the vertex only if the arcs of an attributed graph matched with the grapheme candidates are not overlapped. This means that every arc of an attributed graph should be matched with a grapheme just one time.

On the other side, for the arcs of an attributed graph excluded for the matching, the penalty is imposed on the matching probability, proportional to the length of the unmatched arcs. Thus, the more are the strokes unmatched, the less is the matching probability. To speed up the matching, grapheme candidates are sorted according to their matching probabilities and labels, and the candidates with higher probabilities are matched first. After the matching is finished, we can get the segmented graphemes, as well as the recognition result.

6. Parameter estimation

It has been studied to generate models automatically by synthesizing attributed graphs. However, in many cases it is ambiguous to determine the number of models and their structure automatically by synthesizing various attributed graphs that contain noise. The generated models tend to be too generalized or specialized to describe the attributed graphs. Thus, we think, when we have prior knowledge enough to design the structure, manual design of the structure works better than automatic construction of it.

In the proposed system, while the structure of the models was designed according to the variations of characters, the parameters of the models have been estimated from the training data by EM algorithm. Especially, for the training of each model in the hierarchical graph, we applied embedded training technique. In fact, in order to estimate parameters of each graph model in the hierarchy it is required that all attributed graphs of syllable data should be segmented into graphemes and then into primitive strokes. Since this manual segmentation needs laborious work and may cause inconsistent segmentation, embedded training was adopted.

The proposed embedded training estimates all the parameters of the models in the hierarchical graph at the same time. First of all, an attributed graph of a syllable is matched with the hierarchical graph to find the best partition of the syllable. That is, with the current model parameters, new partition of an attributed graph is obtained from the best matching of an attributed graph with the hierarchical graph. Then, the segmented results

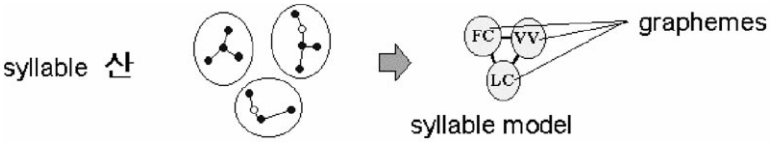


Fig. 13. Syllable matching.

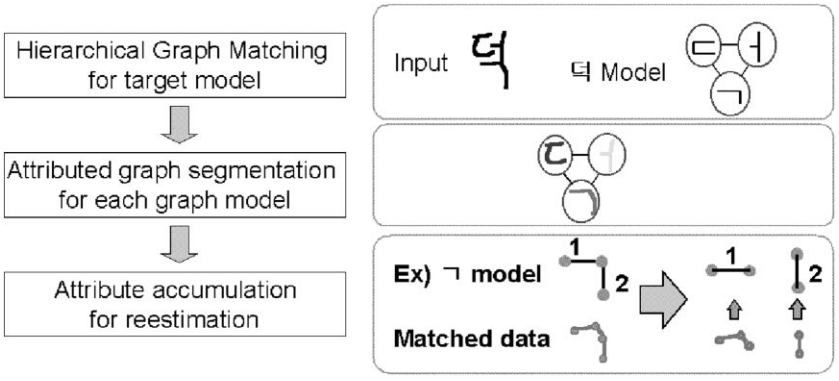


Fig. 14. Grapheme segmentation procedure for parameter estimation.

are accumulated in the attributes of corresponding models so that model parameters can be reestimated with respect to the accumulated attributes after the training data are all consumed. This training procedure is iterated until model parameters converge. Grapheme segmentation procedure is depicted in Fig. 14.

In the matching for training, only the models related to an input syllable are considered. Thus, our training method can be categorized into supervised learning. The syllable data unmatched with target models are excluded in the reestimation. Since model parameters converge to local minimum depending on initial parameters, the model parameters assigned manually are preferable to those given randomly at first time.

7. Experimental results

As an application of the proposed HRG, we have developed Hangul recognition system, where a hierarchical Hangul model is introduced. In the hierarchical Hangul model, the number of models for a grapheme is determined depending on its shape variation. While only one model is sufficient to represent simple graphemes such as 'ㄱ', 'ㄴ', etc., other graphemes containing great varieties such as 'ㅁ' and 'ㅂ' require more than two models. The total number of grapheme models used is 73, which is less than three models per grapheme. The model parameters have been automatically estimated from the training data by applying EM algorithm and embedded training, as explained in Section 6. The structure and initial parameters of the models were given manually with the help of human knowledge.

The recognition experiments were conducted with handwritten Hangul characters of the KU-1 database [13] containing 1000 sets of 520 unconstrained handwritten characters with different labels. To ascertain how much training data is needed to estimate the parameters of the proposed Hangul model with a lot of discrete probability distributions to be estimated, we investigated the change of recognition rate as a function of the size of the training data. In general, when the training data is insufficient to estimate parameters, recognition rate of test data becomes low, but that of the training data becomes high. This situation is called specialization. On the other hand, as the training data keeps increasing, assuming that the data are independently and identically distributed, the recognition rate of test data becomes higher bounded by that of the training data while the recognition rate of the training data becomes lower bounded by that of test data. Ideally, as the size of training data goes infinity, the two recognition rates converge the same value.

In our experiment, as the training data is increased roughly two times, the recognition rate of test data is observed. Twenty sets from the KU-1 database are used for testing, and 6, 12, 25, 50, 100, and 200 sets are used for training in the order of the size. So, 10,400 characters are used for preliminary testing, and from 3120 to 104,000 characters are used for training. Experimental results are given in Fig. 15, where recognition rates for the training data and test data are shown with their recognition rates up to the second and the fifth candidates. As shown in Fig. 15, the recognition rate of the training data gets lower while that of test data gets higher as the training data size becomes larger. With respect to the convergence of the recognition rate for test data and the small difference of the recognition rate between the training ones and test ones, 200 sets from the database seems to be enough to estimate the proposed Hangul model parameters. Therefore, we used 200 sets for training, and 100 sets from the rest for testing. That is, 104,000 characters were used for parameter estimation, and 52,000 characters were used for testing.

For the testing with 52,000 test data, when target syllables were restricted within 520 syllables, 90.4% recognition rate was achieved, while 86.3% recognition rate achieved when the recognition targets were 2350

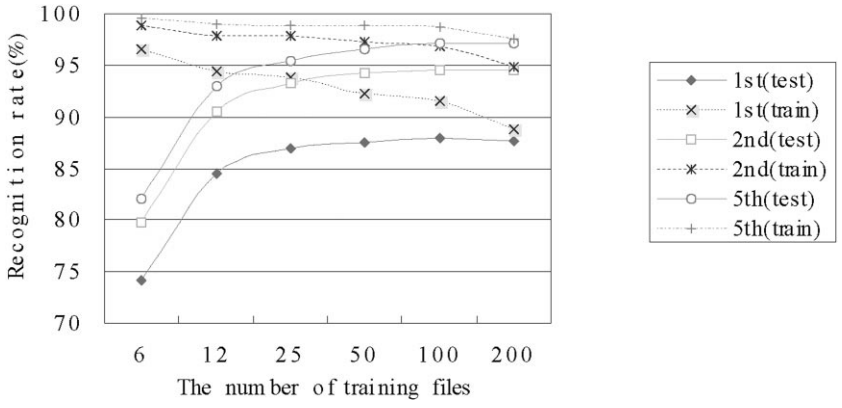


Fig. 15. Recognition rate for training data and test data as a function of the number of the training data files.

Number of categories (char.)		11172	2350	520
Recognition Rate (%)	1st Candidate	84.5	86.3	90.4
	2nd Candidate	91.9	93.1	95.4
	5th Candidate	95.7	96.4	97.2

Fig. 16. Experiments varying the number of categories for recognition.

syllables. Since our approach is grapheme-based, all syllables in Hangul can be recognized, although they are not in the training data, only if their graphemes are involved. The recognition rate up to the second candidate was 95.4% with 520 target syllables, and 93.1% with 2350 target syllables. Experimental results for varying the number of target syllables are given in Fig. 16.

Examples of correctly recognized data are given in Fig. 17. In addition, thinned images and segmented images for some data with correct recognition are listed in Fig. 18. As shown in the figures, handwritten Hangul characters with shape variations and touching of graphemes can be recognized successfully. The experimental results show that the proposed method can accommodate large shape variations with only a small number of models. Moreover, departing from many other structural systems that tend to be sensitive to noisy strokes, the proposed system can resist small noises in an input image and some distortion caused by thinning algorithm.

Fig. 19 illustrates recognition results and their segmented graphemes. As shown in the figure, depending on the recognized result labels, an input syllable is interpreted as a combination of different graphemes with some noises, which lower the matching probability in proportion to their length. There may be so many syllables generated for the recognized candidates of an input character. Among the syllables, the best one with maximum matching probability is selected.

The causes of errors have been investigated from about 2500 test data. According to the error types, they can be

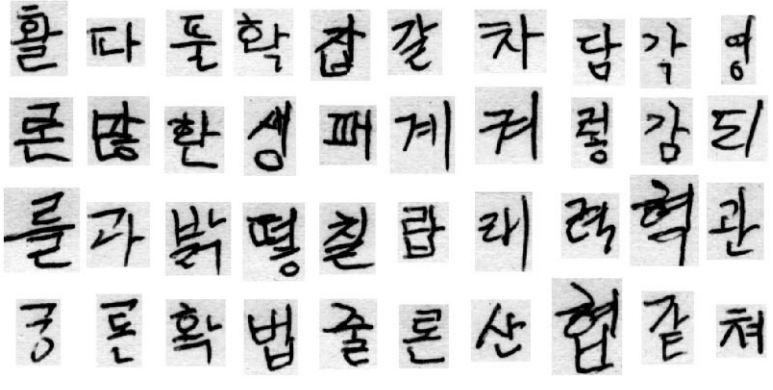


Fig. 17. Examples of correctly recognized data.

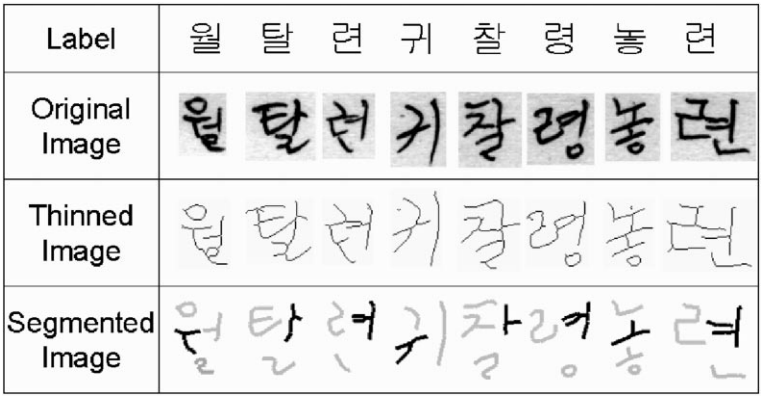


Fig. 18. Comparison of original images, thinned images, and segmented images.

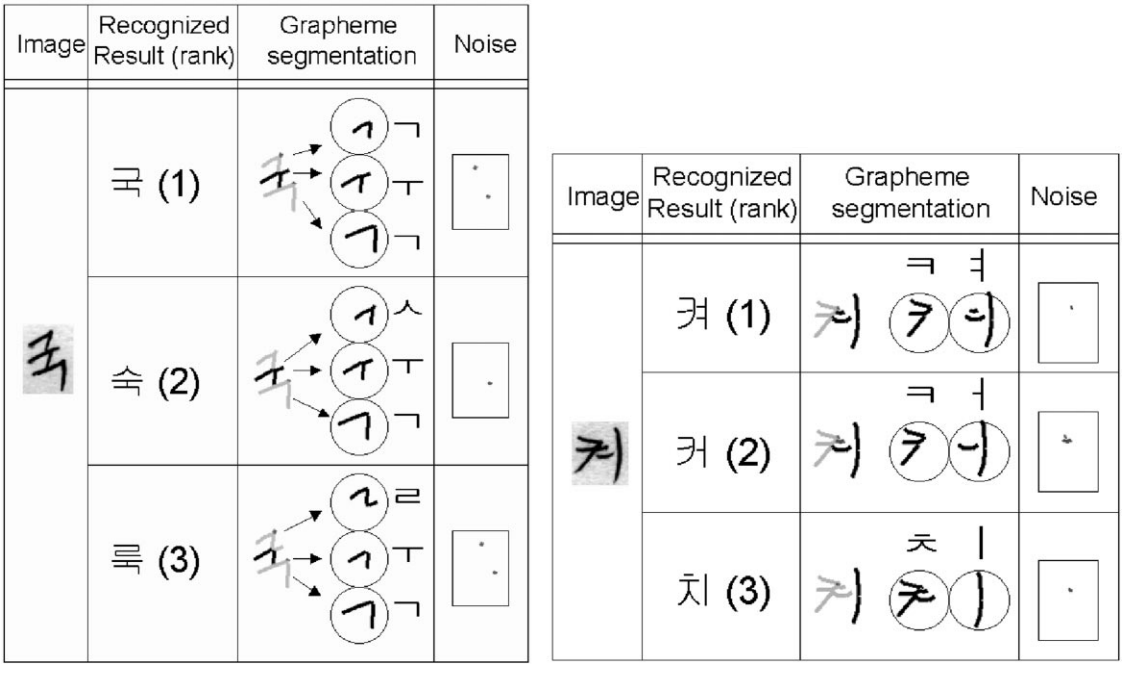


Fig. 19. Recognized results and their segmented graphemes.

classified into 10 types as listed in Fig. 20, where the percentage of each error type is also given. Since it is sometimes very obscure to decide the type of errors, and the ratio of the causes of errors is varied with writers, it is better to catch the tendency of errors from the analysis in Fig. 20, than to get the exact percentages of the errors. Here are some explanations about the error types.

(1) *Insufficient modeling*: There are two kinds of errors in this type: one is insufficient grapheme modeling, and the other is insufficient relationship modeling between graphemes. Fig. 21(a) shows the incorrectly recognized examples due to the grapheme shape confusion. On the other hand, in the case of Fig. 21(b), the segmented graphemes look well shaped even for the incorrectly segmented ones when they are given separately. To get better results for the confusing characters, it is necessary to reinforce the relationship modeling between graphemes.

(2) *Grapheme* '□' *and* '○' *confusion*: Parts of a handwritten grapheme '□' would be smoothly curved rather than straight, while a handwritten grapheme '○' would have sharply curved points. So, it is very difficult to find general rules or features to tell them

Cause of recognition errors	Percentage	How to improve
1. Insufficient modeling	27.2%	More features to discriminate
2. 'ㅁ' and 'ㅇ' confusion	17.9%	(adding new features)
3. Information loss by preprocessing	9.5%	Robust preprocessing
4. Small segments & Noise	8.8%	(utilizing width of strokes, etc.)
5. Penalty for unmatched segments	14.3%	Matching algorithm improvement
6. Missing segmentation points	5.1%	
7. Matching algorithm	2.9%	
8. Similar shape	8.8%	More information required
9. Stroke overlapping	5.1%	(such as contextual knowledge)
10. Truth error	0.4%	
Total	100.0%	

Fig. 20. Causes of recognition errors.

apart. Moreover, thinning algorithms would distort graphemes to make it impossible even for a man to identify them correctly as '□' or 'ㅇ'. To reduce this type of errors, a pair-wise discrimination algorithm is needed (Fig. 22).

(3) *Information loss by preprocessing*: After preprocessing algorithm applied, some characters become more confusing because of informative features missed in the contour of strokes. Examples are given in Fig. 23.

(4) *Small segments and Noise*: Since the width of strokes is not considered when attributed graphs are generated, small noises in background would affect recognition results as in Fig. 24(a).

(5) *Penalty for unmatched segments*: If the penalty for unmatched segments is too large, the recognizer is likely to use as many strokes of an input character to match with graph models, even allowing much distortion. On the contrary, if the penalty is too small, the recognizer is likely to ignore as many strokes as possible. In fact, when the penalty is adjusted, some misrecognized characters can be correctly recognized but some of the others may produce incorrect results. Therefore, to achieve better recognition performance, we think, not only the length of unmatched segments, but also their position in the character should be taken into account.

(6) *Missing segmentation points*: For efficient matching, segmentation points are selected from input strokes. But, failing in detecting necessary segmentation points may result in recognition errors (Fig. 24(b)).

(7) *Matching algorithm error*: While some heuristics applied to the recognition system get the matching faster, they also may be cause of the recognition error.

(8) *Similar shape*: Some handwritten characters are too similar to be discriminated (Fig. 25).

(9) *Stroke overlapping*: When strokes are overlapped, some of them may disappear or have large distortion after thinning. In this case, it seems impossible to recognize the characters correctly only with the thinned images (Fig. 26).

(10) *Truth errors*: This means labeling error in the database.

8. Concluding remarks

Handwritten characters generated from trajectories can be effectively described with trajectories, or strokes, and their relationship. Reflecting such characteristics of handwritten characters, hierarchical random graph (HRG) representation for handwritten character modeling was presented in this paper. The proposed HRG differs from the previous methods in which the shape of strokes is modeled stochastically as well as hierarchical relationship of them so that the recognition can be thoroughly carried out in a probabilistic framework.

In the bottom layer of the HRG, chain graphs are introduced to describe various shapes of strokes. Especially, the length probability distribution in a random arc of a chain graph, similar to the duration density of a variable duration HMM, enables variable length handling so that variable length of connected strokes can

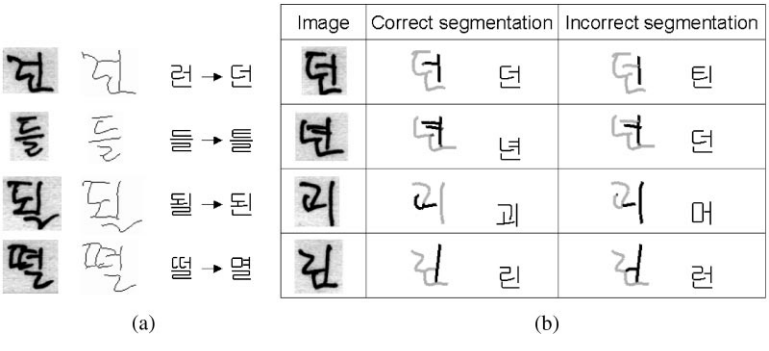


Fig. 21. Misrecognition examples due to insufficient modeling. (a) Grapheme modeling should be improved. (b) Relationship modeling between graphemes should be reinforced.

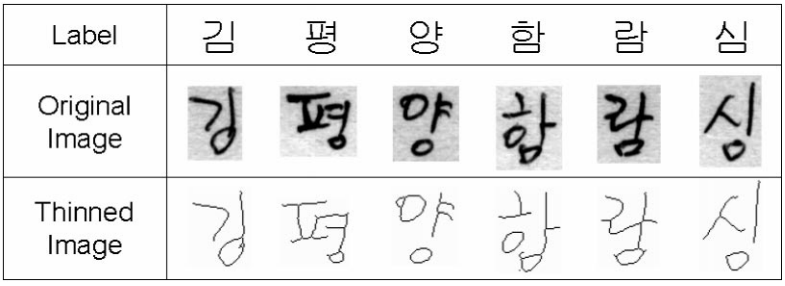


Fig. 22. Misrecognition examples due to '□' and 'ㅇ' confusion.

Fig. 23. Misrecognition examples due to the information loss by preprocessing; images are pairs of an original image and a thinned image; labels are pairs of true label and recognized label.

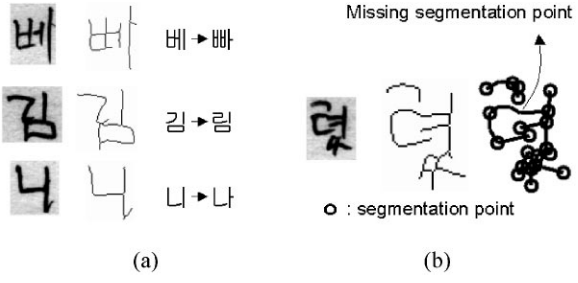


Fig. 24. Misrecognition examples due to; (a) small noise; (b) missing segmentation points.

be matched with a random arc. Since all the points in input strokes can be considered as a matching point of a vertex of a chain graph, the mapping of strokes between an input and a model is very flexible.

Based on the HRG, handwritten Hangul recognition system has been developed, where a hierarchical Hangul model composed of primitive stroke models, grapheme models, and syllable models has been constructed and used. From the recognition experiments conducted with handwritten Hangul characters, it has been shown that the proposed method can accommodate large shape variations with only a small number of models and ignore

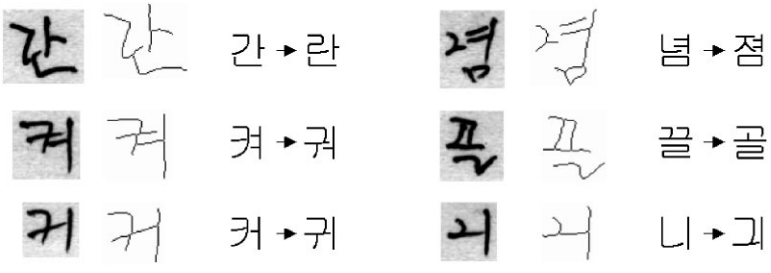


Fig. 25. Examples of characters with similar shape.

Fig. 26. Misrecognition examples due to overlapped strokes.

small noises in an input image. It has been also shown that an automatic parameter estimation algorithm implemented to estimate model parameters of the HRG by applying EM algorithm and embedded training can estimate parameters of the HRG effectively.

References

[1] J. Rocha, T. Pavlidis, A shape analysis model with applications to a character recognition system, IEEE Trans Pattern Anal Mach. Intell. 16 (4) (1994) 393–404.
[2] J. Rocha, T. Pavlidis, Character recognition without segmentation, IEEE Trans Pattern Anal. Mach. Intell. 17 (9) (1995) 903–909.
[3] L.H. Chen, J.R. Lieh, Handwritten character recognition using a 2-layer random graph model by relaxation matching, Pattern Recognition 2 (11) (1990) 1189–1205.
[4] S.W. Lu, Y. Ren, C.Y. Suen, Hierarchical attributed graph representation and recognition of handwritten Chinese characters, Pattern Recognition 24 (7) (1991) 617–632.
[5] A.K.C. Wong, D.E. Ghahraman, Random graphs: structural-contextual dichotomy, IEEE Trans. Pattern Anal. Mach. Intell. 2 (4) (1980) 341–348.
[6] A.P. Dempster, N.M. Laird, D.B. Rubin, Maximum likelihood from incomplete data via the EM algorithm, J. Roy Statist. Soc. 39 (1977) 1–38.
[7] A.K.C. Wong, M. You, Entropy and distance of random graphs with application to structural pattern recognition, IEEE Trans. Pattern Analy. Mach. Intell. 7 (5) (1985) 599–609.
[8] L.R. Rabiner, A tutorial on hidden Markov models and selected applications in speech recognition, Proc. IEEE 77 (1989) 257–285.
[9] D.S. Seong, H.S. Kim, K.H. Park, Formal definition and entropy calculation of hierarchical attributed random graph, Pattern Recognition Lett. 13 (1992) 545–555.
[10] Y.B. Kwon, K.Y. Lee, Y.B. Lee, A novel Hangul recognition algorithm based on stroke extraction, Proc. ICDAR 1 (1991) 272–280.
[11] H.J. Kim, P.K. Kim, Recognition of off-line handwritten Korean characters, Pattern Recognition 29 (2) (1996) 245–254.
[12] B.K. Sin, J.H. Kim, Ligature modeling for online cursive script recognition, IEEE Trans. Pattern Anal. Mach. Intell. 19 (6) (1997) 623–633.
[13] D-I. Kim, S-W. Lee, An automatic evaluation of handwriting qualities for off-line handwritten Hangul character database KU-1. Proceedings of the 25th Korea Information Science Society Conference, Vol. 25 (1) (1998) 707–709.

About the Author—HO YON KIM received the B.S. degree in computer science from Yonsei University in 1992, and M.S. and Ph.D. degree in computer science from KAIST in 1994 and 1999, respectively. He was a visiting researcher at NHK research center for two months in 1997, a visiting researcher at SIEMENS for three months in 1999, and has been working for ETRI since 1999. His research interests include pattern recognition, neural networks, machine learning, etc.

About the Author—JIN H. KIM received the B.S. degree in engineering from Seoul National University in 1971, and M.S. and Ph.D. degree in computer science from University of California, Los Angeles in 1979 and 1983, respectively. He was a research engineer at Korea Institute of Science and Technology (KIST) from 1973 to 1976, and engineering programmer at County of Orange, California, USA, from 1976 to 1977, and a senior staff member in computer science at Hughes Artificial Intelligence center, Calabasas, California, USA, from 1981 to 1985. He joined the faculty of KAIST in 1985. He was a visiting scientist at IBM Watson Research Center for the 1990. From 1995 to 1999, he was the president of Korea R & D Information Center (KORDIC). He was on several editorial boards such as International Journal of Information Processing & Management, International Journal of Autonomous Systems and International Journal of Chinese and Oriental Language Processing. His research interests include pattern recognition, intelligent man machine interface and AI for Education.