Studies on Online Multi-Stroke Character Recognition

蔡, 文杰
九州大学システム情報科学府情報知能工学 / (株)オーリッド(O-RID)

https://doi.org/10.15017/25255

出版情報：九州大学, 2012, 博士（工学）, 課程博士
バージョン：
権利関係：

2012年度 博士論文

Studies on Online Multi-Stroke Character Recognition

2012年6月25日

情報知能工学専攻

(学籍番号: 3IE09011E)

蔡　文杰

九州大学大学院システム情報科学府

Abstract

This thesis copes with the issues of online multi-stroke character recognition, and focuses on two topics related to the recognition task: (i) This thesis reviews the approaches for solving the stroke-order variation problem, and conducts a comparative comparison for clarifying the relative superiority of five methods of stroke correspondence. The five methods are cube search (CS), bipartite weighted matching (BWM), individual correspondence decision (ICD), stable marriage (SM), and deviation-expansion model (DE). (ii) This thesis proposes a novel method to reduce the time and spatial complexity of CS remarkably, by dividing Kanji character reference pattern into radical reference patterns, and reorganizing the stroke correspondence search graph of CS.

Acknowledgements

There are several people who I need to thank for helping me get to this point, and they could never all fit on a single page of a dissertation.

First I would like to give heartful thanks to my supervisor, Prof. Seiichi Uchida, Kyushu University, for his advice during my doctoral research endeavor for the past years. Without his guidance and determination, I would not be the student that I am today. As my supervisor, he has constantly forced me to remain focused on achieving my goal. His observations and comments helped me to establish the overall direction of the research, and to move forward with investigation in depth. I thank him for providing me the opportunity to work with a talented team of researchers.

I would like to express my appreciation to my former supervisor, the late Hiroaki Sakoe, Emeritus Professor of Kyushu University, who helped me to start my doctoral research. He passed away due to sickness caused by his hard work of research. He led me into the world of character recognition, and taught me to know how to be a real researcher.

I would like to express my sincere thanks to my advisor, Prof. Rin-ichiro Taniguchi, Kyushu University, and Prof. Koichi Kise, Osaka Prefecture University. They gave me so many good suggestion, to help me to accomplish my thesis smoothly. I also would like to thank Prof. Ken'ichi Morooka, Kyushu University, for his important comments and suggestions during his reviewing of my thesis.

Lastly, I would like to thank staff members of my laboratory and company. In laboratory, members helped me a lot when I was not in university. In company, my company and co-workers always supported me when I was busy in doctoral research.

Contents

Chapter 1

Introduction

1.1 Background

Handwriting recognition has been a popular area of research for several decades under the field of pattern recognition and image processing, because of its various application potentials like postal automation, bank cheque processing, automatic data entry, etc. For example, for the automatic data entry, which is rapidly growing with the widespread development of various forms of electronic input devices, handwriting recognition is faster than keyboard for multi-stroke large-alphabet languages like Chinese or Japanese Kanji [1, 2].

When writers write their own character styles, it brings variations in handwriting characters. The variations, including shape, size, stroke-number, stroke-order, tilting angle, and writing speed, etc. [3], are most difficult in recognition. Figure 1.1 shows an example of different writing styles. It should be noted that the two handwriting Chinese characters "king" have different stroke-orders and stroke-numbers. The numerals in this figure indicate the stroke-orders. Thus, for a practical sys-

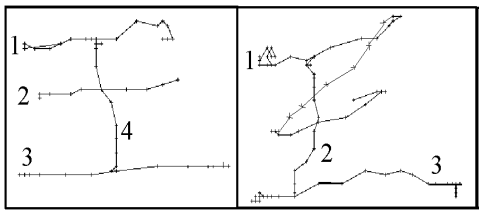


Figure 1.1: Two handwriting Chinese character "king" with different stroke-orders and stroke-numbers.

tem of handwriting recognition, it needs to cope with the natural handwriting of a wide range of different writers and writing styles. Meanwhile, a practical system of handwriting recognition should be convenient for operation, scalable to the device's capability and accurate. In fact, the variations in the above mentioned features among characters, such as stroke-order and stroke-number, are important factors for improving character recognition accuracy.

Handwriting recognition can be divided into two categories: online and offline. An important difference between online and offline handwriting recognition (or OCR) is the fact that spatial and temporal information is available in the former, while only spatial information is available in the latter. For the online case, basically it acquires a string of (x,y) coordinate pairs from an electronic pen touching a pressure sensitive digital tablet, and the character patterns are expressed as the trajectory of pen tip motion. For the offline case, the patterns are expressed as images from scanners, digital cameras or other digital input sources, and recognized by conventional OCR techniques [1, 2, 4].

Although the offline and online handwriting recognition techniques have different approaches, they share a lot of common problems and solutions. For some solutions,

online data are converted to the form of offline data (image) and employ the methods of offline handwriting recognition to recognize the character patterns. Conversely, offline data can also be converted by line thinning to sequences of points similar to online data (but without the temporal information), then achieve character recognition by the methods of online handwriting recognition [2, 5, 6, 7, 8, 9].

This thesis copes with online multi-stroke character recognition. Online multi-stroke character recognition is gaining renewed interest because of the new development of new pen devices and their applications. Compared to several decades ago, we now have more accurate electronic tablets, more compact and powerful computers, and better recognition algorithms. The most common device is the electronic tablet or digitizer, typically with a resolution of 200 points/in, a sampling rate of 100 points/s. An indication of "inking" or pen-down, written by using a stylus that captures information of the pen tip, generally its position, velocity, or acceleration as a function of time [4]. An advantage of online devices is that they capture the temporal or dynamic information of the writing. This information consists of the stroke-numbers, the stroke-orders, the writing direction for each stroke, and the writing speed within each stroke. A stroke is the writing from pen-down to pen-up. Most online devices capture the trace of the handwriting or line drawing as a sequence of coordinate points.

Owing to the availability of both temporal stroke information and spatial shape information, online character recognition has a potential to yield higher accuracy than offline recognition. Online recognition also provides good interaction and adaptation capability because the writer can respond to the recognition result to correct the error or change the writing style. For desktop applications, online recognition is

well-suited to text entry for large alphabets like oriental languages (e.g., Japanese Kanji and Chinese), because the current design of the keyboard is meant for the Latin-based language input [1, 10]. Although there are many input methods based on this keyboard have been developed for oriental language entry, they still suffer for slow input or steep learning curve. If we use handwriting as an input, all of these obstacles will be omitted. However, the recognition of Chinese characters is different from western handwriting recognition and poses a special challenge, because current online recognition still suffers from several weaknesses involving sensitivity to stroke-order, stroke-number, and stroke characteristics variations [8].

A practical recognition system of online multi-stroke character should be stroke-order free to cope with characters with stroke-order variation. Although there are many researches for solving the stroke-order free recognition problem, the problem has not been solved yet. To cope with stroke-order variation, several approaches, such as multiple prototype approach, offline feature approach, and stroke correspondence approach, have been proposed. However, since no extensive comparative study has been made so far, none knows the relative characteristics and performance of the past approaches.

This thesis tries to clarify the relative superiority of stroke-order free recognition approaches from the viewpoints of recognition accuracy and efficiency. In particular, we focus on stroke correspondence approach that is theoretically the most reliable approach among several approaches, and try to conduct a quantitative and qualitative evaluation of five representative methods through an experiment. The five representative methods are cube search (CS) [11, 12], bipartite weighted matching (BWM) [13], individual correspondence decision (ICD) [14, 15], stable marriage

(SM) [16], and deviation-expansion model (DE) [17]. Furthermore, as an application example of the stroke correspondence methods, we consider personal identification by extracting the stroke-order of an online input pattern [18].

Cube search method (CS), one of the above mentioned five methods and proposed by Sakoe and Shin [11, 12], is an promising stroke-order analysis method for online character recognition. However, since CS is with $O(N \cdot 2^{N-1})$ time complexity for an N-stroke character, a more efficient method is hoped to improve the performance of CS, in case of the multi-stroke character with large N (e.g., Chinese character or Japanese Kanji character).

This thesis will propose a novel and efficient method based on radical decomposition to resolve the time complexity problem of CS. A Chinese character has an average of 8-10 strokes, the simplest characters have one stroke, and the most complicated characters have more than 30 strokes. While the stroke-numbers of characters are high, each character is made from about 500 kinds of component sub-characters (called radicals) in predefined positions and written in a predefined order. In character recognition, the radical-based approach attempts to decompose the character into radicals and classifies the character based on the component parts and their placement. We employ the radical decomposition strategy into CS and get exciting result [19, 20].

1.2 Purpose of the studies

For online multi-stroke character recognition, this thesis conducts a comparative study for clarifying the relative superiority of five methods of stroke correspondence,

and proposes a novel method to reduce the time and spatial complexity of CS.

1.2.1 Comparative study of stroke correspondence search methods for stroke-order variation

For stroke-order free online multi-stroke character recognition, stroke-to-stroke correspondence search between an input pattern and a reference pattern plays an important role to deal with the stroke-order variation. Although various methods of stroke correspondence have been proposed, no comparative study for clarifying the relative superiority of those methods has been done before. In this chapter, we firstly review the approaches for solving the stroke-order variation problem. Then, five representative methods of stroke correspondence proposed by different groups, including CS, BWM, ICD, SM, and DE, are experimentally compared, mainly in regard of recognition accuracy and efficiency. The experimental results on an online Kanji character database, showed that 99.17%, 99.17%, 96.37%, 98.54%, and 96.59% were attained by CS, BWM, ICD, SM, and DE, respectively as their recognition rates. Extensive discussions are made on their relative superiorities and practicalities. Based on our results, for the performance of recognition accuracy, relative superiority of CS and BWM over ICD, SM, and DE are established. Additionally, the application of these methods of stroke correspondence to personal identification is addressed.

1.2.2 An efficient radical-based algorithm for stroke-order free online Kanji character recognition

We investigate improvements of the promising stroke correspondence analysis algorithm of online multi-stroke character — CS, based on dynamic programming (DP) and graph search. By dividing Kanji character reference pattern into radical reference patterns, the stroke correspondence search graph is decomposed into intra-radical graphs and an inter-radical graph. This reorganization remarkably reduces the time and spatial complexity of stroke correspondence search DP. It also improves the recognition performance of the CS by prohibiting unnatural stroke correspondence between different radicals. Experimental results on an online Kanji character database showed significant improvements both in operation speed and recognition accuracy.

1.3 Organization of the thesis

The rest of this thesis is organized as follows. Chapter 2 gives a brief review on the processing of online multi-stroke character recognition. Chapter 3 detailedly intruduces the CS, a promising method of stroke correspondence. After that, in Chapter 4, five representative methods of stroke correspondence including CS are detailed and experimentally compared. Then, extensive discussions are made on their relative superiorities and practicalities. Additionally, the application of these methods to forensics is addressed. To improve the operational speed of CS that shows its relative superiority of accuracy over other methods in Chapter 4, Chapter 5 reviews the detail of CS and proposes a novel, efficient and stroke-order free

recognition algorithm based on CS and radical-based reference model. Chapter 6 gives some concluding remarks for the thesis.

Chapter 2

Review of online character recognition

2.1 History of online character recognition

The research of online character recognition started in the 1960s and has been receiving intensive interest from the 1980s. It is a very challenging research since the beginning of the 1960s, when the first attempts to recognize isolated handwritten characters were performed [6, 21, 22]. Since then, numerous methods have been proposed and tested. Since the 1990s, compared to the research in the 1980s, the research efforts aimed to further relax the constraints of handwriting, such as the adherence to standard stroke-orders and stroke-numbers for multi-stroke characters, and the restriction of recognition to isolated characters only. Especially for the oriental language characters with multi-stroke, like Japanese Kanji and Chinese, the target of online recognition in the 1990s has shifted from the character with regular writing to the character with fluent writing and stroke-order and stroke-

number variation.

During recent years, the task of online character recognition has gained a renewed interest and an immense importance in every day applications, mainly due to the increasing popularity of the personal digital assistant (PDA) and digital pen. For the past few years, electronic gadgets such as smart phones, PDAs and tablet PCs have become very popular, especially among the younger generations. The trend is even more obvious in Asian countries like Japan, Korea and China [23, 24]. As these devices become more personal and are made smaller, they reach some physical limitations for having keyboard. Also, data entry for scripts having large alphabet size like oriental languages is difficult when use keyboard. Thus, an alternate man-machine interface is necessary for these devices. The most natural way of entering data would be to use the communication skills of person developed for thousands of years, namely speech and handwriting. So, data entry using pen-based devices is gaining popularity in recent times. An even more important advantage of pen-computing to the oriental languages is that the current design of the keyboard is meant for the Latin-based language input. Although there are many input methods based on this keyboard have been developed, they still suffer for slow input or steep learning curve. If we use handwriting as an input, all these obstacles will be omitted.

The difficulty of designing a practical system of online character recognition is commonly explained as follows: First, for small devices like PDAs and smart phones, the limitations in computational power and memory size are disadvantageous to the system design for these devices. Second, for a writer independent solution, the system has to discriminate between a great variety of different writing styles of users [25]. Even more difficult for online recognition, the handwriting data that looks

similar in a graphical (i.e., offline) representation, can has a different sequential (i.e., online) representation. Thus, a practical handwriting recognition system requires to be convenient for operation, scalable to the device's capability, accurate, and can deal with the natural handwriting of a wide range of different writers and writing styles. Until now, various kinds of online recognizer have been developed. However, currently even the best recognizer does not yet perform sufficiently well as a primary text input method.

Online character recognition techniques have been widely implemented into commercial products in recent years. Besides the application of input of PDAs, smart phones, and tablet PCs, one of other interesting technologies is that it can provide high level of interactivity. In the last few years, recognition techniques of online multi-stroke character are applied to some instructional tools to help children learn to write. The aim of these systems is to help young children to become good writers with fluent movements and a good quality of writing in a shorter time, by using of the real time comparing results like stroke-orders or stroke-numbers between children's characters and templates [26, 27]. Another important application of online character recognition technique is writer verification to judge whether two character sets are from the same writer or not (writer verification), or classify a test character set to a nearest reference character set of known writer (writer identification) [2, 28, 29, 30].

Since the beginning of the 1960s, many pieces of work have been done focusing on online character recognition of Roman, Japanese, Chinese, and Korean scripts, and various approaches have been proposed by the researchers towards handwritten character recognition of them. It should be noted that, the handwriting recognitions of other languages like Indian (i.e., Bangla) and Arabic are also drawing increasing

attention in recent years [31, 32, 33, 34].

2.2 Handwriting data

The handwriting data of pen trajectory has special characteristic and format. It consists of two kinds of information: spatial information and temporal information. Online character data is typically a dynamic, digitized representation of the pen movement, generally describing sequential information about position, velocity, acceleration, or even pen angles as a function of time [25]. In recent years, the UNIPEN database [35] has become the most popular publicly available data collection in online character recognition research. UNIPEN represents a writing curve as a series of so-called pen-down and pen-up components. Each component contains a sequence of pen tip information sampled from the writer's pen movement, usually with regular interval in time. The pen-up and pen-down information is captured to distinguish the stroke between pen-down and pen-up.

Although the input device (tablet) also provides other information such as pen pressure, pen tilts, etc., basically the pen positions (x,y) and directional features are used as the primary information. For the pen position feature, it accepts a string of (x,y) coordinate pairs from an electronic pen touching a pressure sensitive digital tablet. For the directional feature, it can be extracted directly from the online trajectory.

For the data of large number of tablet digitizers, the main measuring precision is characterized by resolution, accuracy and sampling rate. A basic function of the digitizing tablet within a pen-computer is to detect the stylus on the writing

surface and measure its position at its nominal sampling rate. The sampling rate generally varies from 50-200 Hz depending on the application, typically sampled at 100 samples per second [3]. Finer resolution is received with the higher sampling rate, which can accurately measure the fast strokes while reverse is the case for rough resolution.

Based on such a handwriting data mentioned above, the online character recognition has a number of distinguishing features. These features can help online character recognition to get more accurate results and practicability than the offline character recognition [1]:

- *It is a real time process.* While the digitizer captures the data during the writing, the character recognition system can carry out the recognition with or without a lag. Furthermore, the writers can give immediate feedback to the recognizer for improving the recognition rate, as they keeps drawing the symbols on the tablet and observes the results.

- *Not only the spatial information, it also captures the temporal and dynamic information of the pen trajectory.* This information consists of the number and order of pen strokes, the direction of the writing for each pen stroke and the speed of the writing within each pen stroke.

- *Very little pre-processing is required.* The operations, such as smoothing, de-slanting, de-skewing, detection of line orientations, corners, loop and cusps are easier and faster with the pen trajectory data than on pixel images.

- *Segmentation is easy.* Segmentation operations are facilitated by using temporal and pen-lift (from pen-up to pen-down) information, particularly, for

handwriting characters.

2.3 Basic process of online character recognition

To output correct recognition results, a practical online character recognition system has some basic process. In a practical system, the input to the system is a sequence of handwritten character patterns. Prior to any recognition, the acquired data is generally preprocessed to segment the signal into meaningful units, reduce spurious noise, and normalize the various aspects of the trace. The handwriting sequence should be firstly segmented into character (or word) patterns according to the temporal and shape information. Often, the boundary between characters cannot be determined unambiguously before character recognition, so candidate character patterns are generated and recognized and the correct patterns are selected in contextual processing at the end of the process chain [36, 37, 38].

Generally, to recognize the segmented (candidate) character patterns, it involves the following steps: preprocessing, description, and classification [3, 23]. Preprocessing involves noise elimination, shape normalization, etc. Pattern description is also referred to as feature extraction, it represents the input pattern either statistically by feature vectors or structurally by various levels of primitives. Classification is classifying the input pattern into a character class predefined in a reference model database, and often decomposed into coarse classification and fine classification. The model database (also called reference database or recognition dictionary) contains the reference models or classification parameters for coarse classification and fine classification.

2.3.1 Segmentation

Segmentation refers to the processing that must be performed to get a representation of the various basic units that the recognition algorithm will have to process. It generally works at two levels. The first level deals with the whole message and focuses on, e.g., line detection, word segmentation as well as separating nontextual inputs (e.g., gesture commands, equations, diagrams, etc.) from text [39, 40]. At this level, the goal is to define spatial zones or temporal windows, or both, that allow the extraction of disjoint basic units. At the second level, the methodology focuses on segmenting the input into individual characters or even into sub-character units, such as strokes [2, 4, 41, 42, 43]. This processing is among the most challenging, particularly for the recognition of cursive script. In most cases, this segmentation is tentative and is corrected later during classification.

The accuracy of segmenting characters, especially for connected characters, is essential for the performance of a character recognition system. For oriental languages like Chinese, because the key for Chinese recognition is individual character recognition, usually the main effort is focused on segmenting pages into lines, then directly into characters. The word segmentation is generally ignored if semantic information (word lexicon) is not available. Since in Chinese, the characters are always written in "print fashion ", or not connected, in some sense segmentation is easier than in other languages such as English [7]. Two widely used techniques, vertical projection and connected component analysis, are able to help to cope with the problem. On the other hand, Chinese characters usually consist of more than one radical and even some radicals themselves are characters, and handwriting strokes of adjacent char-

acters may join. For western languages like English, generally word segmentation is important and the individual character segmentation is more challenging because of the cursive handwriting [7, 44].

2.3.2 Preprocessing

The isolated character data, directly collected from users are often incomplete, noisy and inconsistent. They are needed to be pre-processed before applying to the system in order to get the correct classification. All the ways (techniques) to refine the data suitable for analyzing are included under the pre-processing technique. Re-sampling, noise elimination, and shape normalization are the basic techniques of pre-processing. Different systems use a variety of different techniques, they vary from one script to another and depending on the goal as well.

Some of the existing techniques are explained below [3, 8, 21, 45].

- *Re-sampling:* The strings of sampled point coordinate as functions of time are recorded along the pen trajectory during the pen movement over the surface of the sensitive screen. This facilitates to track the number of strokes and their order within a character. The length of the strokes (number of coordinates in a string) varies even users write with the same speed. In order to achieve a constant number of coordinates in every string or in constant distance, re-sampling is necessary so that sample points are spaced evenly along the arc length of each stroke. Re-sampling also helps to reduce the anomalous cases such as having a large number of samples at the same position when the user holds down the pen at a point.

- *Noise Elimination:* A stroke inevitably contains noises, typically come from many sources. Hand fluctuation during writing and digitizing error of the input devices are the major sources. The noisy sequence in addition to the information sequence may not rigorously harm the character in offline graphical representation, while severely affect on online data sequence. The noise reduction techniques used in most systems are basically deal with smoothing, filtering, wild point correction, stroke connection, break corrections, etc.

- *Normalization:* The main purpose of normalization is to reduce the pattern variations caused by writer specific characteristics (e.g., speed, size, slant, and rotation), recording hardware (e.g., online ink resolution) and noises. Basically, recognition is better for only in case of nominal size of writing as well as standard orientation and a nominal slant. Common normalization procedures involve correction of baseline drift, compensation of writing slant, and adjustment of the script size, etc.

2.3.3 Feature extraction

Feature extraction is also referred to as pattern description. Quality feature selection can greatly decrease the workload and simplify the subsequent design process of the classifier. Features should contain information required to distinguish between the classes, be sensitive to irrelevant variability of the input, and also be limited to permit efficient computation of discriminant functions and to limit the amount of training data required. When writers write their own styles, it brings variations in handwriting characters, and causes lots of difficulties in recognition. It is easy

for classifier to recognize the symbol if it has feature with sufficient distinguishing characteristics. Different systems use different varieties of feature, have different accuracies and goals as well. The feature used in one system may not fit with the other systems. Features taken from the same input data can be variably used [3, 25].

Feature extraction often represents the input pattern either statistically by feature vectors or structurally by sequence of various levels of primitives which generally are strokes or line segments consisting of feature points, or feature points themselves [21]. One stroke can be divided into a few of line segments (or only one line segment). Line segment representation is especially suited to regular-style characters, because the regular-style characters are composed of mostly straight-line segments. Another primitive of the character is the standard strokes. Mostly in Chinese, Japanese, and Korean characters, the case is used. Since large number of symbols should be used to complete the characters of these languages, many standard strokes (or line segments) are defined as the basic components of the characters. In online character recognition, the directional feature of the line segment or the feature point is also a very important feature, and can be calculated directly by the starting and ending positions of line segment or the online pen points. Typically, the directional features are of eight types, coded from $0 \rightarrow 7$. Not only in the mentioned scripts (Japanese, Chinese, and Korean) but also in Roman, the directional codes are used as features.

Input patterns represented as feature point sequences can be matched with character models represented as feature point sequences, higher-level primitives (such as stroke), or hierarchical structures. Similarly, input patterns represented as line segments can be matched with character models represented as line segments, or higher-level structures.

2.3.4 Character recognition

The methods of online isolated character recognition can be roughly divided into two categories: structural methods and statistical methods. Structural methods are based on stroke analysis, especially for multi-stroke characters. The character models of structural methods can be further divided into stroke-order dependent models or stroke-order free ones, and stroke-number dependent models or stroke-number free ones. Statistical methods mainly utilize the holistic shape information, so it is easier to achieve stroke-order and stroke-number independence [21, 46].

In online character recognition, the input pattern is matched with the reference model of each (candidate) class and the class with the minimum matching distance is taken as the recognition result. We will discuss the character recognition techniques detailedly in Section 2.4.

2.3.5 Post-processing

So far, neither of above approaches, structural or statistical, has led to commercially acceptable results for the processing of isolated cursive script. Although the performance of online systems is generally higher than that of offline systems, the user requirements of almost no online recognition errors have limited the market applications to simple well-segmented handwritten characters. From this point of view, the specific post-processing for reading errors and rejecting them, is necessary and can give a commercial advantage. A feasible approach for character string recognition is widely employed by integrating segmentation and recognition based on over-segmentation.

Linguistic context knowledge is widely employed in post-processing. Generally, during the procedure of character recognition, the classification (or matching) module provides not only the candidate class index, but also the scores (probabilities, similarities, or distances) to a number of candidate classes. In addition, in the segmentation procedure, the handwritten character string is firstly segmented into many small basic units (over-segmentation) at possible positions because the handwritten characters (especially for cursive writing) are generally difficult to be segmented correctly. The linguistic context can provide valuable information for selecting the optimal class from this set of candidates and optimal character segmentation results based on those segmentation units [36, 37, 47]. For page-based handwritten text recognition dealing with sentences of long text, the integration of segmentation and recognition in contextual processing has particular importance.

2.4 Online character recognition strategy

2.4.1 Structural method and statistical method

As above mentioned, over the last decades, online multi-stroke character recognition has covered two distinct families of classification methods: structural methods and statistical classification methods.

Structural methods are based on stroke analysis. It represents the input pattern structurally by various levels of primitives [3, 48]. The character models of structural methods can be further divided into stroke-order dependent models and stroke-order free ones. Statistical methods mainly utilize the holistic shape information, and represent the input pattern statistically by feature vectors [49]. So it is easier to

achieve stroke-order independence.

The structural representation scheme has long been dominating the online Chinese character recognition technology, whereas the statistical scheme (or statistical-structural hybrid scheme) are receiving increasing attention in recent years. Chinese characters are known to be naturally structured patterns, and it is commonly admitted that considering structural information would be beneficial in the Chinese character recognition systems. In statistical representation, the input pattern is described by a feature vector, while the model database (also called parameter database) contains the classification parameters. Statistical methods are currently offering the highest performances since they are more suited to handle imprecision and variation of writing styles, including stroke connections, stroke-order variability and shape distortions. Besides, statistical models are easier to learn from samples in comparison with structural models that often require a tedious manual design task.

On the other hand, there is a statistical-structural hybrid scheme. The statistical-structural scheme is only used for describing the reference models. It takes the same structure as the traditional structural representation, yet the structure elements (primitives) and/or relationships are measured probabilistically. HMMs can be regarded as instances of the statistical-structural representation [50, 51, 52, 53].

Rule-based methods can be treated as a kind of structural methods. In rule-based methods, the prior knowledge of character structure and writing appears as heuristic rules or constraints [21, 54, 55, 56]. The constraints can be used to efficiently reduce the search space of structural matching. The rules represent the knowledge of basic strokes allowed for a character and the invariant geometric features of strokes.

2.4.2 Coarse and fine strategy

To speed up the recognition of the large category set, a fast coarse classification procedure is commonly used to first prune the search candidate classes to which the input pattern is expected to belong to. Then, the input pattern is classified into one of these candidate classes in the fine classification stage [21, 23]. This two-stage recognition strategy has been widely adopted by now. Character recognition of a large vocabulary, like Chinese and Japanese, commonly employs coarse and fine classification scheme for improving both the classification speed and accuracy.

Coarse classification can be accomplished by class set partitioning or dynamic candidate selection. In class set partitioning, the character classes are divided into disjoint or overlapping groups. The input pattern is first assigned to a group or multiple groups and then, in fine classification, the input pattern is compared in detail with the classes in the group(s). In dynamic candidate selection, a matching score (similarity) is computed between the input pattern and each class and a subset of classes with high scores is selected for detailed classification. The average number of candidates can be significantly reduced without loss of precision via selecting a variable number of candidates by confidence evaluation.

2.4.3 Radical-based method

Radical-based approaches decompose a large set of characters into a smaller set of radicals. Then, the problem is converted to radical extraction and optimization of combination of the radical sequences. In particular, for the Chinese character recognition, radical-based approaches are widely employed because modeling of radicals is

a simpler task compared to modeling the whole characters as the number of radicals is only in few hundreds. An overall recognition result of a character is obtained by combining the results of a few relevant radicals [4, 23, 57]. This approach need less examples for training but would require large amount of work to manually create the radical composition and order of characters.

When a reference model is constructed, the constituent radical prototypes are rescaled to the actual sizes and aspect ratios. During the matching between a reference model and an input pattern, the deformation vectors of a constituent radical are scaled back to square box and used to adjust the radical prototype. In a structured representation of reference models, the radical models are shared by different characters such that a character model is constructed dynamically by using the constituent radicals. This strategy can largely save the storage space of model database, considering the fact that hundreds of distinct radicals are shared by thousands of characters.

2.4.4 Classifier combination strategy

One effective approach to improve the performance of handwriting recognition is to combine multiple classifiers. However, it's not a simple problem to collect votes from different classifiers while adopting proper voting weight strategy [8, 58, 59, 60, 61]. In online multi-stroke character recognition, recently a combination of classification decisions from online and pseudo-online classifiers is performed to improve the recognition of online classifier only. The attained exciting results demonstrate that the pseudo-online representation by converting relevant online information to the offline representation of the character, is useful as the combination of classifiers perform

better than those based solely on pure online information.

2.5 Discussion on online recognition methods

Despite the higher accuracies achieved so far, some problems still remain unsolved. The rule-based approach proposed in the 1960s was abandoned for many years because of the difficulties encountered in formulating general and reliable rules as well as in automating the generation of these rules from a large database of characters. This approach has been rejuvenated recently with the incorporation of fuzzy rules and grammars that use statistical information on the frequency of occurrence of particular features. However, from a global point of view, for this approach to survive, robust and reliable rules will have to be defined.

In the past researches, HMM was introduced into Hiragana recognition with 46 classes, and educational Kanji character recognition with 881 classes, in which each character was independently modeled by an HMM. In this simple approach, the total size of models was proportional to the number of characters to recognize. Even for a small subset of Japanese characters, one study [62] required the model size of approximately 2MB and considered to be already too large for small-sized applications, such as PDAs. Moreover, the required model size would increase if we take into account a number of possible variations of stroke-orders of handwriting Kanji, since they are composed of up to 30 strokes for each and may be written in writer's own stroke-orders.

For Chinese characters, the main problem in online recognition is to overcome the stroke-order and stroke-number variations existing in multi-stroke characters. The

current systems can recognize regular script with high accuracy, whereas the recognition of cursive style still remains unsolved and requires more intensive research efforts because people naturally write this way. The cursive handwriting defined here is the faster handwriting approach of the standard Chinese characters. Due to many strokes in Chinese characters, during the writing of these characters, certain transformation of the characters happened: (i) Stroke-order variation. It is due to writer cannot remember all of the correct stroke-order of large number of Chinese character. (ii) Intra-character ligatures. This is due to no pen-lift between two consecutive strokes. It is normal in the Latin language as well. (iii) Strokes missing. This is very obvious for shorter strokes in between two strokes. The writer tends to omit these strokes to proceed to the following one. (iv) Shape deforms.

For stroke-order problem, stroke correspondence searching is a key technique for handwriting recognition. Stroke correspondence approach is theoretically the most reliable approach to estimate the actual stroke-order of input pattern explicitly, and can cope with even rare stroke-order variations of each class. Also, if we analyze the stroke-order variation, we can get important information to identify the writer, or give a hint for more beautiful handwriting, and so on. The technique to establish the stroke correspondence also can be applied into the various fields that need order analysis of feature sequences, such as DNA analysis, gesture recognition, etc.

In this thesis, we focus on the stroke-order issue of isolated multi-stroke characters and study stroke correspondence approach.

Chapter 3

Cube search

3.1 Introduction

As mentioned above, in this thesis, we focus on the stroke-order issue of isolated multi-stroke characters. Stroke correspondence approach is theoretically the most reliable approach to estimate the actual stroke-order of input pattern, even for the rare stroke-order variations of each class. In this chapter, we will detailedly introduce a typical method of stroke correspondence called cube search (CS).

The CS, proposed by Sakoe and Shin [11, 12, 63, 64, 65], is an effective stroke-order analysis algorithm for online character recognition. In CS, an N-dimensional cube graph stroke-order generation model is defined for N-stroke character to impose bijection property on the stroke correspondence. Then, the stroke correspondence search problem is formulated as an optimal path search problem on the cube graph. An efficient dynamic programming (DP) is used to search for the shortest path on the cube graph, with $O(N \cdot 2^{N-1})$ time complexity. In the following, we will first introduce the basic principle of stroke correspondence search, then give detailed

introduction about CS.

3.2 Basic principle of stroke correspondence search

We define the input character as a stroke sequence,

$$A = A_1 A_2 \dots A_k \dots A_N, \tag{3.1}$$

where the kth stroke A_k is the time sequence representation of local feature (e.g., moving direction and coordinate of pen-point).

$$A_k = a_{1k} a_{2k} \dots a_{ik} \dots a_{Ik}, I = I(k).$$

Similarly, we define the reference character as,

$$B = B_1 B_2 \dots B_l \dots B_N, \tag{3.2}$$

$$B_l = b_{1l} b_{2l} \dots b_{jl} \dots b_{Jl}, J = J(l).$$

We define stroke distance $\delta(k,l) = D(A_k, B_l)$ as the dissimilarity measure between the input stroke A_k and the reference stroke B_l, and calculate it by DP-matching [11, 74]. The character distance between A and B becomes $\sum_k \delta(k, l(k))$. The family of mappings $\{l = l(k)\}$ generates stroke-order variations. Therefore, it is a reasonable way to search for the optimal stroke-to-stroke correspondence by the following minimization problem.

$$D(A,B) = \min_{l(1),\dots,l(k),\dots,l(N)} \left[\sum_{k=1}^{N} \delta(k, l(k)) \right]. \tag{3.3}$$

A bijection constraint should be imposed on the mapping $l = l(k)$ to keep the property of actual stroke-order variation phenomena.

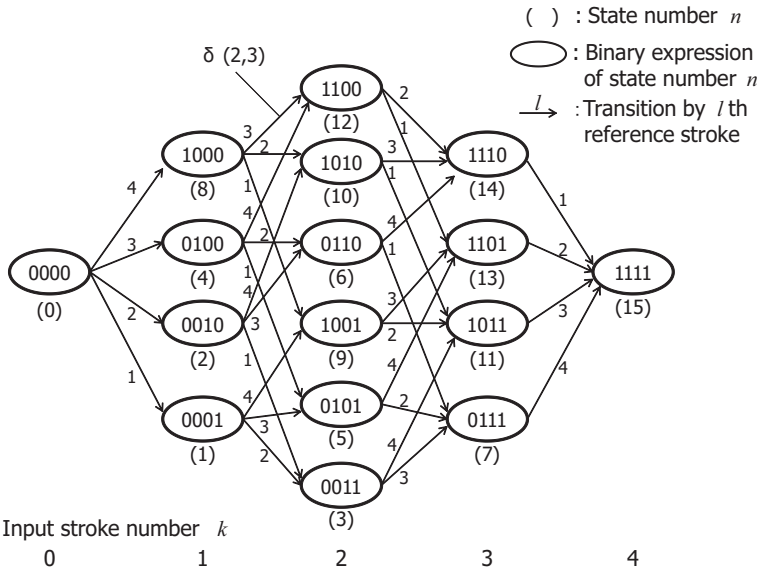


Figure 3.1: Cube search graph ($N = 4$).

3.3 Cube search (CS)

CS [11, 12] is a global optimization method for stroke correspondence search. Figure 3.1 is an example of the mechanism of CS for generating bijective stroke-order variations. It is an N-dimensional cube graph for N-stroke character. In CS, the N-dimensional cube graph is used for representing stroke correspondence; every path from the leftmost node to the rightmost node represents a stroke correspondence $l(1), \ldots, l(k), \ldots, l(N)$ and the graph can represent all $(N!)$ possible correspondences.

Specifically, as shown in Fig. 3.1, the graph is comprised of 2^N nodes and each of them is indexed by an N-bit binary number for identifying each node and controlling the selection of node-to-node transition (edge) as flags. Each bit position corresponds to the reference pattern stroke number l, and the bit value 1 means that this reference stroke has already been matched to some input stroke up to k. With

input stroke transition $k-1 \rightarrow k$, an edge is selected for causing a new reference stroke match. Let the lth reference stroke be matched to the kth input stroke, then the lth bit of node number is inverted as $0 \rightarrow 1$. For example, the node "1100" indicates that the first and the second input strokes corresponds to the third and the fourth reference strokes respectively, *or*, the fourth and the third reference strokes respectively, with the input stroke transition $1 \rightarrow 2$.

CS treats the stroke correspondence problem as a global path optimization problem under the condition that the two nodes linked by an edge should have only one different bit. For example, there is an edge from the node "0100" to "1100". This edge indicates that the second input stroke corresponds to the fourth reference stroke. Clearly, any path from the leftmost node "0000" to the rightmost node "1111" will satisfy this condition and represents a bijective correspondence. Since each edge indicates that a specific input stroke corresponds a specific reference stroke, the stroke distance $\delta(k, l)$ is attached to the edge. For example, the distance $\delta(2, 4)$ will be attached to the edge from the node "0100" to "1100".

Consequently, the optimal stroke correspondence problem is organized as a minimum distance path problem. The objective function is the summation of $\delta(k, l)$ along the correspondence $l = l(k)$, and the obtained minimum summation is used as character distance. Therefore, the problem is formulated as,

$$D(A, B) = \min_{l(1),\ldots,l(k),\ldots,l(N)} \left[\sum_{k=1}^{N} \delta(k, l(k)) \right]. \tag{3.4}$$

An efficient DP algorithm is used to search for the "globally" minimum distance path on the cube graph. In DP terminologies, the input stroke number k stands for stage, the node stands for state, edge cost or equivalently stroke distance stands

for local cost, and their sum total stands for objective function. The recurrence equation of DP is formulated as,

$$G(n) = \min_{m} \left[G(m) + \delta(k, l) \right], \tag{3.5}$$

where $G(n)$ is the minimum cumulative distance to the state n and there should be the relation that the binary number n has k "1"-bits and two binary numbers m and n are different only at the lth bit. The character distance $D(A, B)$ is obtained as $G(n)$ at the final state $n =$"1111". The time complexity of CS is $O(N \cdot 2^{N-1})$.

Since the computational complexity of the original CS is an exponential order of N, we need to introduce some technique to reduce the complexity for practice (especially, when deal with a multiple stroke character with large N). The *beam search* acceleration technique or pruning technique has been often used at a cost of global optimality. The CS with beam search (abbreviated as CSBS) provides suboptimal solution with far less computations than CS.

However, for characters with large N, CS is still time-consuming, even when the beam search is resorted to. Note that if we can consider that each multi-stroke character is comprised of *radicals*, we can expect that there is few stroke-order confusion in different radicals and then reduce computations drastically [20]. In Chapter 5, we start from CS, and derive an efficient radical-based method to solve the computational complexity problem of CS.

Chapter 4

Comparative study of stroke correspondence search methods for stroke-order variation

4.1 Introduction

Online multi-stroke character recognition is gaining renewed interest because of the new development of new pen devices and their applications. An important distinction between online and offline handwriting recognition (or OCR) is the fact that spatial and temporal information are available in the former, while only spatial information is available in the latter. For the online case, character patterns are expressed as the trajectory of pen tip motion during writing. For the offline case, the patterns are expressed as scanned images and recognized by conventional OCR techniques.

A practical online multi-stroke character recognition system should be *stroke-*

order free to cope with characters with stroke-order variation. In fact, for multiple stroke characters like Chinese characters or Japanese Kanji characters, the number of strokes of a single character often exceeds 20 and their stroke-order varies in many ways. Figure 4.1 shows a simple and typical example of stroke-order variation of a Japanese Kanji character, "right".

Although there are several researches for solving the stroke-order free recognition problem, the problem has not been solved yet. Moreover, since no extensive comparative study has been made so far, none knows the relative characteristics and performance of the past approaches. In other words, there has been no research that helps to choose the most suitable stroke-order free recognition approach for individual applications.

The main contributions of this chapter are twofold.

- A review is made to clarify the relative superiority of stroke-order free recognition approaches from the viewpoints of recognition accuracy and efficiency.

- Especially for stroke correspondence approach, theoretically the most reliable approach among several approaches, we conduct a quantitative and qualitative evaluation of five representative methods through an experiment, by using 17,983 samples of online Japanese Kanji character.

To cope with stroke-order variation, several approaches, such as multiple prototype approach, offline feature approach, and stroke correspondence approach, have been proposed. Our investigation shows that, in the past decade, the conventional multiple prototype approach was frequently reported, especially for Hidden Markov Models (HMMs) based methods. This approach is simple and prepares several pro-

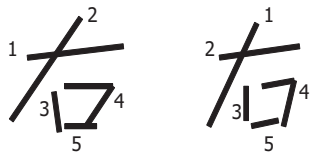


Figure 4.1: Example of stroke-order variation.

totypes with typical stroke-order variations for each class. It might be attributed to the emerging application of HMMs in online multi-stroke character recognition during recent years, and it is short of effective strategy to cope with stroke-order variation. The offline feature approach was also intensively studied in the past decade and gave exciting recognition result. This approach converts the motion trajectory into a 2D image to avoid stroke-order variation problem. The stroke correspondence approach, which was intensively studied in 1980s and 1990s, is the traditional strategy to cope with stroke-order variation. This approach first optimizes the stroke correspondence between an input pattern and a reference pattern and then evaluates the dissimilarity between those patterns.

In this chapter, we focus on the stroke correspondence approach. This is because the stroke correspondence approach has the following promising characteristics:

- It is theoretically the most reliable approach among several approaches.
- It is possible to estimate the actual stroke-order of the input pattern explicitly and thus useful to analyze stroke-order variation. It can provide important information to identify the writer, or give a hint for more beautiful handwriting, and so on.
- The stroke correspondence approach can cope with even rare stroke-order vari-

ations of each class.

- The technique to establish the stroke correspondence is still very important because it can be applied into the various fields that need order analysis of feature sequences, such as DNA analysis, gesture recognition, etc.

Since the stroke correspondence approach is formulated as an optimization problem, its characteristics depend on its optimization criterion and optimization strategy. Several research groups have proposed promising stroke correspondence-based methods for stroke-order free online multi-stroke character recognition, based on different formulations of the optimization problem.

In this chapter, a comparative study is made to clarify the relative superiority of representative stroke correspondence-based methods from the viewpoints of recognition accuracy and efficiency. Specifically, we pick up the following five methods:

- *The cube search method* (CS) by Sakoe and his colleagues [11, 12].
- *The bipartite weighted matching method* (BWM) by Hsieh et al. [13].
- *The individual correspondence decision method* (ICD) by Wakahara and his colleagues [14, 15].
- *The stable marriage method* (SM) by Yokota et al. [16].
- *The deviation-expansion model method* (DE) by Lin et al. [17].

Note that our comparative experiments adhere to "stroke-number fixed" condition, where no stroke concatenation occurs. This is because we try to concentrate on the influence of stroke-order variations on recognition performance of those methods.

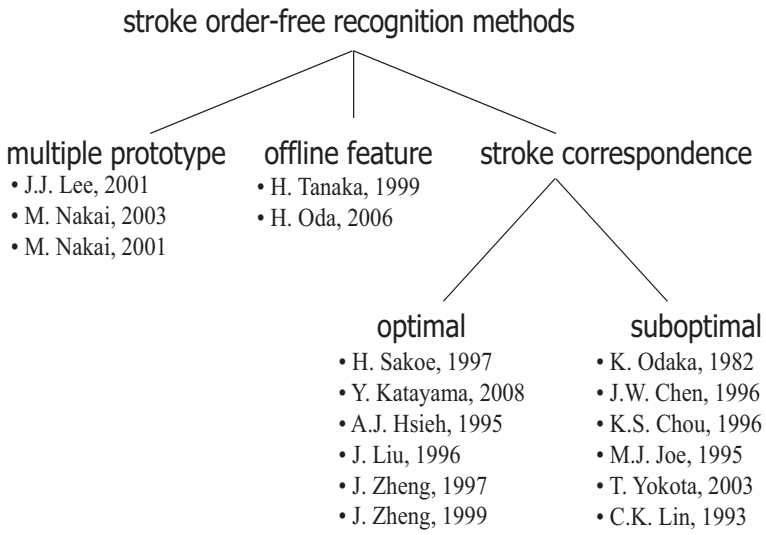


Figure 4.2: Classification of methods of dealing with stroke-order variation.

In the following, we begin with a brief review on the methods of online multi-stroke character recognition in Section 4.2. In Section 4.3, the basic principle of stroke correspondence search and some details of the above five methods are summarized. Our test results and analysis will be presented in Section 4.4. Section 4.5 gives an extensive discussion on the five methods. Section 4.6 addresses the application of stroke correspondence search in forensics, and Section 4.7 provides some concluding remarks.

4.2 Review of stroke-order free methods

In online multi-stroke character recognition, the problem of stroke-order variation has been studied for many years. Various methods have been published to solve the problem [21]. As shown in Fig. 4.2, we roughly divide the methods of dealing with stroke-order variations into three classes, i.e., multiple prototype, offline feature, and stroke correspondence. Furthermore, we divide the stroke correspondence approach

into two classes, i.e., optimal approach and suboptimal approach.

4.2.1 Multiple prototype approach

The multiple prototype approach is the simplest approach to cope with stroke-order variations. In this approach, several prototypes with different stroke-order variations are prepared for every class. In general, the patterns with typical stroke-order variations are collected as the prototypes. When an input pattern is recognized, the most similar prototype is simply selected and then the input pattern is recognized as the class of the prototype.

Lee et al. [66] have proposed a method based on the multiple prototype approach using HMMs. First, each online character pattern is represented as a sequence of straight-line segments (i.e., stroke), and modeled by HMM. Then, typical stroke-orders of each class are selected by clustering the training pattern of the class. For each of the selected stroke-order, an individual HMM is prepared. Consequently, the multiple HMMs are prepared for each class. Further, for efficient representation of variations, those HMMs are combined to form a single HMM architecture, called a multiple parallel-path HMM [66].

In [52], Nakai et al. proposed a multiple prototype approach with a hierarchical structured dictionary. In this dictionary, the common subpattern stroke-orders are for reducing higher time complexity drastically. The dictionary is represented as a network and constructed hierarchically by using shared subnets which define stroke-order rules of Kanji subpatterns. Based on the dictionary network, thousands of stroke-order variations of Kanji patterns can be produced using a small number of subpattern stroke-order rules. They successfully applied it to their previous study

of substroke-based HMM [53], and showed that the recognition speed was fast.

The merit of the multiple prototype approach is its simplicity. However, it is clear that it is difficult to define typical stroke-order variations for general use. In addition, it can not cope with rare stroke-order variations. Furthermore, its time complexity is large when allowing huge variations.

4.2.2 Offline feature approach

The offline feature approach employs "inking" process that converts an online character pattern into a 2D image. Then we can apply OCR techniques to recognize the online character patterns, need not deal with stroke-order variations explicitly.

Tanaka et al. [67] converted an online character pattern into a bitmap image, and then, an offline classifier was applied to recognize it. In order to improve recognition accuracy, a classifier combination strategy is adopted, where this offline classifier and another online classifier are combined. The offline classifier works as a coarse classifier to provide candidates for the online classifier, so that the time complexity of the online classifier can be decreased.

Oda et al. [68] also proposed a combined classifier, composed of an online classifier and an offline classifier. For the latter classifier, the online character patterns are converted to bitmap images. In [68], in order to decrease time complexity and memory size, they proposed several approaches to reduce the dictionary sizes of the online and offline classifiers significantly, especially for the offline dictionary. Specifically, for the online classifier, they employed a structured character pattern representation dictionary based on the common subpatterns of Kanji patterns, to reduce the total memory size. For the offline classifier, the modified quadratic

discriminant function (MQDF2) is used, and its dictionary size is drastically reduced by reducing parameters for MQDF2.

4.2.3 Stroke correspondence approach

The stroke correspondence approach determines the optimal or suboptimal stroke correspondence between an input pattern and a reference pattern. If the actual stroke-order of the input pattern can be estimated, it is possible to eliminate the effect of stroke-order variations, even when rare stroke-order variation occurs. In addition, the stroke-order information estimated by the stroke correspondence approach can be utilized to identify the writer, to give a hint for more beautiful handwriting, and so on.

In the following, we will briefly introduce some methods based on the stroke correspondence approach. As shown in Fig. 4.2, those methods are classified into two classes, that is, methods for optimal stroke correspondence and the methods for suboptimal stroke correspondence. Generally speaking, the former gives more accurate correspondence with larger computations (This fact will be first proven experimentally by this chapter). Note that among the following methods, CS, BWM, ICD, SM, and DE, will be detailed in Section 4.3, then compared experimentally and discussed in Sections 4.4 and 4.5.

In this chapter, the classification between optimal and suboptimal is done whether a method can give the globally optimal solution of the stroke correspondence problem formulated later. Thus, some method becomes suboptimal if it gives an optimal solution in, e.g., a constrained problem.

Methods for optimal stroke correspondence

As introduced in Chapter 3, in CS [11, 12] Sakoe and Shin formulate the stroke correspondence search problem as a shortest path search problem on an N-dimensional cube graph. This cube graph is introduced to impose the bijection property on the stroke correspondence between an N-stroke input pattern and an N-stroke reference patterns. An efficient dynamic programming (DP) algorithm is used to search for the shortest path on the cube graph. Note that CS can be extended to be a stochastic stroke correspondence model [50]. In this case, the cube graph becomes an HMM.

Hsieh et al. [13] proposed a straightforward stroke correspondence method by defining the stroke matching problem as BWM, where the stroke distances are served as the weights of edges of an $N \times N$ bipartite graph. The BWM problem is to find the minimum weight perfect matching of the bipartite graph. Hsieh et al. formulated this minimization problem as integer programming (IP) problem, and applied the Hungarian algorithm to solve this IP problem.

In [69], Liu et al. represented both of the reference pattern and the input pattern as complete attributed relational graphs (ARG). In a complete ARG, its nodes and arcs describe line segments and relations between any two segments, respectively. They defined a measure for the optimal correspondence of nodes and arcs between two ARGs. The correspondence searching is formulated as a heuristic search problem in a state space tree. An A* algorithm is used to perform the heuristic search.

In [70], to overcome the shortcoming of being too strict description of primitive (strokes or line segments) relationship in popular ARG, Zheng et al. proposed so-

called a fuzzy ARG (FARG). In FARG, the attributes of nodes and/or arcs are represented with fuzzy sets. The FARG is employed to define reference patterns for Chinese characters. An elastic matching algorithm is used to search the optimal stroke correspondence between an input pattern and the FARG.

In [71], Zheng et al. proposed a path-controlled HMM (PCHMM). It is an improved version of popular HMM-based methods and needs not prepare multiple HMMs for the stroke-order variations of each class. In PCHMM, only one HMM is modeled for each class, corresponding to the optimal state transition path on the state sequence space. A path controlling function is defined for directly controlling the optimal state transition path. The path controlling function can place some constraints on state sequence space, according to the optimal line segment correspondence between input and reference patterns, searched by using an A* algorithm.

Methods for suboptimal stroke correspondence

ICD [14] is the simplest method to establish a suboptimal stroke correspondence. It cannot guarantee one-to-one correspondence because it relies on a simple greedy algorithm. Specifically, for each of input strokes, one reference stroke with minimum distance (called stroke distance) is selected. In [55], [72], and [73], the greedy algorithm is also employed to search the stroke correspondence under different constraints. In [55], under the constraint of predefined rules of reference strokes, such as possible predefined stroke types and invariant geometric features of strokes. In [72], under the constraint of the minimum stroke distance that satisfies some conditions on feature point coordinates and slopes of strokes. In [73], under the constraint of the minimum stroke distance, for which input stroke and reference stroke to be matched

must have the same number of segments. Clearly, the above methods cannot avoid one-to-many correspondence.

SM [16], which will be detailed later, is different from other methods because it uses the rank of the stroke distance instead of the value of the stroke distance for the criterion of correspondence optimization. The rank can be considered as a rough approximation of the stroke distance as discussed later. SM has a very efficient algorithm for determining the stroke correspondence under this rank criterion.

DE, proposed by Lin et al. [17] is a model to represent the reference pattern. DE contains prior knowledge of possible writing deviations. Specifically, a tree graph is constructed with limited branches for representing stroke-order variations, and then a DP algorithm is used to search for the optimal path on the tree. Because of the limitation, DE cannot deal with all possible stroke-orders and thus DE is treated as a suboptimal method.

4.3 Details of five representative methods based on stroke correspondence approach

In this section, we will formulate the optimal stroke correspondence problem. Then, we will pick up and detail five representative methods to solve the problem, that are, two (globally) optimal methods, CS and BWM, and three suboptimal methods, ICD, SM, and DE.

4.3.1 General problem of determining optimal stroke correspondence

Let A denote an input pattern with N strokes,

$$A = A_1 A_2 \ldots A_k \ldots A_N, \tag{4.1}$$

where A_k is the kth stroke and represented as a sequence of feature vectors. For example, A_k is a sequence of three-dimensional vectors, each of the vector is comprised of the local direction and $x - y$ coordinates. Similarly, let B denote the reference pattern,

$$B = B_1 B_2 \ldots B_l \ldots B_N. \tag{4.2}$$

Let $\delta(k, l)$ denote a *stroke distance* between the input stroke A_k and the reference stroke B_l. Note that the dimensionalities of A_k and B_l are often different due to the difference of their stroke lengths. Thus, we generally cannot calculate the simple Euclidean distance between them. Instead, DP-matching distance [11, 74] has been often utilized for calculating a distance between a pair of strokes with different lengths.

Consider a mapping $l = l(k)$ for representing the stroke correspondence between A_k and B_l. Under the mapping $l(k)$, the stroke A_k corresponds to $B_{l(k)}$. Thus, the *character distance* between A and B becomes $\sum_k \delta(k, l(k))$. The mapping $l(k)$ should be bijective (one-to-one) from $\{A_k\}$ onto $\{B_l\}$.

The *optimal stroke-order* of A can be obtained as $l(1), \ldots, l(k), \ldots, l(N)$ to minimize the criterion $\sum_k \delta(k, l(k))$. It is very important to note that stroke correspondence methods try to minimize the criterion $\sum_k \delta(k, l(k))$ in their own ways. Some

of them are very efficient suboptimal methods and the others are global optimization methods. Hereafter, we will compare five stroke correspondence-based methods.

4.3.2 Cube search (CS)

As introduced in Chapter 3, CS [11, 12] is a global optimization method for stroke correspondence search. In CS, an N-dimensional cube graph is used for representing stroke correspondence. CS treats the stroke correspondence problem as a global path optimization problem on the cube graph. An efficient DP algorithm is used to search for the "globally" minimum distance path on the cube graph.

The time complexity of CS is $O(N \cdot 2^{N-1})$. Since the computational complexity of the original CS is an exponential order of N and time-consuming with large N, the *beam search* acceleration technique or pruning technique has been often used at a cost of global optimality. The CS with beam search (CSBS) provides suboptimal solution with far less computations than CS.

4.3.3 Bipartite weighted matching (BWM)

BWM [13] is also an optimal method for stroke correspondence search. In [13], Hsieh et al. modeled the stroke correspondence problem as a bipartite weighted graph matching problem. Figure 4.3 shows an example of bipartite weighted graph for determining the optimal stroke correspondence. The kth left vertex stands for the kth input stroke A_k, whereas the lth right vertex stands for the lth reference stroke B_l. Let the weight of edge (A_k, B_l) be the stroke distance $\delta(k, l)$. A matching in a graph is a set of edges, no two of which share a vertex. If every vertex $A_k \in A$ is incident to an edge in the set, then the matching is perfect.

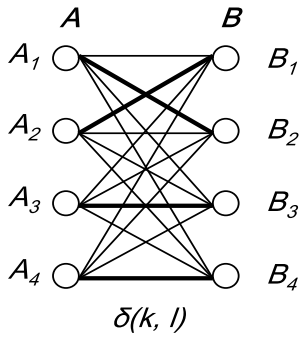


Figure 4.3: A weighted bipartite graph ($N = 4$). A perfect matching is marked by thick lines.

BWM problem is to find the globally optimal perfect matching such that the sum of the weights (i.e. stroke distances) of matching edges is minimum. Under the assumption that the stroke distances are given, Hsieh et al. [13] formulated this minimization problem by the following IP:

$$D(A,B) = \min_{x_{11},\ldots,x_{kl},\ldots,x_{NN}} \left[\sum_{k=1}^{N} \sum_{l=1}^{N} \delta(k,l) x_{kl} \right]. \tag{4.3}$$

where x_{kl} is a binary value (0 or 1) and satisfy $\sum_{k=1}^{N} x_{kl} = 1$ for all l and $\sum_{l=1}^{N} x_{kl} = 1$ for all k. Note that $x_{kl} = 1$ when A_k is matched with B_l.

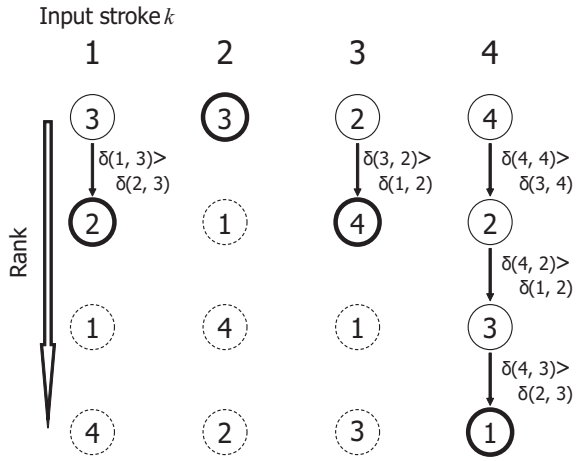
In [13], this IP problem is solved by the Hungarian method [75, 76, 77], the well-known primal-dual algorithm. By applying the Hungarian method, the algorithm solves the N strokes correspondence problem in $O(N^3)$ computations.

4.3.4 Individual correspondence decision (ICD)

ICD [14, 15] is a suboptimal method for stroke correspondence search. In ICD, the stroke B_l corresponding to the stroke A_k is determined independently. First, an $N \times N$ stroke distance matrix whose element equals $\delta(k,l)$ is defined. Figure 4.4

Input stroke k	Reference stroke l			
	①	②	③	④
1	**62**	14★	13	87
2	61	139	**10**★	86
3	87	**25**	97	36★
4	158★	64	97	**42**

(a) The stroke distance table of the example. The stroke correspondences by ICD and SM are indicated by underlines and ⋆, respectively. The optimal one-to-one correspondence is indicated by boldface.



(b) Find the stroke correspondence by SM ($N = 4$).

Figure 4.4: An example of finding the stroke correspondence.

(a) shows an example of the matrix. Then, the minimum value in each row k of the matrix is searched for. That is, for the kth input stroke, the lth reference stroke with the minimum stroke distance from k is determined as the corresponding stroke (e.g., in Fig. 4.4 (a), ③ = $l(1)$). Consequently, the minimized character distance $D(A, B)$ can be described as

$$D(A, B) = \sum_{k=1}^{N} \min_{l} \delta(k, l(k)). \tag{4.4}$$

There is no guarantee that the result of stroke correspondence becomes one-to-one as shown in the example of Fig. 4.4 (a). The time complexity of ICD is $O(N^2)$.

4.3.5 Stable marriage (SM)

According to the formulation of Section 4.3.1, SM [16] is a suboptimal searching method. The principle of SM is often explained by the situation that N men and N women have expressed mutual preferences (each man must say how he feels about each of the N women and vice versa). The SM problem is to find a set of N stable marriages according to the preferences [78]. For stable marriages, we should avoid unstable marriage, e.g., the situation that in two married couples (M_1, F_1) and (M_2, F_2), M_1 prefers F_2 to F_1 and F_2 prefers M_1 to M_2.

Yokota et al. [16] applied SM to the stroke correspondence problem, where input strokes $\{A_k\}$ and reference strokes $\{B_l\}$ stand for men and women, respectively. A natural way to express the preferences is the stroke distance $\delta(k, l)$. The process of determining SM for Fig. 4.4 (a) is illustrated in Fig. 4.4 (b). For example, when $k = 1$, the reference stroke ③ is first selected as the candidate stroke and matched with the input stroke 1. Then, when $k = 2$, since $\delta(2, 3) < \delta(1, 3)$, the input stroke 2 will match with the reference stroke ③ and the input stroke 1 has to select the lower order of reference stroke ②.

The most important point is that SM does not use the value $\delta(k, l)$ directly but uses the rank of the preference for establish the correspondence. This fact can be confirmed by the fact that if we transform the distance values in Fig. 4.4 (a) into the rank (the resulting table is called "preference list"), the resulting correspondence is the same. Because of this property, SM is treated as a suboptimal method in this

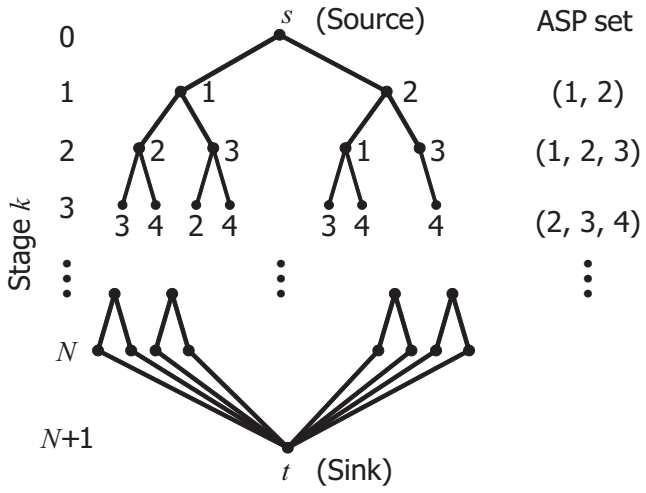


Figure 4.5: Tree for DE.

chapter. When employ $l(k)$ in the above method, the character distance is given as $\sum_k \delta(k, l(k))$. The time complexity of SM is $O(N^2)$.

4.3.6 Deviation-expansion model (DE)

In this chapter, DE [17] is also viewed as a suboptimal method because it practically can deal with only a limited number of stroke-orders as follows. Lin et al. [17] proposed DE as a general concept of stroke correspondence model. Unfortunately, in [17], they gave no solution algorithm except for a very limited case, where stroke-order deviation is limited to the two directly adjacent orders. The correspondence algorithm for this limited case is briefly given below.

Figure 4.5 shows an example of the tree for DE. In DE, the corresponding stroke for k is limited as $l \in \{k-1, k, k+1\}$. This set is called adjacency stroke position (ASP). The branch of the tree is constrained by ASP. A stroke distance $\delta(k, l)$ is assigned at each node of the tree. In [17], the minimum distance path on the tree is

searched for by DP[1]. Since the tree branches exponentially, the time complexity of DE is $O(2^N)$.

4.3.7 Comparison of computational complexity

The time complexities of the above five methods are summarized in Table 4.1. It is clear that ICD, SM, and BWM require polynomial computations and have the relative superiority over the other methods. Note that CSBS has a parameter (the threshold for beam search) and its computational complexity depends on the parameter. Note that their actual computation times will be compared experimentally in the next section.

4.4 Comparative experiment and evaluation

4.4.1 Experimental setup

In order to compare the performance of the above methods, we conducted a comparative recognition experiment on a workstation (with a 1.7GHz CPU). In Section 4.4.2, we will first observe the accuracy of the stroke correspondence by the methods. We also observe the recognition accuracy by the methods.

Since we want to analyze the performances of the methods on two sets of correct and incorrect stroke-order respectively, a dataset with known stroke-orders of characters was employed, rather than a common database like the well-known Kuchibue database [79] or the recent CASIA database [80]. A total of 17,983 Kanji char-

[1]Intuitively speaking, this is a "reversed" tree search where the searching process starts from leaves and ends in the root.

Table 4.1: Performance summary of the methods of stroke correspondence.

Methods	**Solution**	**Time complexity**	**Recognition rate(%)**	**Computation time per character (ms)**	**Rates of character samples with perfect stroke correspondences(%)**	
					Correct stroke -order samples	**Incorrect stroke -order samples**
DM	—	None	93.38	< 0.1	100.00	0.00
CS [11]	Optimal	$O(N \cdot 2^{N-1})$	99.17	553.8	98.34	95.75
CSBS [11]	Suboptimal	Depends on pruning threshold	99.15	43.5	98.32	95.53
BWM [13]	Optimal	$O(N^3)$	99.17	6.4	98.34	95.75
ICD [14]	Suboptimal	$O(N^2)$	96.37	0.4	68.55	54.02
SM [16]	Suboptimal	$O(N^2)$	98.54	2.2	83.67	73.66
DE [17]	Suboptimal	$O(2^N)$	96.59	1212.9	98.62	47.79

∗Note that the direct matching method (DM) that establishes stroke correspondence directly by the original stroke-order, is also listed here for emphasizing the importance of stroke correspondence optimization.

Figure 4.6: Samples of our dataset.

acter patterns (882 categories, 1-20 strokes) written by 30 persons were used in the dataset. The stroke-order of each pattern was confirmed detailedly, and 14,786 patterns were written with correct stroke-order, 3,197 patterns were written with incorrect stroke-order. Figure 4.6 shows some Kanji samples of this dataset. Again, we adhered to exclude variations in stroke-numbers for understanding only the effect of stroke-order variations in stroke correspondence. Accordingly, each character in the dataset was written with its correct (standard) stroke-number.

Each stroke (A_k or B_l) was represented as a sequence of the $x-y$ coordinate feature and the (quantized) directional feature. The stroke distance $\delta(k,l)$ were calculated by DP-matching and all of the methods used the same $\delta(k,l)$ for optimizing the stroke correspondence.

4.4.2 Experimental results

The accuracies of the stroke correspondence by the five methods are shown in Table 4.1. Specifically, this is the rate of characters with perfect stroke correspondences. The test patterns used were written in not only correct stroke-order but also incorrect stroke-order.

To emphasize the importance of stroke correspondence optimization, the results of the direct matching method (DM), are also listed in Table 4.1. DM establishes stroke correspondence directly by the original stroke-order; that is, $k = l(k)$. Accordingly, DM cannot deal with stroke-order variations.

Table 4.1 also shows the experimental results of recognition rates and computation times for each of the five methods. Note that 39ms for calculating local distances $\{\delta(k, l)\}$ were excluded from the computation times in the table because this 39ms were common for all of the methods.

To analysis the influence of the number of input strokes on recognition rate, the error rates of character recognition depending on the number of input strokes N were evaluated for each of the methods and shown in Table 4.2. Table 4.2 also shows the number of character categories on N in our dataset.

4.4.3 Performance analysis

Table 4.1 shows that CS (including CSBS) and BWM could provide accurate stroke correspondence for both correct and incorrect stroke-order input patterns. In particular, for input patterns with incorrect stroke-order, CS (including CSBS) and BWM show remarkable superiority than the other methods.

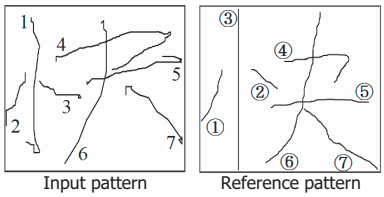
Table 4.2: The error rates of character recognition depending on N.

N	♯ of categories	Character error rate (%)						
		DM	**CS**	**CSBS**	**BWM**	**ICD**	**SM**	**DE**
1	1	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	10	8.39	2.80	2.80	2.80	2.80	2.80	2.80
3	19	9.79	2.29	2.29	2.29	3.54	2.29	3.33
4	45	7.51	1.34	1.34	1.34	3.75	1.43	3.04
5	65	10.64	1.47	1.47	1.47	3.31	1.47	4.59
6	65	8.93	0.98	1.04	0.98	3.65	1.50	5.28
7	79	8.27	1.18	1.12	1.18	5.01	1.91	4.16
8	94	9.61	0.76	0.76	0.76	4.55	1.42	4.92
9	86	4.40	0.77	0.77	0.77	3.19	1.43	2.97
10	80	6.11	0.46	0.53	0.46	4.25	1.53	3.19
11	77	4.80	0.47	0.47	0.47	2.47	1.47	3.80
12	86	3.72	0.31	0.25	0.31	4.72	1.51	2.52
13	51	2.99	0.43	0.43	0.43	2.56	1.28	1.18
14	46	3.55	0.39	0.66	0.39	2.37	0.79	0.92
15	27	2.83	0.00	0.00	0.00	2.12	0.24	0.94
16	16	0.42	0.00	0.00	0.00	0.84	0.00	0.00
17	4	0.00	0.00	0.00	0.00	0.00	0.00	0.00
18	11	6.38	0.71	0.71	0.71	3.55	2.13	2.84
19	3	0.00	0.00	0.00	0.00	0.00	0.00	0.00
20	3	2.13	0.00	0.00	0.00	0.00	0.00	0.00

Figure 4.7 (b) gives examples of stroke correspondences between an input pattern "快" (cheerful) and a reference pattern "快" shown in Fig. 4.7 (a), by CS, BWM, ICD, SM, and DE. Figure 4.7 (c) shows the stroke distance table between the two patterns. In this case, as shown in Fig. 4.7 (b), CS and BWM gave perfect stroke correspondences. The stroke correspondence of ICD was not one-to-one, and SM and DE also gave incorrect stroke correspondences. Based on Fig. 4.7 (c), SM gave the incorrect correspondences of input strokes 4, 5, and 7, because SM only determined their correspondences locally by iteratively comparing the preferences in their ranks, and could not avoid correction of the errors by considering global optimality. DE could not give the correct correspondence because the stroke-order variation in the input pattern had exceeded the expected variation range (± 1).

Table 4.1 shows that all stroke correspondence search methods can give higher recognition accuracy than DM and proves the importance of stroke correspondence search. Table 4.1 also shows that CS (including CSBS) and BWM can obtain higher recognition accuracy, and ICD, SM, and BWM can obtain higher recognition speed. On the other hand, although CSBS is slower than ICD, SM, and BWM, its recognition speed is also fast and practical. DE is very slower than the other methods as expected from its exponential computations, $O(2^N)$.

In Table 4.2, CS and BWM show the same lowest error rates of character recognition, and the poor correspondence accuracy of SM, DE, and ICD caused many errors. It is observed that, for all of the methods, there exists a tendency that the error reduces as N increases (the sharper variation at $N = 18$ is just due to the small number of input patterns at there). It seems reasonable to conclude that the information conveyed by a character pattern increases as the stroke-number in-



(a) An input pattern "快" (cheerful), a reference pattern "快", and their stroke-orders.

		Correspondence of reference stroke				
		CS	BWM	ICD	SM	DE
Input stroke	1	③	③	③	③	①
	2	①	①	①	①	③
	3	②	②	②	②	②
	4	④	④	④	⑦	④
	5	⑤	⑤	④	④	⑤
	6	⑥	⑥	⑥	⑥	⑥
	7	⑦	⑦	⑤	⑤	⑦
Character distance		92	92	75	150	131

(b) Stroke correspondences and character distances given by each method for the character pair of Fig. 4.7 (a).

The correct correspondence is indicated by the underline.

Input stroke	Reference stroke						
	①	②	③	④	⑤	⑥	⑦
1	15	44	10	95	96	38	90
2	3	21	37	70	62	72	61
3	13	4	65	29	18	70	42
4	98	63	121	23	59	79	98
5	74	35	128	10	23	90	62
6	37	43	20	68	66	7	63
7	49	37	75	27	18	53	22

(c) Stroke distance table between the character pair of Fig. 4.7 (a).

Figure 4.7: Stroke correspondence examples by the five methods.

creases, and sufficient information is available to discriminate characters. With CS and BWM, error rates monotonically reduce and keep the lowest error rate at each N. Especially at large values of N, CS and BWM show remarkable superiority. It means that, for the complex input Kanji with large stroke-number, CS and BWM can utilize the increased information more efficiently than ICD, SM, and DE.

For the above methods, we can summarize the experimental results as follows.

- Since CS and BWM can obtain the globally optimal stroke correspondence (from all possible correspondences), they can give the same optimal stroke correspondence and obtain the same highest recognition accuracy, especially for difficult input patterns, that is, patterns with incorrect stroke-order and/or large stroke-number.

- There is only a slight difference in perfect stroke correspondence rate, as well as recognition rate, between CSBS and CS. In contrast, there is a drastic reduction in computation time by CSBS. This indicates that, although CSBS is a suboptimal method by the introduction of beam search (pruning), it is possible to find the optimal solution in many cases.

- ICD cannot give accurate stroke correspondence as shown in Table 4.1. This also results in the poor recognition rate 96.37%. Those facts indicate that the stroke correspondence problem is difficult to solve by a simple greedy optimization method.

- SM could not provide accurate stroke correspondence. This is because only SM employs a rank criterion instead of the distance criterion. The rank criterion

sometimes underestimates large difference and overestimates small difference, and thus is insufficient to evaluate the stroke correspondence.

- DE could not provide correct stroke correspondence results for incorrect stroke-order input. This is because that DE cannot deal with stroke-order variation beyond the range permit of model assumption. Unfortunately, it is practically impossible enlarge the range because DE graph becomes huge and then its computation becomes intractable.

From above results, clearly, for the performance of recognition accuracy, relative superiority of CS and BWM over ICD, SM, and DE are established.

4.5 Discussions

Since the above mentioned methods can give different performances in different conditions of stroke-order and stroke-number of input patterns, they have their own appropriate characters. For the optimal methods of CS and BWM, it is reasonable to employ them to recognize the characters with large stroke-number, such as Chinese character or Japanese Kanji (or employ them in the other pattern recognition fields, eg., the recognition of gestures with complex and variable orders of bodily motion). For the suboptimal methods of ICD and SM, it is reasonable to employ them to recognize the characters (or other kinds of patterns) with small stroke-numbers, such as Roman and Japanese Kana. However, application of ICD and SM to the Indian characters like Bangla [32] will not be appropriate even though they have small stroke-number. This is because the shapes of Bangla character strokes are very similar to each other, globally optimal stroke correspondence will be necessary

Table 4.3: The application condition for each method.

Methods	**Application condition**			
	♯ of strokes	**Alphabet size**	**Speed**	**Ex. of character script**
CS (CSBS)	Large / Small	Large	Middle	Chinese, Kanji, Bangla
BWM	Large / Small	Large	Fast	Chinese, Kanji, Bangla
ICD	Small	Small	Fast	Roman, Kana
SM	Small	Small	Fast	Roman, Kana
DE	—	—	Slow	—

to keep the higher recognition accuracy than ICD and SM. CS and BWM are also advised for the characters with large alphabet, such as Indian characters, Kanji characters or subpatterns of Kanji. Since the recognition accuracy always decreases as the alphabet increases, the optimal methods of CS and BWM are needed to keep the higher recognition accuracy, for the large alphabet characters. On the other hand, if we consider the recognition speed (or computational complexity), the ICD, SM, and BWM are advised, especially for those simple recognition systems (e.g., portable data terminal). Table 4.3 shows our recommendation of the application conditions for each of the methods.

The above mentioned methods of stroke correspondence approach also can combine with the methods of multiple prototype and offline feature approach to decrease the total time complexity and improve the recognition accuracy. CS and BWM give stroke correspondence with higher accuracy, between the input pattern and the reference pattern of a class. Their stroke correspondence results can help the methods

of multiple prototype approach to pick up some more possible prototypes so as to decrease the complexity, rather than all of the prototypes of the class, especially for the class with large stroke-number. On the other hand, for methods of offline feature approach, it seems reasonable to employ the information of one-to-one stroke correspondence, given by above methods like CS, BWM, and SM, to evaluate the image character distances and improve the recognition accuracy.

4.6 Toward forensics by stroke-order variation

In this Section, we consider personal identification using stroke-order variations of online multi-stroke character patterns written on, e.g., electric tablets. To extract the stroke-order variation of an input character pattern, it is necessary to establish the accurate stroke correspondence between the input pattern and the reference pattern of the same category.

We assume that the writing order of a multiple-stroke character pattern (i.e., stroke-order) represents writer's individuality and thus discuss how we can determine the stroke-order of an unknown input character pattern. Our target is "online" character patterns acquired from electric tablets or other electric pen devices and represented as the motion trajectory of the pen tip. Different from "image" character patterns acquired from scanners or cameras, online character patterns can convey the stroke-order information and thus we can examine the effectiveness of the stroke-order on forensic applications by them. Figure 4.8 shows three different stroke-orders of a four-stroke Chinese character "king", acquired by an electric tablet. The numerals in this figure indicate the order of the four strokes.

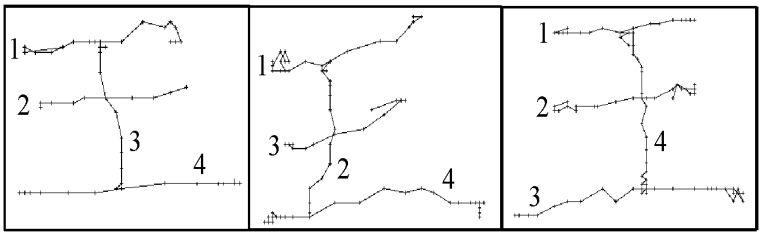


Figure 4.8: Three typical stroke-orders of Chinese character "king".

The author's preliminary observation on our dataset of online character patterns (mentioned in Section 4.4.1) indicates that stroke-orders are *not* random. For example, in the case of the character "king" of Fig. 4.8, only those three stroke-orders were found among 4! = 24 possible orders in a dataset by 30 different writers. Another preliminary observation indicates that each writer always uses the same stroke-order. For example, the stroke-orders of "king" by the same writer were always the same. Thus, we can expect that the stroke-order is useful for forensics.

Toward the realization of forensics systems based on the individuality of stroke-order, we need some methodology to determine the stroke-order of an unknown input pattern automatically and accurately. In fact, without such a methodology, it is difficult to prove that the above preliminary observations are reliable through a very large-scale experiment. In addition, it is also difficult to realize an automatic forensic system that can identify the writer based on stroke-order. The above result of the comparative evaluation of those methods, i.e., CS, BWM, ICD, SM, and DE, will be valuable for selecting the stroke correspondence method for forensics.

4.7 Summary

In this chapter, we have reviewed the methods for solving the stroke-order variation problem. Among various approaches for dealing with this problem, we have focused on the stroke correspondence search methods. Especially, five representative methods — CS, BWM, ICD, SM, and DE, were discussed and compared to clarify their relative superiority. From the viewpoints of not only recognition accuracy but also stroke correspondence accuracy, the five methods have been experimentally compared on the same test set. According to the experimental results under the stroke-number fixed condition, performance superiorities of CS and BWM over ICD, SM, and DE were established. This indicates that global optimal solution under a stroke distance-based criterion is necessary for higher stroke correspondence accuracy. A detailed discussion on the five methods showed that these methods could be applied to different conditions of character recognition tasks, and combined with other approaches for dealing with stroke-order variation problem. Furthermore, we considered the stroke-order variation as a novel feature for forensics identification, and demonstrated that the results of the comparative evaluation of the five methods could be effectively applied to forensics.

Chapter 5

An efficient radical-based algorithm for stroke-order free online Kanji character recognition

5.1 Introduction

As introduced in Chapter 3, the CS proposed by Sakoe and Shin [11, 12] is an effective stroke-order analysis algorithm for online character recognition. In CS, an N-dimensional cube graph stroke-order generation model is defined for N-stroke character to impose the bijection property on the stroke correspondence. Then, the stroke correspondence search problem is formulated as an optimal path search problem on the cube graph. An efficient dynamic programming (DP) is used to search for the shortest path on the cube graph, with $O(N \cdot 2^{N-1})$ time complexity. However, for multi-stroke character with large N (e.g., Chinese character or Japanese Kanji character), a more efficient algorithm is hoped for.

It is well known that Kanji characters are composed of limited number of radicals [1]. We redefine the character model as a set of component radicals and break down the analysis process into two levels — radical level and character level. This decomposition is based on the same principle as that of two-level connected speech recognition algorithm by Sakoe [81, 82]. By this reorganization the original cube graph is decomposed into several small scale intra-radical graphs and a small scale inter-radical graph, and allows considerable reduction in complexity. It also allows a higher recognition accuracy by omitting such unrealistic stroke transitions that connect two consecutive radicals without completing the former radical. In fact, according to our investigation, such stroke transitions are rare and only 26 samples are found in our 17,983-character online dataset of stroke-number fixed, based on our radical definition.

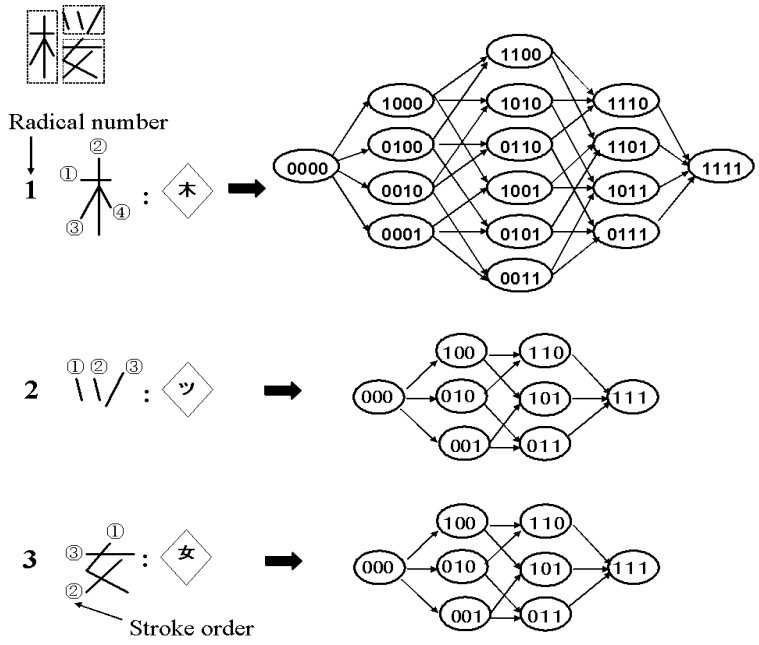
There are a variety of reported radical-based online character recognition algorithms, e.g., [21, 83, 84]. In this chapter, we report a novel, efficient and stroke-order free recognition algorithm based on CS and radical-based reference model.

5.2 Radical-based cube search

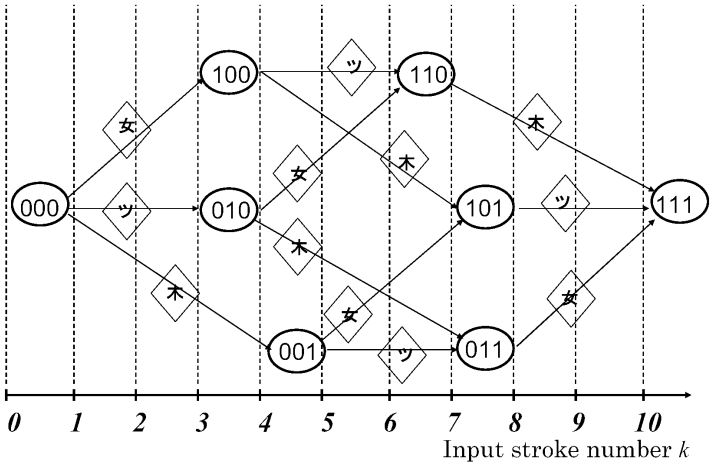
The main idea is described as following: On the basis of radicals, we redefine the reference character as,

$$B = R_1 R_2 \dots R_p \dots R_m, \tag{5.1}$$

[1]The radical does not remain in the morphological definition. Radicals in this chapter may artificially be designed with the objective of efficiency or accuracy improvement.



(a) Intra-radical cube graphs



(b) Inter-radical cube graph

Figure 5.1: Radical-based cube search graph (character "桜", N=10, m=3, n_1=4, n_2=n_3=3).

where, R_p is the pth reference radical of B with n_p strokes, and $\sum_{p=1}^{m} n_p = N$. We consider an intra-radical cube graph that generates intra-radical stroke-order for each of these radicals. We further consider an inter-radical cube graph that controls bijective property of radical-to-radical correspondence. In the inter-radical cube graph, an edge stands for a radical, and node flags control the radical-to-radical match. This reorganization is based upon a reasonable assumption that, the stroke transition that connects two consecutive radicals without completing the former radical, is unrealistic. If we consider the exponential property of cube graph complexity, considerable simplification can be expected, by this reorganization.

Figure 5.1 gives an example of character "桜" that is divided into three radicals "木", "ツ", and "女". In conventional CS, for 10-stroke character "桜", a cube graph with $10 \cdot 2^9 = 5120$ edges is required. When decomposition is applied, three radicals "木", "ツ", and "女" require 32, 12, and 12 edges cube graph, respectively (see Fig. 5.1(a)). The inter-radical cube graph includes 12 edges. By substituting intra-radical graphs for corresponding edges in the inter-radical graph, there exist a total of $4\times32+4\times12+4\times12+12 = 236$ edges. Thus, the computational burden for stroke-order search is considerably reduced.

Based on the two-level DP principle [81, 82], we propose the following search algorithm comprising a radical level process and a character level process.

Step 1 Radical Level Process :

Corresponding to n_p-stroke reference radical pattern, n_p-stroke input stroke sequence starting from input stroke $s+1$ is hypothesized as an input radical,

$$A(s, n_p) = A_{s+1}A_{s+2}\ldots A_{s+n_p}. \tag{5.2}$$

A stroke-based CS is carried out between $A(s, n_p)$ and reference radical R_p on the intra-radical cube graph to obtain the dissimilarity between $A(s, n_p)$ and R_p as radical distance $\lambda(s, p) = D(A(s, n_p), R_p)$. This is repeated for all possible combinations of s and p.

Step 2 Character Level Process :

Radical distances $\lambda(s, p)$ are summed up according to the inter-radical graph, and the optimum path is searched for which gives the minimum radical distance sum. This minimum value gives character distance,

$$D(A, B) = \min_{p=p(q)} \left[\sum_{q=1}^{m} \lambda(s, p) \right]. \tag{5.3}$$

Constraints are (i) $p(q)$ is bijective and (ii) $\sum n_{p(q)} = N$. These constraints are equivalent to the regulation by inter-radical graph. The minimization is accomplished by DP similar to Eq. 3.5, except for the edge cost is the radical distance $\lambda(s, p)$.

In the implementation, some practical techniques can be applied. From Fig. 5.1(b), we can see that we can limit the calculation of radical distances $\lambda(s, p)$ to those (s, p), where the radical R_p follows to the node at $k = s$. Consider that some considerable number of radicals are common to different characters. A technique can be implemented where radical distances of those common radicals are stored in a suitable table and accessed to when different character is processed, thus can avoid repeated calculations of the same radical distance.

Table 5.1: Experimental results.

Algorithm	Stroke base	Radical base
Recog. rate(%)	99.15	99.31
Calculation time of stroke distances(ms)	39	30
Cube search time(ms)	43	10
Average recog. time(ms)	82	40

5.3 Design policy of reference character

Complexity of the proposed radical-based CS depends on the design of radical units. A simple mathematics derives the following minimality condition for time complexity,

$$m = n_1 = n_2 = \ldots = n_m = \sqrt{N}. \tag{5.4}$$

When we take into account of the integer approximation of Eq. 5.4, the design policy will be to compromise the following requirements.

(1) Radicals in morphological meaning are bases.

(2) Number of radicals m, and the size of each radical n_p should be near by $\sqrt{N}$.

(3) When specific radical is frequently written with stroke transition to external without completing the radical, the radical is to be divided at the transition point.

5.4 Experimental results and discussions

In order to clarify the effectiveness of the proposed algorithm, we conducted experiments on a workstation (DELL Precision WorkStation 530, 1.7GHz). A total of 17,983 stroke-order free Kanji (882 classes, 1 — 20 strokes) written by 30 persons were used as test data with no stroke connection. As the feature information a_{ik} and b_{jl} of each point on the stroke, combination of the coordinates of the pen-point and eight directional vector approximation of the stroke direction was used. Based on the policy of Section 5.3, a set of 1,009 reference radicals were designed. By these radicals reference characters were divided into 1 — 4 reference radicals.

Table 5.1 shows the experimental results for the conventional stroke-based algorithm and the proposed radical-based algorithm, including recognition rate, average stroke distance calculation time, average stroke correspondence analysis time, and average recognition time. The correspondence analysis speed is considerably improved, contributing to a significant improvement in recognition speed. Unignorable accuracy improvement is also attained.

In order to confirm the validity of the Eq. 5.4, we tested morphologically reasonable three decomposition ways of a character "興", shown in Fig. 5.2. This character is of 16 strokes, and the optimum decomposition is $m = 4$, $n_1 = n_2 = n_3 = n_4 = 4$. Type (c) decomposition ($m = 4$, $n_1 = 4$, $n_2 = 6$, $n_3 = n_4 = 3$) with the nearest approximation to this condition attained the highest speed. The test result is shown in Table 5.2.

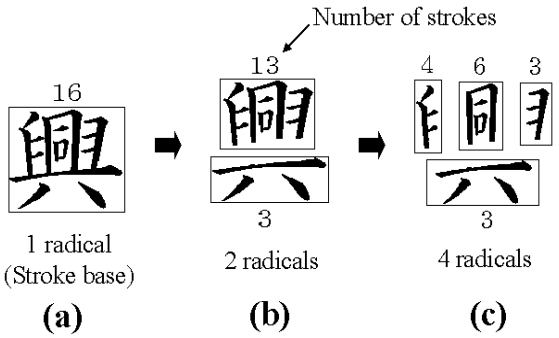


Figure 5.2: Three ways of radical decomposition of character "興".

Table 5.2: Cube search time for an input "興" (ms).

Beam search	without	with
(a) Stroke base	215.20	5.71
(b) 2 radicals	43.15	6.95
(c) 4 radicals	0.62	0.43

5.5 Summary

An improved method is described for stroke correspondence search of online multi-stroke character. Stroke-based cube graph model for generating stroke-orders is decomposed into intra-radical cube graphs and an inter-radical cube graph. An efficient two-level DP algorithm is presented for searching for the optimum path on these graphs, which gives the optimum stroke correspondence. By the radical decomposition, a considerable enhancement in search speed and a significant improvement in recognition accuracy are achieved.

Chapter 6

Conclusion

Recognition of handwritten characters has been a popular research area for many years because of its various application potentials. However, there are still a lot of problems need to be solved. This thesis coped with the problems of online multi-stroke character recognition, and conducted a comparative study for clarifying the relative superiority of five methods of stroke correspondence, and proposed a novel method to reduce the time and spatial complexity of CS.

Firstly, Chapter 2 gave a brief review on the processing of online multi-stroke character recognition, and Chapter 3 intruduced a promising method of stroke correspondence of CS. After that, Chapter 4 began with a brief review on the approaches for solving the stroke-order variation problem in online multi-stroke character recognition. Among various approaches for dealing with this problem, we focused on the stroke correspondence approach. Especially, five representative methods — CS, BWM, ICD, SM, and DE, were discussed and compared to clarify their relative superiority. From the viewpoints of not only recognition accuracy but also stroke correspondence accuracy, the five methods have been experimentally compared on

the same test set. According to the experimental results under the stroke-number fixed condition, performance superiorities of CS and BWM over ICD, SM, and DE were established. For the CS, although it was slower than ICD, SM, and BWM, its recognition speed was also fast and practical by the introduction of pruning strategy of beam search. A detailed discussion on the five methods showed that these methods could be applied to different conditions of character recognition tasks, and combined with other approaches for dealing with stroke-order variation problem. Furthermore, the stroke-order variation was considered as a novel feature for forensics identification. It was demonstrated that the results of the comparative evaluation of the five methods could be effectively applied to forensics.

For the CS, Chapter 5 proposed a novel, efficient and stroke-order free recognition algorithm based on CS and radical-based reference model, to solve the time complexity problem of CS. The basic idea was to utilize the characteristic that Kanji characters are composed of limited number of radicals. Stroke-based cube graph model for generating stroke-orders was decomposed into intra-radical cube graphs and an inter-radical cube graph. An efficient two-level DP algorithm was presented for searching for the optimum path on these graphs, to give the optimum stroke correspondence. By the radical decomposition, a considerable enhancement in search speed and a significant improvement in recognition accuracy were achieved, especially for the character patterns with large number of strokes.

Our future studies will focus on the following points.

- For the stroke correspondence methods (especially CS and BWM), we should treat "stroke-number free" condition, where some strokes are connected into

one stroke by cursive writing. Note that CS has already been extended to deal with connected stroke [11, 12].

- Besides the online character recognition, we should study how to apply the technique to establish the stroke correspondence into other fields that need order analysis of feature sequences, such as DNA analysis and gesture recognition.

- For the proposed algorithm based on CS and radical-based reference model, we should study how to give the optimal radical decomposition design of reference character automatically, and how to treat the "stroke-number free" condition.

References

[1] N. Arica and F.T. Yarman-Vural, "An Overview Of Character Recognition Focused On Off-line Handwriting," IEEE Trans. Systems, Man, and Cybernetics-Part C: Applications and Rev., vol. 31, no. 2, pp. 216-233, 2001.

[2] R. Plamondon and S.N. Srihari, "On-Line and Off-Line Handwriting Recognition: A Comprehensive Survey," IEEE Trans. Pattern Anal. Mach. Intell., vol. 22, no. 1, pp. 63-84, 2000.

[3] K.C. Santosh and C. Nattee, "A Comprehensive Survey on On-line Handwriting Recognition Technology and Its Real Application to The Nepalese Natural Handwriting," Kathmandu University Journal of Science, Engineering and Technology, vol. 5, no. 1, pp. 31-55, 2009.

[4] C.C. Tappert, C.Y. Suen, and T. Wakahara, "The State of the Art in On-Line Handwriting Recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 12, no. 8, pp. 787-808, 1990.

[5] H. Bunke, "Recognition of Cursive Roman Handwriting — Past, Present and Future," Proc. 7th Int. Conf. Document Analysis and Recognition, pp. 448-459, 2003.

[6] S. Madhvanath and V. Govindaraju, "The Role of Holistic Paradigms in Handwritten Word Recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 23, no. 2, pp. 149-164, 2001.

[7] S.N. Srihari, X. Yang, and G.R. Ball, "Offline Chinese handwriting recognition:

an assessment of current technology," Frontiers of Computer Science in China, vol. 1, no. 2, pp. 137-155, 2007.

[8] T. Steinherz, E. Rivlin, N. Intrator, and P. Neskovic, "An Integration of Online and Pseudo-Online Information for Cursive Word Recognition," IEEE Trans. Pattern Anal. Mach. Intell., vol. 27, no. 5, pp. 669-683, 2005.

[9] C.Y. Suen, M. Berthod, and S. Mori, "Automatic recognition of handprinted characters — The state of the art, "Proc. IEEE, vol. 68, pp. 469-487, 1980.

[10] U. Garain, B.B. Chaudhuri, and T.T. Pal, "Online Handwritten Indian Script Recognition: A Human Motor Function based Framework," Proc. 16th Int. Conf. Pattern Recognition, vol. 3, pp. 164-167, 2002.

[11] H. Sakoe and J. Shin, "A Stroke Order Search Algorithm for Online Character Recognition," Research Reports on Information Science and Electrical Engineering of Kyushu University, vol. 2, no. 1, pp. 99-104, 1997 (in Japanese).

[12] J. Shin, and H. Sakoe, "Stroke Correspondence Search Method for Stroke-Order and Stroke-Number Free On-Line Character Recognition — Multilayer Cube Search —," IEICE Trans. Inf. & Syst., vol. J82-D-II, no. 2, pp. 230-239, 1999 (in Japanese).

[13] A.J. Hsieh, K.C. Fan, and T.I. Fan, "Bipartite Weighted Matching for On-line Handwritten Chinese Character Recognition," Pattern Recognition, vol. 28, no. 2, pp. 143-151, 1995.

[14] K. Odaka, T. Wakahara, and I. Masuda, "Stroke Order Free On-line Handwrit-

ten Character Recognition Algorithm," IEICE Trans. Inf. & Syst., vol. J65-D, no. 6, pp. 679-686, 1982 (in Japanese).

[15] T. Wakahara, H. Murase, and K. Odaka, "On-Line Handwriting Recognition," Proc. IEEE, vol. 80, no. 7, pp. 1181-1194, 1992.

[16] T. Yokota *et al.*, "An On-line Cuneiform Modeled Handwritten Japanese Character Recognition Method Free from Both the Number and Order of Character Strokes," IPSJ Journal, vol. 44, no. 3, pp. 980-990, 2003 (in Japanese).

[17] C.K. Lin, K.C. Fan, and F.T.P. Lee, "On-line Recognition by Deviation-expansion Model and Dynamic Programming Matching," Pattern Recognition, vol. 26, no. 2, pp. 259-268, 1993.

[18] W. Cai, S. Uchida, and H. Sakoe, "Toward Forensics by Stroke-Order Variation — Performance Evaluation of Stroke Correspondence Methods," Proc. 4th Int. Workshop Computational Forensics, pp.43-55, 2010.

[19] W. Cai, S. Uchida, and H. Sakoe, "An Efficient Stroke-Order-Free On-Line Character Recognition Algorithm Based on Radical Reference Pattern," IEICE Trans. Inf. & Syst., vol. J88-D-II, no. 7, pp. 1187-1195, 2005 (in Japanese).

[20] W. Cai, S. Uchida, and H. Sakoe, "An Efficient Radical-Based Algorithm for Stroke-Order-Free Online Kanji Character Recognition," Proc. 18th Int. Conf. Pattern Recognition, vol. 2, pp. 986-989, 2006.

[21] C.L. Liu, S.Jaeger, and M. Nakagawa, "Online Recognition of Chinese Characters: The State-of-the-Art," IEEE Trans. Pattern Anal. Mach. Intell., vol. 26, no. 2, pp. 198-213, 2004.

[22] M. Nakagawa, "Non-Keyboard Input of Japanese Text — On-Line Recognition of Handwritten Characters as the Most Hopeful Approach, "J. Information Processing, vol. 13, no. 1, pp. 15-34, 1990.

[23] Y.H. Tay, M. Khalid, and R. Yusof, "Online Chinese Handwritten Character Recognition: A Brief Review," http://citeseer.ist.psu.edu/596162.html.

[24] Industry Research & Statistics, Semiconductor Equipment and Material International (SEMI), June 2002.

[25] C. Bahlmann and H. Burkhardt, "The Writer Independent Online Handwriting Recognition System frog on hand and Cluster Generative Statistical Dynamic Time Warping," IEEE Trans. Pattern Anal. Mach. Intell., vol. 26, no. 3, pp. 299-310, 2004.

[26] W. Xia and L. Jin, "A Kai Style Calligraphic Beautification Method for Handwriting Chinese Character," Proc. 10th Int. Conf. Document Analysis and Recognition, pp. 798-802, 2009.

[27] X. Zhu and L. Jin, "Calligraphic Beautification of Handwritten Chinese Characters: A Patternized Approach to Handwriting Transfiguration," Proc. 11th Int. Workshop Frontiers in Handwriting Recognition, pp. 135-140, 2008.

[28] R. Kashi, J. Hu, W.L. Nelson, and W. Turin, "A Hidden Markov Model Approach To On-line Handwritten Signature Verification," Int. Journal Document Analysis and Recognition, vol. 1, no. 2, pp.102-109, 1998.

[29] J.G.A. Dolfing, E.H.L. Aarts, and J.J.G.M. Van Oosterhout, "On-Line Signa-

ture Verification with Hidden Markov Models," Proc. 14th Int. Conf. Pattern Recognition, pp. 1309-1312, 1998.

[30] K. Huang and H. Yan, "On-Line Signature Verification Based on Dynamic Segmentation and Global and Local Matching," Optical Eng., vol. 34, no. 12, pp. 3480-3488, 1995.

[31] U. Pal and B.B. Chaudhuri, "Indian script character recognition: a survey," Pattern Recognition, vol. 37, no. 9, pp. 1887-1899, 2004.

[32] K. Roy, N. Sharma, T. Pal, and U. Pal, "Online Bangla Handwriting Recognition System," Proc. 6th Int. Conf. Advances in Pattern Recognition, pp. 117-122, 2007.

[33] C. L. Liu and C. Y. Suen, "A new benchmark on the recognition of handwritten Bangla and Farsi numeral characters," Pattern Recognition, vol. 42, no. 12, pp. 3287-3295, 2009.

[34] R.J. Kannan and R. Prabhakar, "Off-Line Cursive Handwritten Tamil Character Recognition," WSEAS Trans. on Signal Processing, vol. 4, no. 6, pp. 351-360, 2008.

[35] I. Guyon, L.R.B. Schomaker, R. Plamondon, M. Liberman, and S. Janet, "UNIPEN Project of On-Line Data Exchange and Recognizer Benchmarks, "Proc. 12th Int. Conf. Pattern Recognition, pp. 29-33, 1994, http://www.unipen.org/.

[36] H. Zhang and C. L. Liu, "A Lattice-Based Method for Keyword Spotting in

Online Chinese Handwriting," Proc. 11th Int. Conf. Document Analysis and Recognition, pp. 1064-1068, 2011.

[37] H. Zhang, D.H. Wang, and C.L. Liu, "Keyword Spotting from Online Chinese Handwritten Documents Using One-vs-All Trained Character Classifier," Proc. 12th Int. Workshop Frontiers in Handwriting Recognition, pp. 271-276, 2010.

[38] T.F. Gao and C.L. Liu, "LDA-Based Compound Distance for Handwritten Chinese Character Recognition," Proc. 9th Int. Conf. Document Analysis and Recognition, pp. 904-908, 2007.

[39] L.K. Welbourn and R.J. Whitrow, "A Gesture Based Text and Diagram Editor," Computer Processing of Handwriting. R. Plamondon and C.G. Leedham, eds., pp. 221-234, Singapore, World Scientific, 1990.

[40] C.G. Wolf and P. Morrel-Samuels, "The Use of Hand-Drawn Gestures for Text Editing," Proc. Int. J. Man-Machine Studies, vol. 27, pp. 91-102, 1987.

[41] A. Hennig, N. Sherkat, and R.J. Whitrow, "Zone Estimation for Multiple Lines of Handwriting Using Approximating Spline Functions," Proc. 5th Int. Workshop Frontiers in Handwriting Recognition, pp. 325-328, 1996.

[42] R.K. Powalka, N. Sherkat, and R.J. Whitrow, "Word Shape Analysis for a Hybrid Recognition System," Pattern Recognition, vol. 30, no. 3, pp. 421-445, 1997.

[43] J. Wang, M.K.H. Leung, and S.C. Hui, "Cursive Word Reference Line Detection," Pattern Recognition, vol. 30, no. 3, pp. 503-512, 1997.

[44] Y. Lu and M. Shridhar, "Character segmentation in handwritten words — An overview," Pattern Recognition. vol. 29, pp. 77-96, 1996.

[45] D.H. Wang, C.L. Liu, J.L. Yu, and X.D. Zhou, "CASIA-OLHWDB1: A Database of Online Handwritten Chinese Characters," Proc. 10th Int. Conf. Document Analysis and Recognition, pp. 1206-1210, 2009.

[46] R. Plamondon, D. Lopresti, L.R.B. Schomaker, and R. Srihari, "On-Line Handwriting Recognition," Encyclopedia of Electrical and Electronics Eng., J.G. Webster, ed., vol. 15, pp. 123-146, New York:Wiley, 1999.

[47] S. Clergeau-de-Tournemire and R. Plamondon, "Integration of Lexical and Syntactical Knowledge in a Handwriting Recognition System," Machine Vision and Applications, special issue cursive script recognition, vol. 8, no. 4, pp. 249-260, 1995.

[48] K.C. Santosh and C. Nattee, "Structural Approach on Writer Independent Nepalese Natural Handwriting Recognition," 2nd IEEE Int. Conf. Cybernetics Intelligent Systems, pp. 711-716, 2006.

[49] K.S. Nathan, H.S.M. Beigi, J. Subrahmonia, G.J. Clary, and H. Maruyama, "Real-time On-line Unconstrained Handwriting Recognition Using Statistical Methods," Proc. 1995 Int. Conf. Acoustics, Speech, and Signal Processing, vol. 4, pp. 2619-2622, 1995.

[50] Y. Katayama, S. Uchida, and H. Sakoe, "A New HMM for On-Line Character Recognition Using Pen-Direction and Pen-Coordinate Features," Proc. 19th Int. Conf. Pattern Recognition, 2008.

[51] Y. Katayama, S. Uchida, and H. Sakoe, "An HMM Representing Stroke Order Variations and Its Application to Online Character Recognition," IEICE Trans. Inf. & Syst., vol. J91-D, no. 5, pp. 1434-1441, 2008 (in Japanese).

[52] M. Nakai, H. Shimodaira and S. Sagayama, "Generation of Hierarchical Dictionary for Stroke-order Free Kanji Handwriting Recognition Based on Substroke HMM", Proc. 7th Int. Conf. Document Analysis and Recognition, pp. 514-518, 2003.

[53] M. Nakai, N. Akira, H. Shimodaira, and S. Sagayama, "Substroke Approach to HMM-Based On-Line Kanji Handwriting Recognition", Proc. 6th Int. Conf. Document Analysis and Recognition, pp. 491-495, 2001.

[54] J.W. Chen and S.Y. Lee, "A Hierarchical Representation for the Reference Database of On-Line Chinese Character Recognition, "Advances in Syntactic and Structural Pattern Recognition, P. Perner, P. Wang, and A. Rosenfeld, eds., pp. 351-400, Springer, 1996.

[55] J.W. Chen and S.Y. Lee, "On-Line Handwriting Recognition of Chinese Characters via Rule-Based Approach," Proc. 13th Int. Conf. Pattern Recognition, vol. 3, pp. 220-224, 1996.

[56] Y.J. Liu and J.W. Tai, "Structural Approach to On-Line Chinese Character Recognition, "Proc. 9th Int. Conf. Pattern Recognition, pp. 808-810, 1988.

[57] A.D. Eric and A.S. Mace, "Explicit fuzzy modeling of shapes and positioning for handwritten Chinese character recognition," Proc. 10th Int. Conf. Document Analysis and Recognition, pp. 1121-1125, 2009.

[58] S. Tulyakov, S. Jaeger, V. Govindaraju, and D. Doermann, "Review of Classifier Combination Methods," Studies in Computational Intelligence: Machine Learning in Document Analysis and Recognition, vol. 90, pp. 361-386, 2008.

[59] A.F.R. Rahman and M.C. Fairhurst, "Multiple classifier decision combination strategies for character recognition: A review," Int. J. Document Analysis and Recognition, pp. 166-194, 2003.

[60] A.F.R. Rahman and M.C. Fairhurst, "Multiple expert classification: A new methodology for parallel decision fusion," Int. J. Document Analysis and Recognition, pp. 40-55, 2000.

[61] A.F.R. Rahman, H. Alam, and M.C. Fairhurst, "Multiple classifier combination for character recognition: Revisiting the majority voting system and its variations," In D. Lopresti, J. Hu, and R. Kashi, editors, 5th Int. Workshop Document Analysis Systems, pp. 167-178, LNCS 2423, Springer, 2002.

[62] K. Takahashi, H. Yasuda, and T. Matsumoto, "On-line Handwritten Character Recognition Using Hidden Markov Model," IEICE Japan, Technical Report, PRMU96-211, pp. 143-150, 1997 (In Japanese).

[63] H. Sakoe, "Cube Search — Stroke Order Search Algorithms for Online Character Recognition," IEICE Japan, Technical Report, PRU95-112, 1995 (In Japanese).

[64] H. Sakoe and J.P. Shin, "An Stroke Order Search Algorithm for Online Character Recognition," IEICE Japan, Technical Report, PRU95-59, 1995 (In Japanese).

[65] J.P. Shin and H. Sakoe, "Cube Search Algorithm for Stroke-Order and Stroke-Number Free Online Character Recognition," IEICE Japan, Technical Report, PRMU96-84, 1996 (In Japanese).

[66] J.J. Lee, J. Kim, and J.H. Kim, "Data-driven design of HMM topology for online handwriting recognition", Int. J. Pattern Recognition and Artificial Intelligence, vol. 15, no. 1, pp. 107-121, 2001.

[67] H. Tanaka, et al., "Hybrid Pen-Input Character Recognition System Based on Integration of Online-Offline Recognition", Proc. 5th Int. Conf. Document Analysis and Recognition, pp. 209-212, 1999.

[68] H. Oda, et al., "A Compact On-line and Off-line Combined Recognizer", Proc. 10th Int. Workshop Frontiers in Handwriting Recognition, pp. 133-138, 2006.

[69] J. Liu, W.K. Cham, and M.M.Y. Chang, "Stroke Order and Stroke Number Free On-Line Chinese Character Recognition Using Attributed Relational Graph Matching," Proc. 13th Int. Conf. Pattern Recognition, vol. 3, pp. 259-263, 1996.

[70] J. Zheng, X. Ding, and Y. Wu, "Recognizing On-Line Handwritten Chinese Character via FARG Matching," Proc. 4th Int. Conf. Document Analysis and Recognition, pp. 621-624, 1997.

[71] J. Zheng, X. Ding, Y. Wu, and Z. Lu, "Spatio-Temporal Unified Model for On-Line Handwritten Chinese Character Recognition," Proc. 5th Int. Conf. Document Analysis and Recognition, pp. 649-652, 1999.

[72] K.S. Chou, K.C. Fan, and T.I. Fan, "Radical-Based Neighboring Segment

Matching Method for On-Line Chinese Character Recognition," Proc. 13th Int. Conf. Pattern Recognition, vol. 3, pp. 84-88, 1996.

[73] M. J. Joe and H. J. Lee, "A Combined Method on the Handwritten Character Recognition", Proc. 3rd Int. Conf. Document Analysis and Recognition, pp. 112-115, 1995.

[74] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm Optimization for Spoken Word Recognition," IEEE Trans. Acoust., Speech, Signal Processing, vol. ASSP-26, no. 1, pp. 43-49, 1978.

[75] H. W. Kuhn, "The Hungarian Method for the Assignment Problem," Naval Research Logistics Quarterly, vol. 2, pp. 83-97, 1955.

[76] J. Munkres, "Algorithms for the Assignment and Transportation Problems," J. Soc. Indust. Appl. Math., vol. 5, no. 1, pp. 32-38, 1957.

[77] C. H. Papadimitriou and K. Steiglitz, Combinatorial Optimization: Algorithms and Complexity, Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[78] R. Sedgewick, Algorithms, Addison-Wesley, second edition, pp. 499-504, 1988.

[79] M. Nakagawa and K. Matsumoto, "Collection of on-line handwritten Japanese character pattern databases and their analysis", Int. J. Document Analysis and Recognition, vol. 7, no. 1, pp. 69-81, 2004.

[80] C.L. Liu, F. Yin, D.H. Wang, and Q.F. Wang, "CASIA Online and Offline Chinese Handwriting Databases," Proc. 11th Int. Conf. Document Analysis and Recognition, pp. 37-41, 2011.

[81] H. Sakoe, "Two-Level DP-Matching—A Dynamic Programming-Based Pattern Matching Algorithm for Connected Word Recognition," IEEE Trans. Acoust. Speech Signal Process., vol. ASSP-27, no. 6, pp. 588-595, 1979.

[82] H. Sakoe, "A Generalized Two-Level DP-Matching Algorithm for Continuous Speech Recognition," Trans. IECE Japan, vol. E65, no. 11, pp. 649-656, 1982.

[83] A. Kitadai, and M. Nakagawa, "Prototype Learning for Structured Character Pattern Representation Used in On-Line Recognition of Handwritten Japanese Characters," IEICE Trans. Inf. & Syst., vol. J86-D-II, no. 1, pp. 1-11, 2003 (in Japanese).

[84] X. H. Xiao and R. W. Dai, "On-Line Handwritten Chinese Character Recognition Directed by Components with Dynamic Templates," Int. J. Pattern Recognition and Artificial Intelligence, vol. 12, no. 1, pp. 143-157, 1998.