

Regular Article

# Extraction of Dynamic Trajectory on Multi-Stroke Static Handwriting Images Using Loop Analysis and Skeletal Graph Model

Vo Anh Kha<sup>1</sup>, Ha Hoang Kha<sup>1</sup>, Michael Blumenstein<sup>2</sup>

<sup>1</sup> Dept. Electronics and Electrical Engineering, Ho Chi Minh City University of Technology, Vietnam

<sup>2</sup> Sch. Information and Communication Technology, Griffith University, Queensland, Australia

Correspondence: Vo Anh Kha, voanhkha@yahoo.com

Communication: received 26 October 2015, revised 15 January 2016, accepted 9 May 2016

Online publication: 24 August 2016, Digital Object Identifier: 10.21553/rev-jec.131

The associate editor coordinating the review of this article and recommending it for publication was Dr. Le Vu Ha.

**Abstract**– The recovery of handwriting’s dynamic stroke is an effective method to help improve the accuracy of any handwriting’s authentication or verification system. The recovered trajectory can be considered as a dynamic feature of any static handwritten images. Capitalising on this temporal information can significantly increase the accuracy of the verification phase. Extraction of dynamic features from static handwritings remains a challenge due to the lack of temporal information as compared to the online methods. Previously, there are two typical approaches to recover the handwriting’s stroke. The first approach is based on the script’s skeleton. The skeletonisation method has highly computational efficiency whereas it often produces noisy artifacts and mismatches on the resulted skeleton. The second approach deals with the handwriting’s contour, crossing areas and overlaps using parametric representations of lines and thickness of strokes. This method can avoid the artifacts, but it requires complicated mathematical models and may lead to computational explosion. Our paper is based on the script’s extracted skeleton and provides an approach to processing static handwriting’s objects, including edges, vertices and loops, as the important aspects of any handwritten image. Our paper is also to provide analysing and classifying loops types and human’s natural writing behavior to improve the global construction of stroke order. Then, a detailed tracing algorithm on global stroke reconstruction is presented. The experimental results reveal the superiority of our method as compared with the existing ones.

**Keywords**– Handwriting recognition, trajectory recovery, off-line system, skeletonisation, stroke order recovery.

## 1 INTRODUCTION

Handwriting recognition can be categorized into two different types, online and offline, based on the method of inputting and the available temporal information of the handwriting. Online methods use the tablet digitizer with a pen to record the movement and other aspects of the handwriting such as pressure, velocity and drawing order as a function of time, while in offline recognition systems, a scanned (static) two-dimensional image of text or signature [1–3] is obtained and processed without any additional temporal information. Therefore, online methods have found in various applications, for example, signature verification in banking transactions, handwritten word recognition for student identification [4] or a numerous of handwritten word recognition software on smart phones. Offline methods remain the technical challenges due to the unavailability of temporal information of pen-tip movement [5–8]. From the relations between special points, edges, intersections of a static handwriting, one can properly recover the pen-tip trajectory of the writer. The extraction and recovery of dynamic information, especially the stroke order, from static images therefore plays an important role in improving the performance of off-line systems.

### 1.1 Previous Works

Yasuhara *et al.* proposed a general and thorough method of the handwriting’s skeleton analysis [9]. This approach requires a set of heuristic rules and a complex tracing algorithm after implementing the process of skeleton analysis. This method could recover handwritings with a single stroke, many double-traced lines and distinct start and end points. However, it does not evaluate all the possible combinations, especially the relations between loops and their neighboring edges. Yasuhara’s method cannot be applied on multi-stroke handwritings. Alternatively, Steinherz *et al.* [10] focused on the investigation and detection of off-line loops without involving any thinning process. This approach is based on a novel loop theory with sophisticated algorithms on the contour of the handwriting. Although the experimental results of this approach were not presented in their paper, their method however may help the researchers have the basis on various loop resolution scenarios. L’Homer [11] proposed a model of strokes that a stroke is a family of a disc, its radius and its position. By this model, L’Homer completely excludes the process of skeletonisation and focuses on the characteristics of stroke crossing types. This method is best used with handwriting’s simple texts or isolated characters but it is not suitable for the hand-

writings which include many overlapped strokes like signatures. Doermann *et al.* [12] proposed a method to recover hidden loops based on the contour-dependent blob analysis. This approach deals with the contour's shape investigation. Qiao and Nishiara [5] studied on the skeletal image with a new framework of "Edge Continuity Relation" to categorize the skeletal nodes of the handwriting into different types. This approach has some encouraging restoration results, but also has several drawbacks on some handwritings with noisy artifacts, unidentified start node, misidentified nodes or high-degree nodes.

## 1.2 Our Major Contribution









In this paper, we introduce a method to restore the stroke order not only from single-stroke handwritings, but also from multi-stroke static handwritten images. In addition, we propose a novel approach based on loops generated in the handwriting to improve the trajectory tracing performance. Human's natural writing behavior can generate loops in writing by various ways, specifically in Latin characters and digits. A deep analysis in this problem could better restore the stroke order and significantly increases the accuracy in the recognition or verification process of the later phases. In comparison with other existing methods, we construct a more compact skeletal graph model by categorizing human's natural writing behavior in different manners, with vertices, edges and loops as the main targets. Our approach not only proposes a systematic approach on the skeleton's analysis and tracing algorithm, but also aims to resolve the problems of the restoration and recognition in a general and clear framework of algorithms. Unlike most previous works which focused only on single-stroke images, our method can be applied to multi-stroke handwritings, therefore can alleviate the problems of decomposing the handwritings into sub-problems with each one dealing with one stroke.

The rest of this paper is organized as follows. Section 2 presents the preprocessing techniques applied on the raw handwriting and the skeletonisation problems. Section 3 describes the method of modeling the handwriting's skeleton in details and deep investigation on loop types. The problem of detecting start vertex, local loop tracing method and the global tracing scheme is presented in Section 4. The experimental results, comparisons with other methods and further discussion are shown in Section 5. Finally, Section 6 derives the paper's conclusions.

## 2 STATIC IMAGE PROCESSING

In this section, we discuss the problems of the static handwriting's stroke recovery including the assumptions of the handwritings, the preprocessing methods as well as the usually occurring errors of skeletonisation process. The input image must satisfy the pre-defined assumptions to prevent errors occurring during the global stroke tracing phase. The static image is then

Table I  
ASSUMPTION SUMMARY OF HANDWRITINGS

	Satisfy	Violate
1	 (a) Clear start and end points	 (b) No start and end points
2	 (c) 2-line intersection	 (d) 3-line intersection
3	 (e) Double-traced line, indicated by the thin ellipse	 (f) Triple-traced line, indicated by the thin ellipse
4	 (g) Two separate loops	 (h) Two overlapped loops

preprocessed to remove noisy artifacts and, finally it will be skeletonised before going to the process of the skeletal graph modeling in Section 4.

To be successfully recovered of the handwriting's stroke, these assumptions are made for any input image:

- 1) *Clear terminal points*: The stroke must have a distinctive start point and a distinctive end point.
- 2) *2-line intersection*: If there exists any intersection, it only consists of 2 intersecting lines.
- 3) *2-times traced*: There exists no line which is traced more than twice.
- 4) *No loop overlap*: If there exists greater than 2 loops in the handwriting, they must be separate from each other.

All of the sample scripts that satisfy and violate the assumptions are shown in Table I.

Before the process of skeletonisation, the handwriting must be preprocessed to remove the undesirable noisy artifacts. The scanned image of the handwriting is



Figure 1. A handwritten letter 'h' and its skeleton

converted into a gray-level one. Then, it is turned into a binary image, in which each pixel is presented by a '0' or a '1' corresponding to a black or a white pixel, respectively. An image smoothing technique is used to fill all the small holes in the image comparing to the predefined size threshold of a hole. Possible noise in the input such as spur points and isolated point clusters is also removed from the image by a filtering operation.

Skeletonisation, or thinning process is one of the most essential process that should be focused in this paper. The skeletal image of the binary handwriting is extracted using a classical thinning algorithm [13]. In other words, the handwriting is shrunk into a 1-pixel-width skeleton. The skeletal image is the basis to detect start points (pen-down events), end points (pen-up events), branch points and edges of the handwriting. A more complex and reliable thinning algorithm called "Voronoi Skeletonisation" [14] appears largely invariant with respect to typical noise conditions in the image and geometric transformations. This paper will provide an insight at skeletonisation problems which usually occur when applying the ordinary classical thinning algorithm.

An illustrative example of the skeletonisation process is shown in Figure 1. As depicted in the figure, letter 'h' is written by hand and its stroke satisfies all four assumptions. One can see that the stroke contains one clear start point, one clear end point, one loop and one double-traced line. Its skeletal image is the 1-pixel-width graph of the image itself. The skeletal double-traced line now has the same width with the single-traced line although in the original image, the double-traced line is easier to be recognized. The problem of detecting special aspects of the skeleton and constructing the global trajectory will be discussed in a later section of this paper.

Achieving the skeleton should ease up the detection of start point, end point, branch points and edges of the handwriting. To be more precise, the following definitions are used throughout this paper:

- *Adjacent*: 2 pixels are called 'adjacent' to each other if they are the 8-neighbor of each other.
- *Terminal pixel, line pixel, or branch pixel*: a pixel which is adjacent to only one pixel, to exactly two pixels, or to more than two pixels, respectively. A terminal pixel may be a start pixel or an end pixel of the trajectory, or may be an end pixel of any double-traced line.
- *Edge*: a continuous cluster of adjacent line pixels

that construct a continuous line and connect two intersections.

- *Intersection*: a cluster of adjacent branch pixels that represent a specific intersection of two edges.
- *Vertex*: a vertex may be a terminal pixel or an intersection.
- *Degree* (of a vertex): the number of edges starting from a specific vertex. An intersection may have the degree of 3 or 4, while a terminal has the degree of 1.
- *Loop*: A set of vertices and edges that circles a closed region of the handwriting.

The thinning process may result in several unexpected errors in the skeletal image [15]. Some specific techniques are needed to be applied to the skeleton to remove L-connection pixels (which are the pixels with three 8-neighbors but are considered line pixels) and spur pixels (which are the pixels with only one 8-neighbor but are not terminal pixels). One should note these errors would massively affect the result of the trajectory recovery process which will be mentioned later in this paper.

### 3 SKELETAL GRAPH MODEL

This section presents the mathematical model of the handwriting's skeleton and provides the investigation on handwriting's loops.

#### 3.1 Modeling the Handwriting's Skeleton

In order to theorize the global trajectory recovery scheme, we construct a model of the skeletal graph of the handwriting based on the basis of Y. Kato and M. Yasuhara [9]. We propose a new object identity named *loop* and new theory about the relations between objects in the skeleton. A skeleton is modeled by a graph

$$G = (V, E, L), \quad (1)$$

where  $V = \{v_1, v_2, \dots, v_m\}$  is a set of  $m$  vertices. Each vertex is represented by a terminal pixel, or an intersection.  $E = \{e_1, e_2, \dots, e_n\}$  represents a set of  $n$  edges, and  $L = \{l_1, l_2, \dots, l_p\}$  is a set of  $p$  loops. One can easily see that vertices are the most basic objects in the skeletal graph, which composes of edges and loops. Therefore each individual vertex  $v_k$  must have the specific attributes, or characteristics, given by

$$v_k = \{(c_k^x, c_k^y), \rho(v_k), \theta(v_k), (b_k^1, b_k^2, \dots, b_k^{\rho(v_k)})\}, \quad (2)$$

where  $(c_k^x, c_k^y)$  is the two-dimensional coordinate positions of the vertex  $v_k$ . If  $v_k$  is a terminal pixel then its coordinate positions are the positions of the pixel itself. In case  $v_k$  is an intersection, its coordinate positions are the average position of the individual pixels comprised of the intersection.  $\rho(v_k)$  is the degree of  $v_k$ , and  $\theta(v_k)$  is the number of times that the tracing algorithm meets the vertex  $v_k$ .  $\theta(v_k)$  is used in the trajectory recovery algorithm as a variable which will be mentioned in Section 4. At first,  $\theta(v_k)$  is equal to 0. During the process of trajectory tracing,  $\theta(v_k)$  is increased by one

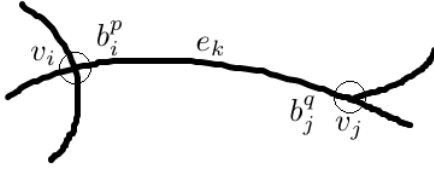


Figure 2. The adjacency relations of skeletal objects

each time whenever it is met by the tracing algorithm, while  $\rho(v_k)$  is kept the same as its initial value. Finally,  $(b_k^1, b_k^2, \dots, b_k^{\rho(v_k)})$  is a set of  $\rho(v_k)$  numbered branches starting from vertex  $v_k$ . Each edge  $e_k$  is composed of two unordered branches  $b_i^p$  and  $b_j^q$  which are belong to vertices  $v_i$  and  $v_j$ , respectively

$$e_k = \{b_i^p, b_j^q\}. \quad (3)$$

With the definitions mentioned earlier in this section, we notice that vertex  $v_i$  is *adjacent* to vertex  $v_j$  (if  $i \neq j$ ) (and vice-versa), as denoted by

$$\alpha(v_i) = v_j \text{ or } \alpha(v_j) = v_i. \quad (4)$$

In other words, branch  $b_i^p$  is *adjacent* to branch  $b_j^q$ , and they share a common edge  $e_k$  as denoted by

$$b_i^p \xrightarrow{e_k} b_j^q \text{ or } b_j^q \xrightarrow{e_k} b_i^p. \quad (5)$$

The above adjacency relation between labeled edges and vertices is depicted in Figure 2.

Regarding the loop's definition in the previous parts of this section, a loop  $l_k$  is represented by  $a$  edges  $e_{k1}, \dots, e_{ka}$  and  $b$  vertices  $v_{k1}, \dots, v_{kb}$ , which results in

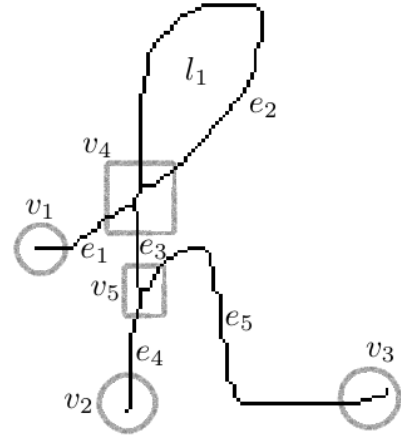
$$l_k = \{e_{k1}, \dots, e_{ka}, v_{k1}, \dots, v_{kb}\} \quad (6)$$

where  $(a, b) \in \{(1,1), (2,2)\}$ . Equation (6) demonstrates that some edges and vertices may be sub-components of a loop. These edges and vertices are needed to be excluded from the skeletal graph as they are already presented in a loop. Figure 3 demonstrates all the objects' definitions mentioned in this section. From this figure, we can summarize the personal data collected after the process of skeletal graph model construction:

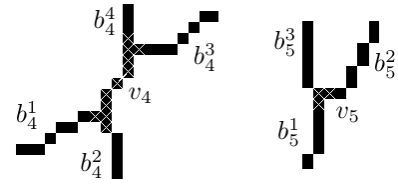
$$\left. \begin{array}{l} e_1 = \{b_1^1, b_4^1\} \\ e_3 = \{v_5^3, b_4^2\} \\ e_4 = \{v_2^1, v_5^1\} \\ e_5 = \{v_3^1, v_5^2\} \\ l_1 = \{e_2, v_4\} \end{array} \right\} \text{ Skeletal graph of Figure 3}$$

### 3.2 Loop Investigation

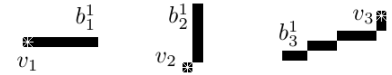
Because of different human's natural writing styles, loops in handwritten scripts can be generated in different ways. Using a general trajectory tracing algorithm for all loop types without taking the categorization of loops into consideration may negatively affect the recovery results. This section provides a basic approach



(a) The skeleton of the handwritten letter 'h' presented by black thin lines. Two gray rectangles labeled  $v_4$  and  $v_5$  indicate two intersections in this skeleton, as shown in details in Figure 3(b). Three gray circles labeled  $v_1$ ,  $v_2$  and  $v_3$  indicate three terminal pixels, as shown in Figure 3(c). Loop  $l_1 = \{e_2, v_4\}$  is treated as a separate object and its composing components  $e_2, v_4$  are removed from the skeletal graph model.



(b) Two intersections named  $v_4$  and  $v_5$  as indicated in Figure 3(a). An intersection is marked by a group of adjacency branch pixels with symbol  $\times$  and each branch is labeled from  $b_4^1$  to  $b_4^4$  for vertex  $v_4$  and  $b_5^1$  to  $b_5^3$  for vertex  $v_5$ .



(c) Three terminal pixels as indicated in Figure 3(a). A terminal pixel is marked by symbol  $*$  and its only branch is labeled.

Figure 3. Typical examples of different objects in the skeletal graph of a handwriting letter 'h'.

to categorize loops into different types when each loop type has some common characteristics. A loop's pixel may be part of an edge or a vertex of that loop. Different number of composing vertices and adjacent edges results in different types of loops as shown in Table II.

The reason why we define a loop as a separate skeletal object besides edges and vertices is that we aim at recovering the local trajectory of a loop, which has two end points like an edge, before tracing the global trajectory. In this manner, one can treat a loop as an edge which has the chain of adjacent pixels which create the edge's trajectory. There should be noticed that some loop's pixels may be traced more than once to construct its local trajectory, but a whole loop treated as one handwritten object is only traced once during the global trajectory construction phase. This is clearly different to an edge that can be traced more than once during the global trajectory construction phase, but the pixels of an edge are traced only once to build its local trajectory.

Table II  
SUMMARY OF HANDWRITING'S LOOP TYPES

Loop type	Number of vertices	Number of adjacent edges	Examples
1	1	2	
2	2	2	

As shown in Table II, loops can be classified into two types based on the number of vertices and adjacent edges. Loop type 1 has one vertex and two edges starting from that vertex. This type of loop is most usually generated by starting at one edge, going through the loop and end at the other edge, like the left example image. However in some special cases like the right example image, the loop type 1 can be traced at the vertex and then through the loop pixels first, then continue to go out and then go in at one edge (this edge is double-traced) and finally go out at the other edge. This sub-type of loop type 1 mostly presents in handwriting letter 'd'. Loop type 2 consists of two different vertices and, therefore, it has two adjacent edges, each starts from one vertex. As shown in the example images, the loop itself has an edge which connects two vertices. This edge is excluded from the global skeletal graph because it already presents in the loop. One should be noted that this section only generally describes each type. Section 4 will deeply analyze each type and provide a scheme to recover the local trajectory of each loop type.

## 4 TRAJECTORY TRACING SCHEME

This section presents the global stroke tracing scheme, including the start vertex detection, the global tracing algorithm and the local loop tracing algorithm. Figure 4 shows the sequence of steps of our approach. First, the static image is preprocessed to remove noisy artifacts before being skeletonised to extract its skeletal image. Some techniques to detect the edges, vertices and loops help to construct the skeletal graph model of the handwriting, based on the analysis on each individual pixel and its neighboring relation of the image [15]. Next, the start vertex of the written stroke must be identified. If the handwriting has any loop, the next process is to find the loops' stroke. Finally, an algorithm is proposed to construct the global stroke of the handwritten image.

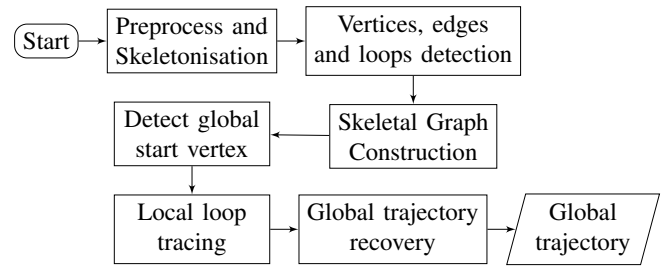


Figure 4. Diagram chart of our method to recover the handwriting's stroke.

### 4.1 Start Vertex Detection

One important task that importantly affects the recovery result is to detect the start vertex of the handwriting. Y. Kato and M. Yasuhara's work [9] assumes that the handwriting must have the distinct start and end vertex. This section presents a scheme to detect the start point of the handwriting's stroke, including the ones with multiple strokes. We assume that people usually start to writing at the top-left region of the paper. Then the start vertex should be the vertex with degree 1 that appears in the top-left region of the image. In multi-stroke handwritten images, once the tracing algorithm reaches the end vertex of the first stroke, the start vertex operation is performed again by the same way to detect the start vertex of the next stroke, by estimating the remain untraced degree-1 vertices.

### 4.2 Global Trajectory Tracing

Let  $v$  denote the current processing vertex of the tracing algorithm and  $v'$  is the next vertex after selecting the correct path at the current vertex  $v$ . Let  $b_{in}$  and  $b_{out}$  be the input and output branch of the current vertex, respectively. Our tracing algorithm selects the output branch  $b_{out}$  after the analysis process at the current vertex  $v$  and its input branch  $b_{in}$ , as denoted by

$$b_{in} \Rightarrow b_{out}. \quad (7)$$

The output branch  $b_{out}$  of the current vertex shares a common *edge* to a new input branch of the next vertex  $v'$ , denoted by  $b'_{in}$ . Combining Equation (5) and (7) yields

$$b_{in} \Rightarrow b_{out} \xrightarrow{edge} b'_{in}. \quad (8)$$

Equation (8) is used when  $v$  is not part of a loop. If  $v$  is a member of a loop, then the local trajectory of the loop is extracted before the algorithm reaches the output branch  $b_{out}$

$$b_{in} \rightarrow \text{local loop trajectory} \rightarrow b_{out} \xrightarrow{edge} b'_{in}. \quad (9)$$

The following principles are used to design the tracing algorithm flow chart shown in Figure 5.

**4.2.1 Principle 1:** The tracing algorithm ends when the next vertex  $v'$  is the last vertex which is met by the tracing algorithm.

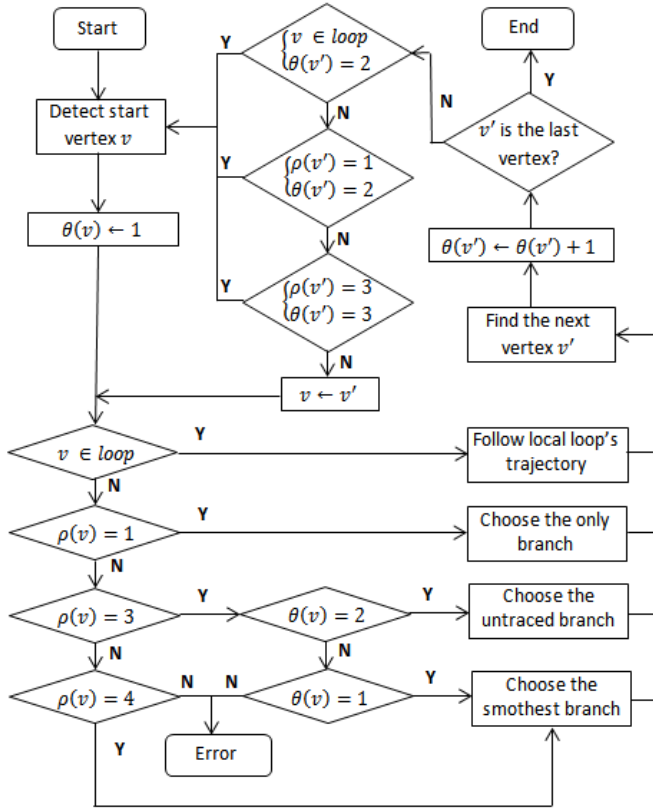


Figure 5. The handwriting's trajectory tracing algorithm.

**4.2.2 Principle 2:** The tracing of a single stroke ends if one of the three following conditions occurs: if an edge has been crossed 3 times or a vertex degree 3 has been met 3 times or a loop has been met twice. If one of these conditions occurs, the algorithm will detect the start vertex of the next stroke (search for the start vertex of all the vertices that have not been met before) and, then, continue to trace the stroke normally until Principle 1 is met.

**4.2.3 Principle 3:** If the current vertex  $v$  has the degree of 1, the output branch  $b_{out}$  is the same as the input branch  $b_{in}$ , as it is the only branch of the vertex.

$$\text{if } \rho(v) = 1 \text{ then } b_{out} = b_{in}.$$

**4.2.4 Principle 4:** If the current vertex has the degree of 3 and it has not been met before, the smoothest output branch is selected by the tracing algorithm

$$\text{if } \rho(v) = 3 \text{ and } \theta(v) = 1 \text{ then } b_{out} = b_{in}^{smooth}$$

where  $b_{in}^{smooth}$  is the smoothest output branch of the input branch  $b_{in}$ . Figure 6 illustrates a typical example of the smoothest branch choosing. In Figure 6(a), the scheme of detecting the smoothest output branch is represented.  $\phi_1$  and  $\phi_2$  are the angles of the input branch  $b_{in}$  with two output branches  $b_{out_1}$  and  $b_{out_2}$ . The smoother output branch will have the angle closer to  $\pi$ , or  $180^\circ$ . Figure 6(b) shows the example of labelling each output branch after the choosing process.

**4.2.5 Principle 5:** If the current vertex has degree of 3 and it has been met once, the remaining untraced branch is selected by the tracing algorithm

$$\text{if } \rho(v) = 3 \text{ and } \theta(v) = 2 \text{ then } b_{out} = b_{in}^{untraced}$$

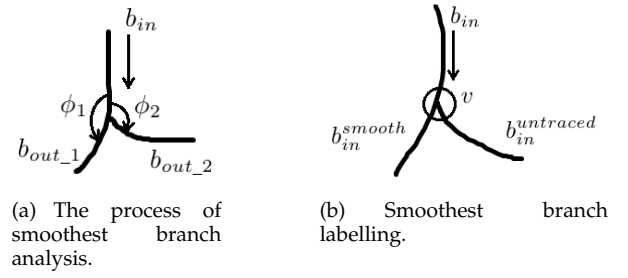


Figure 6. Smoothest branch analysis process.

where  $b_{in}^{untraced}$  is the only remaining untraced output branch of the input branch  $b_{in}$ . Figure 6(b) shows the labels of the output branches relative to the input branch. Specifically, in any vertex of degree 3, once an output branch is selected between two output candidates as a smoothest branch in the first meeting of the vertex, the remaining branch will automatically be the only untraced branch when the tracing algorithm reaches this vertex for the second time.

**4.2.6 Principle 6:** If the current vertex has the degree of 4, the smoothest output branch is selected by the tracing algorithm

$$\text{if } \rho(v) = 4 \text{ then } b_{out} = b_{in}^{smooth}$$

where  $b_{in}^{smooth}$  is the smoothest output branch of the input branch  $b_{in}$ .

**4.2.7 Principle 7:** If the current vertex is a member of a loop, the local trajectory of that loop is obtained and the output branch depends on the local trajectory of the loop.

One typical example of the trajectory tracing algorithm of handwritten letter 'h' in Figure 3 is briefly summarized in Table III. When the algorithm stops, we obtain the final global trajectory by concatenating the constituent trajectories of each step: Start  $\rightarrow b_1^1 \xrightarrow{e_1} b_4^1 \rightarrow b_4^3 \xrightarrow{e_2} b_4^4 \rightarrow b_4^2 \xrightarrow{e_3} b_5^3 \rightarrow b_5^1 \xrightarrow{e_4} b_2^1 \rightarrow b_2^2 \xrightarrow{e_4} b_5^1 \rightarrow b_5^2 \xrightarrow{e_5} b_3^1 \rightarrow$  End.

### 4.3 Local Loop Tracing

**4.3.1 Loop type 1:** This type of loop has 1 vertex and 2 adjacent edges, then one of these 2 edges is the input branch. Such a loop is written before reaching the output branch, which is the other vertex. This loop type is the most common generated one by the natural behavior of human. By this explanation, either of the two adjacent vertices is the input branch or the output branch, depending on the handwriting.

**4.3.2 Loop type 2:** Loop type 2 is one of the special cases of loops that have different ways of stroke-generation. Some examples of loop type 2 are shown in Table II. In this paper, we assume that human's behavior chooses the smoothest branch when the stroke meets any 3-degree-branch. The loop type 2 can be traced by complying these following steps:

- *Step 1:* Choose the smoothest output branch at the processing vertex (this vertex is part of the loop).
- *Step 2:* Follow the loop until it reaches the other vertex of the loop.



Table III  
A TYPICAL EXAMPLE OF THE TRAJECTORY TRACING ALGORITHM STEPS OF HANDWRITTEN LETTER 'H'.

Current vertex $v$	Degree $\rho(v)$	Input branch $b_{in}$	Comments	Output branch $b_{out}$	Next vertex $v'$	Meet times of next vertex $\theta(v')$	Next input branch $b'_{in}$	Trajectory = $b_{in} \rightarrow b_{out} \xrightarrow{edge} b'_{in}$
$v_1$	1		Start vertex of degree 1. Choose the only branch (Principle 3)	$b_1^1$	$v_4$	$0 \rightarrow 1$	$b_4^1$	Start $\rightarrow b_1^1 \xrightarrow{e_1} b_4^1$
$v_4$	loop	$b_4^1$	$v_4$ belongs to a loop with input branch $b_4^1$ . Follow the local loop's trajectory before going to the output branch $b_4^2$ (Principle 7)	$b_4^2$	$v_5$	$0 \rightarrow 1$	$b_5^3$	$b_4^1 \rightarrow b_4^2 \xrightarrow{e_2} b_4^3 \rightarrow b_4^4 \xrightarrow{e_3} b_5^3$
$v_5$	3	$b_5^3$	Vertex $v_5$ (degree 3) has not been met before, so choose the smoothest output branch $b_5^1$ relative to its input branch $b_5^3$ (Principle 4)	$b_5^1$	$v_2$	$0 \rightarrow 1$	$b_2^1$	$b_5^3 \rightarrow b_5^1 \xrightarrow{e_4} b_2^1$
$v_2$	1	$b_2^1$	$\rho(v_2) = 1$ so choose the only branch (Principle 3)	$b_2^1$	$v_5$	$1 \rightarrow 2$	$b_5^1$	$b_2^1 \rightarrow b_2^2 \xrightarrow{e_4} b_5^1$
$v_5$	3	$b_5^1$	Vertex $v_5$ (degree 3) has been met once before, so choose the remaining uncrossed branch $b_5^2$ (Principle 5)	$b_5^2$	$v_3$	$0 \rightarrow 1$	$b_3^1$	$b_5^1 \rightarrow b_5^2 \xrightarrow{e_5} b_3^1$
$v_3$	1	$b_3^1$	Algorithm stops because vertex $v_3$ is the last processed vertex of the skeleton (Principle 1)					$\rightarrow$ End

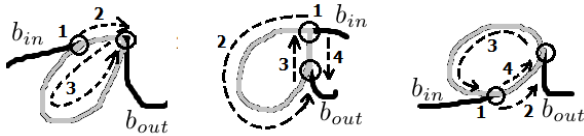


Figure 7. The tracing steps of loop type 2.

- Step 3: Trace the loop's circle in the higher direction until all the pixels of the circle are met.
- Step 4: Find the shortest edge that leads to the output branch.

The tracing steps of loop type 2 are depicted in Figure 7. The loop circle has the gray color and two adjacent branches are black, which are labeled  $b_{in}$  and  $b_{out}$  corresponding to the input branch and output branch, respectively. Two vertices of the loop are marked by two thin black circles. The dashed arrow represents the trajectory and the numbers on the arrows indicate the step number of the tracing algorithm.

## 5 EXPERIMENT AND DISCUSSION

To test the tracing algorithm performance, we conduct the experiment on four sets of handwritings. The first and the second sets are respectively composed of 100 single-stroke and 100 multi-stroke English character images created from the Unipen database [16]. This database contains a huge number of online samples with different styles like digits, upper/lower case letters, symbols, printed words and texts. These samples have the sequence of pen-tip coordinates. To retrieve the static images from this online database, we assume that the stroke width is 3 pixels. The noisy artifacts are removed from the extracted scripts. For a more thorough analysis on the complicated handwritings rather than English characters, we also collect handwritings from

Table IV  
EXPERIMENTAL RESULTS OF OUR APPROACH

Set	Settings	Accuracy
1	UNIPEN corpus, 100 single-stroke characters	96.0%
2	UNIPEN corpus, 100 multi-stroke characters	94.0%
3	100 single-stroke general handwritings	95.0%
4	100 multi-stroke general handwritings	74.0%

Table V  
EXPERIMENTAL RESULTS OF PREVIOUS METHODS

Author	Settings	Accuracy
Abuhaiba <i>et al.</i> [17]	2 writers, 65 strokes, 159 blobs	83.6%
Allen and Navarro [18]	1248 characters by 12 writers	91.6%
Plamondon <i>et al.</i> [8]	200 city names	89.0%
Lallican <i>et al.</i> [19]	260 characters by 10 writers	90.0%
Kato and Yasuhara [9]	100 single-stroke handwritings	91.6%
Doermann <i>et al.</i> [12]	1270 words by 5 writers	84.0%
Rousseau <i>et al.</i> [20]	5800 single-stroke characters	87.0%
Qiao and Yasuhara [21]	UNIPEN corpus, words contain vertices with degree 3 or degree 4 only	93.7%
Qiao <i>et al.</i> [5]	UNIPEN corpus, 708,881 static images & 187 single-stroke images	96.0%

different sources for the third and the fourth set. They are 100 single and multi-stroke, respectively, handwritings including English words, human signatures and artificial shapes. The recovery accuracy of these four sets is shown in Table IV. As a result, 96 samples from the first set and 94 samples from the second set are successfully recovered, which result in the accuracy of 96.0% and 94.0%, respectively, for the experimental subjects of English characters. In comparison with others' results shown in Table V, it can be noted that our method has the promising outcome for the recovery of English characters, compared to the methods of Allen and Navarro [18], Lallican *et al.* [19], Rousseau *et al.* [20],



Figure 8. The successfully recovered handwritings.

which have the average accuracy of 91.6%, 90.0% and 87.0% respectively.

We also make some comparisons with others' approaches which deal with general handwritings, including the methods of Abuhaiba *et al.* [17] (83.6%), Plamondon *et al.* [8] (89.0%), Kato and Yasuhara [9] (91.6%), Doermann *et al.* [12] (84.0%), Qiao *et al.* [21] [5] (93.7%, 96.0%). We obtain the accuracy of 95.0% for single-stroke general handwritings from set 3. Our result shows that the accuracy of our method is higher than that of the other approaches for single-stroke handwritings. Finally, for a general multi-stroke scripts, we also achieve the accuracy of 74.0%, approximate to three quarters of general scripts. There are specific technical reasons to explain the degradation of accuracy when working with multi-stroke scripts. We collect multi-stroke scripts from different sources to build the fourth experimental set, including some complicated multi-stroke handwritings which are hard to recover its correct trajectory. Most of the errors occur in the phase of detecting the start point of each sub-stroke, mainly because the sub-strokes are overlapped and the tracing algorithm cannot detect the start point of each sub-stroke. Some other mismatch cases are the wrong detection of stroke order, or wrong tracing direction due to smoothest output branch misidentification on the intersection. Several mis-detected samples are shown and explained in Figure 9.

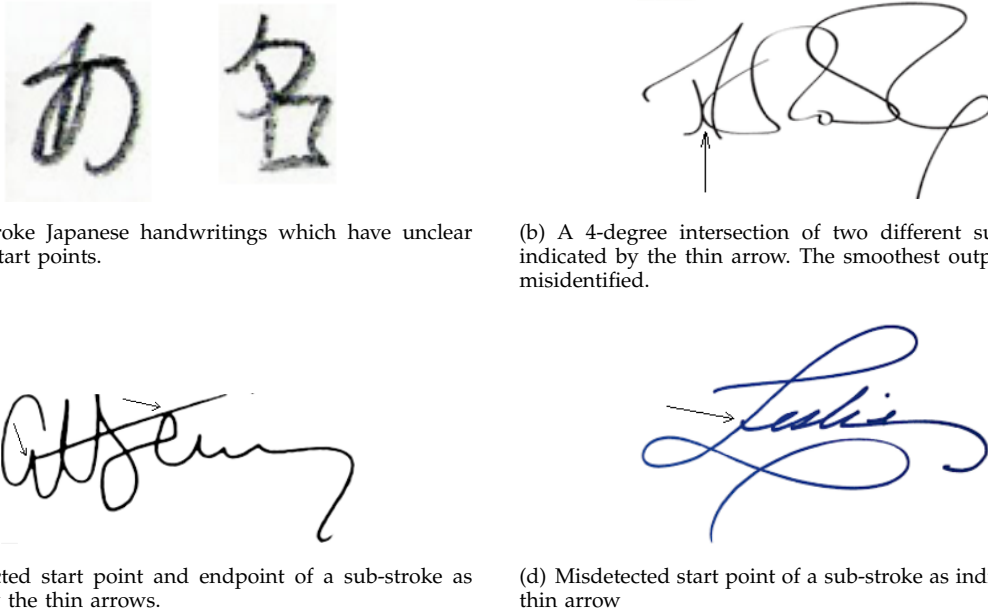
The experimental results show that our approach can recover the stroke of static handwritings, including single-stroke and multi-stroke scripts. Based on the idea of loop investigation, our proposed method is capable of recovering complicated strokes which include loops and intersections. The tracing algorithm is mostly based

on human natural writing behavior. The scripts must satisfy the pre-defined assumptions to be recovered. When a loop is being processed to extract its local trajectory, its vertices and adjacent edges are used for the analysis process. The missing detected strokes are most due to the loop's circle smoothness, which are not specific skeletal characteristics presented in this paper. Then loop's smoothness analysis must be taken into consideration when performing the task of tracing the loop's circle. One other proposal that should be highlighted in future studies is the categorization and recovery of local loops. There should be noted that this paper does not cover all the possibilities of generated loops. Different number of vertices, edges and adjacent edges of a loop, and even overlapped loops can be analyzed for better performance.

## 6 CONCLUSION

This paper has presented a method to recover the handwritten stroke of static handwriting images based on the novel approach on loop analysis. The recovery capability is based on skeletal objects (edges, vertices and loops) and can be applied to single-stroke as well as multi-stroke handwritings. The tracing algorithm only ends if all the vertices of the handwriting's skeleton are traced. This method therefore can detect all the pen-down and pen-up events, corresponding to the single strokes which construct the multi-stroke image. Loop analysis is performed before the global stroke construction phase as a separate process. This lays a foundation for a new model of handwriting's skeleton, which treats a loop as an individual object rather than the combination of vertices and edges. Evaluating all the





(a) Multi-stroke Japanese handwritings which have unclear sub-stroke start points.

(b) A 4-degree intersection of two different sub-strokes, as indicated by the thin arrow. The smoothest output branch is misidentified.

(c) Misdetected start point and endpoint of a sub-stroke as indicated by the thin arrows.

(d) Misdetected start point of a sub-stroke as indicated by the thin arrow

Figure 9. Unsuccessful recovered trajectory handwritten samples

combination possibilities of loops can help to improve the recovery performance significantly. Further study will focus on the heuristic rules of written loop, which varies in different languages. A more intricate analysis on loops, especially on overlapped loops should be noticed in our work in future.

#### ACKNOWLEDGMENT

This research is funded by Ho Chi Minh city University of Technology, VNU-HCM under grant number TNCS-DDT-2015-21.

#### REFERENCES

- [1] R. Mandal, P. Roy, U. Pal, and M. Blumenstein, "Signature segmentation and recognition from scanned documents," in *13th International Conference on Intelligent Systems Design and Applications (ISDA)*, Dec 2013, pp. 80–85.
- [2] H. Suwanwiwat, V. Nguyen, M. Blumenstein, and U. Pal, "Off-line handwritten Thai name recognition for student identification in an automated assessment system," in *International Joint Conference on Neural Networks (IJCNN)*, July 2014, pp. 2347–2353.
- [3] R. Kunwar, U. Pal, and M. Blumenstein, "Semi-supervised Online Bayesian Network Learner for Handwritten Characters Recognition," in *22nd International Conference on Pattern Recognition (ICPR)*, Aug 2014, pp. 3104–3109.
- [4] H. Suwanwiwat, V. Nguyen, M. Blumenstein, and U. Pal, "Off-Line Handwritten Bilingual Name Recognition for Student Identification in an Automated Assessment System," in *14th International Conference on Frontiers in Handwriting Recognition (ICFHR)*, Sept 2014, pp. 271–276.
- [5] Y. Qiao, M. Nishiara, and M. Yasuhara, "A Framework Toward Restoration of Writing Order from Single-Stroke Handwriting Image," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 28, no. 11, pp. 1724–1737, Nov 2006.
- [6] T. Huang and M. Yasuhara, "A total stroke SLALOM method for searching for the optimal drawing order of off-line handwriting," in *Systems, Man and Cybernetics, 1995. IEEE International Conference on Intelligent Systems for the 21st Century.*, vol. 3, Oct 1995, pp. 2789–2794 vol.3.
- [7] R. Plamondon and S. Srihari, "Online and off-line handwriting recognition: a comprehensive survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 1, pp. 63–84, Jan 2000.
- [8] R. Plamondon and C. Privitera, "The segmentation of cursive handwriting: an approach based on off-line recovery of the motor-temporal information," *IEEE Transactions on Image Processing*, vol. 8, no. 1, pp. 80–91, Jan 1999.
- [9] Y. Kato and M. Yasuhara, "Recovery of drawing order from single-stroke handwriting images," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 9, pp. 938–949, Sep 2000.
- [10] T. Steinherz *et al.*, "Offline Loop Investigation for Handwriting Analysis," in *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 2, 2009, pp. 193–209.
- [11] E. L'Homer, "Extraction of strokes in handwritten characters," *Pattern Recognition*, vol. 33, no. 7, pp. 1147–1160, 2000.
- [12] D. Doermann, N. Intrator, E. Rivin, and T. Steinherz, "Hidden loop recovery for handwriting recognition," in *Eighth International Workshop on Frontiers in Handwriting Recognition, 2002*, 2002, pp. 375–380.
- [13] T. Pavlidis, *Algorithms for Graphics and Image Processing*. Springer, 1982.
- [14] R. Ogniewicz and M. Ilg, "Voronoi skeletons: theory and applications," in *IEEE Proceedings on Computer Vision and Pattern Recognition*, Jun 1992, pp. 63–69.
- [15] V. A. Kha and H. H. Kha, "Extraction of dynamic features from static handwritten image and recognition using Hidden Markov model," *IEEE Electron Device Letters*, 2015.
- [16] I. Guyon, L. Schomaker, R. Plamondon, M. Liberman, and S. Janet, "UNIPEN project of on-line data exchange and recognizer benchmarks," in *Pattern Recognition*, vol. 2, Oct 1994, pp. 29–33 vol.2.
- [17] I. Abuhaiba, S. Datta, and M. Holt, "Processing of text documents: straight line approximation and lost loop recovery," in *Proceedings of the Third International Conference*

on *Document Analysis and Recognition*, vol. 2, Aug 1995, pp. 1157–1160 vol.2.

- [18] C. Allen and A. Navarro, "The recognition of Roman hand-written characters using dynamic information recovery," in *Sixth International Conference on Image Processing and Its Applications*, vol. 2, Jul 1997, pp. 741–745 vol.2.
- [19] P. Lallican and C. Viard-Gaudin, "A Kalman approach for stroke order recovering from off-line handwriting," in *Proceedings of the Fourth International Conference on Document Analysis and Recognition*, vol. 2, Aug 1997, pp. 519–522 vol.2.
- [20] L. Rousseau, E. Anquetil, and J. Camillerapp, "Recovery of a drawing order from off-line isolated letters dedicated to on-line recognition," in *Proceedings of Eighth International Conference on Document Analysis and Recognition*, Aug 2005, pp. 1121–1125 Vol. 2.
- [21] Y. Qiao and M. Yasuhara, "Recovering Drawing Order from Offline Handwritten Image Using Direction Context and Optimal Euler Path," in *IEEE International Conference on Acoustics, Speech and Signal Processing*, vol. 2, May 2006, pp. II–II.



**Vo Anh Kha** was born in Ho Chi Minh City, Vietnam. He received his B.Eng. and M.Eng. degrees from the French excellent engineering training program named Programme de Formation d'Ingenieurs d'Excellence au Vietnam, Ho Chi Minh City University of Technology, Vietnam in 2014 and 2015, respectively. Currently he is pursuing his Ph.D study at the Griffith University, Australia. His research interests are Pattern Recognition, Electroencephalography and Digital Signal Processing.



**Ha Hoang Kha** was born in Dong Thap, Vietnam. He received the B.Eng. and M.Eng. degrees from Ho Chi Minh City University of Technology, in 2000 and 2003, respectively, and the Ph.D. degree from the University of New South Wales, Sydney, Australia, in 2009, all in Electrical Engineering and Telecommunications. From 2000 to 2004, he was a research and teaching assistant with the Department of Electrical and Electronics Engineering, Ho Chi Minh City University of Technology. He was a visiting research fellow at the School of Electrical Engineering and Telecommunications, the University of New South Wales, Australia, from 2009 to 2011. He was a postdoctoral research fellow at the Faculty of Engineering and Information Technology, University of Technology Sydney, Australia from 2011 to 2013. He is currently a lecturer at the Faculty of Electrical and Electronics Engineering, Ho Chi Minh City University of Technology, Vietnam. His research interests are in digital signal processing and wireless communications, with a recent emphasis on convex optimization techniques in signal processing for wireless communications.



**Michael Blumenstein** is a Professor in the School of Information and Communication Technology at Griffith University, where he previously served as the Dean (Research) in the Science, Environment, Engineering and Technology Group, and prior to this was the Head of the School of Information and Communication Technology. Michael is a nationally and internationally recognised expert in the areas of automated Pattern Recognition and Artificial Intelligence, and his current research interests include Multi-Script Handwriting Recognition and Signature Verification. He has published over 120 papers in refereed books, conferences and journals. His research also spans various projects applying Artificial Intelligence to the fields of Engineering, Environmental Science, Neurobiology and Coastal Management. In 2009 Michael was named as one of Australia's Top 10 Emerging Leaders in Innovation in the Australian Top 100 Emerging Leaders Series supported by Microsoft. Michael is a Fellow of the Australian Computer Society and a Senior Member of the IEEE.