H-053

Recovering Drawing Order from Static Handwritten Images Using Probabilistic Tabu Search

Takayuki Nagoya[†] **Hiroyuki Fujioka**[‡]

Abstract

This paper considers the problem of recovering a drawing order from static handwriting images with single stroke. The problem is analyzed and solved by employing the so-called graph theoretic approach. The central issue is to obtain the smoothest path of stroke from a graph model of input handwriting image. We first introduce an index on double-traced lines. Then, it is shown that the graph is transformed to a semi-Eulerian graph. Thus, the restoration problem reduces to maximum weight matching problem. Such a problem is solved by a probabilistic tabu search.

1. Introduction

Recovering the drawing order from static handwritten images is a key problem in the off-line recognition, and thus it has been studied extensively. For such problems, the graph theory approach has been used frequently and various types of recovery algorithm have been developed (see e.g. [1, 2]). In particular, Kato and Yasuhara in [3] have developed the hybrid approach of local tracing and global searching methods. In the method, we first determines the the types of each edge and vertex by locally analyzing the structure of the graph model, we get the labeling information of stroke. Tracing the graph according to the labeling information, we obtain the dynamic information of stroke. Authors in [4] have recently improved Kato and Yasuhara's method and the faster recovery algorithm has been developed. In these hybrid approaches, using only the local structure of graph, the drawing order are determined at each vertex. Thus, the obtained stroke may not correspond to the smoothest path of the possible ones.

In this paper, we develop a new method for recovering a drawing order from static handwriting images with single stroke. Such a stroke may include the so-called double-traced lines (D-lines). The problem is analyzed and solved by employing the graph theoretic approach [4]. Then the central issue is to obtain the smoothest path of stroke from a graph model of input handwriting image. First, the graph model is constructed from the input handwriting image by employing thinning algorithm. Then, we locally analyze the structure of graph at each vertex. In particular, the method to identify D-lines is developed by introducing an index on D-lines. The method enables us to transform any graph models including D-lines to semi-Eulerian graph models. Then, the restoration problem reduces to maximum weight perfect matching problem of graph, thus a probabilistic tabu search algorithm is developed to solve the problem. The effectiveness and usefulness are examined by some experimental studies.

In the sequel, we here assume that an undirected graph G is constructed from the handwritten image I by skeltonization techniques. Each edge of G corressponds to an part of stroke in the skeleton and each vertex of G corresponds to a geometrical feature point at which an edge terminates and multi-edge is connected.

2. Identification of Double-Traced Lines

Now, suppose that a graph G with vertex set $V(G) = \{v_1, v_2, \cdots, v_n\}$ and edge set $E(G) = \{e_1, e_2, \cdots, e_m\}$ is modeled from a handwritten image input I. If the graph G is a semi-Eulerian, there exists a walk on G which traverses all of the edges exactly once from a vertex with degree one (i.e. start vertex) to another one (i.e. end vertex). Such a walk is referred as 'Euler path'. It is well known that the problem of finding a Euler path in a semi-Eulerian graph can be computed in polynomial time. In addition, we may often face the situations in which a stroke is drawn twice. Such a stroke which is traversed twice is called as 'double-traced line (D-line)'. In such cases, the graph G obtained from input I never become semi-Eulerian. Then, we need to detect all the D-lines in order to construct a semi-Eulerian graph from G.

Fig. 1 shows a set of possible D-lines in the stroke. Here, we see that the all the types of D-lines except to Fig. 1 (a), say Fig. 1 (b)-(f), are represented as the graph with same structure. Thus, it may be difficult to distinguish the differences among their D-lines from only local structure information of G. To avoid such a difficulties, we here introduce an index of D-lines (Section 2.1). The index is

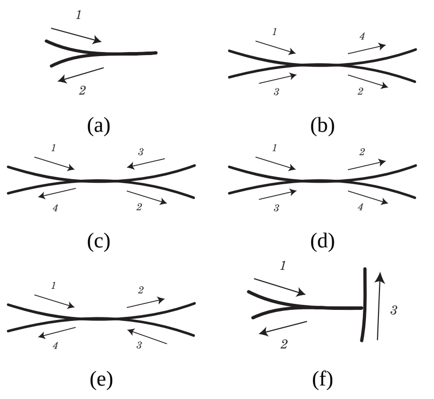


Fig. 1: Six types of possible d-lines.

[†]Department of Information Systems, Tottori University of Environmental Studies

[‡]Department of System Management Fukuoka Institute of Technology

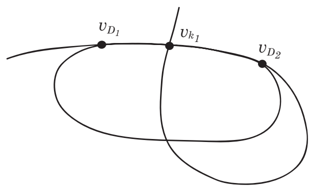


Fig. 2: D-line with $P_D = \langle (v_{D_1}, v_{k_1}), (v_{k_1}, v_{D_2}) \rangle$.

used to locally identify all the D-lines. We then develop the method to transform graph G with D-lines into semi-Eulerian one by employing the so-called path duplication method (Section 2.2). Meanwhile, the differentiation on the types of D-line will be achieved by global analysis using a continuity graph and probabilistic tabu search algorithm in later section.

2.1. Index on D-lines

We present the index of D-lines. All the types of D-lines in Fig. 1 consist of an edge between two vertices with odd degree, where the length of edge is generally small. In addition, the stroke may intersect with D-line as shown in Fig. 2. In such a case, each D-line is represented as a path P_D of G with length more than or equal to 1. When the D-line is given as a path between two vertices $v_{D_1}, v_{D_2} \in V(G)$, the path P_D is expressed as

$$P_D = \langle (v_{D_1}, v_{k_1}), (v_{k_1}, v_{k_2}), \cdots, (v_{k_l}, v_{D_2}) \rangle \tag{1}$$

with $v_{k_i} \in V(G), i = 1, 2, \cdots, l$, where (v_p, v_q) denotes an unordered pair of adjacent vertices v_p and v_q, hence an edge between v_p and v_q.

In order to determine whether a path P_D is D-line, we introduce the following index of D-lines. The index is based on the straightness and smoothness between the path P_D and the incident edges to P_D. Let $DLI(P_D) \in \mathbf{R}^+$ be index of D-lines for the path P_D defined by

$$DLI(P_D) = ST(P_D) + \lambda SM(P_D), \tag{2}$$

where $\lambda \in \mathbf{R}^+$ is a weighted parameter, and $ST(P_D)$ and $SM(P_D)$ denote the ratios on straightness and smoothness for P_D defined as follows.

First, the straightness ratio $ST(P_D)$ is evaluated by

$$ST(P_D) = \frac{d(P_D)}{l(P_D)}, \tag{3}$$

where $d(P_D)$ is Euclidean distance between two terminals v_{D_1} and v_{D_2} in (1) and $l(P_D)$ is the total number of pixels on P_D. It then holds that $ST(P_D) \leq 1$ with upper bound being achieved when the path becomes exactly straight line. As the straightness of path P_D decreases, $ST(P_D)$ approaches to 0.

Next, the smoothness ratio $SM(P_D)$ in (2) is generally defined as

$$SM(P_D) = \min\left\{ \sum_{i=1,2} S\left(e_D, e_i^{v_{D_1}}\right), \sum_{i=1,2} S\left(e_D, e_i^{v_{D_2}}\right) \right\} \tag{4}$$

with

$$S(e_D, e_i^{v_{D_j}}) = \begin{cases} \frac{|\alpha|}{\pi} & \text{if } \alpha \neq 0 \\ 1 & \text{if } \alpha = 0 \end{cases}, \; i, j = 1, 2. \tag{5}$$

Here, $e_i^{v_{D_j}}$ denotes an edge $(v_{D_j}, v_i^{D_j})$ for $i, j = 1, 2$, where $v_i^{D_j}$ is an adjacent vertex to v_{D_j}. e_D denotes the edge that is obtained by merging all edges of P_D to one. $|\alpha| \in [0, \pi]$ is a difference angle between the edges e_D and $e_i^{v_{D_j}}$. It then holds that $S(e_D, e_i^{v_{D_j}})$ approaches to 1 as $|\alpha|$ approaches to π. Also, when $\alpha = 0$, we set $S(e_D, e_i^{v_{D_j}}) = 1$. The case of $\alpha = 0$ may not usually occur, but this expression may be used for constructing a continuity graph in Section 3.1.

Remark 1. *If the path P_D is D-line in Fig. 1 (a), the degree for either of v_{D_i}, $i = 1, 2$ may be 1, i.e.* $\deg(v_{D_i}) = 1$. *Then, there exists no incident vertices for such a vertex v_{D_i} with* $\deg(v_{D_i}) = 1$. *Thus, if* $\deg(v_{D_i}) = 1$ *on P_D satisfies for either $i = 1$ or $i = 2$, we have only to evaluate $SM(P_D)$ in (4) as*

$$SM(P_D) = \sum_{i=1,2} S\left(e_D, e_i^{v_{D_j}}\right) \qquad \text{if } \deg(v_{D_k}) = 1 \tag{6}$$

for $j, k \in \{1, 2\}$, $j \neq k$.

2.2. Constructing Semi-Eulerian Graph

We are now in the position to develop a method to transform the graph G with D-lines to a semi-Eulerian graph. Then, main issue is to identify D-lines by the index of D-lines and remove the all the odd-degree vertices.

When there exist some D-lines on the stroke, the odd-degree vertices may consist of those of D-line or terminal points of stroke (i.e. the start and end points) in G. Now, let $V_1 \subseteq V(G)$ be a set of vertex with degree one. Then each vertex $v_i \in V_1$ is either a terminal point of stroke or a degree one vertex of a D-line as shown in Fig. 1 (a). In other words, there exist exactly two vertices, denoted by $v_s, v_t \in V_1$, which correspond to start and end points of stroke. We then identify v_s and v_t by using DLI as follows. For each $v_i \in V_1$, we find the nearest vertex $v_j \in V(G) (i \neq j)$ from v_i whose degree is odd greater than or equal to three. Let $P(v_i, v_j)$ be the shortest path in G between v_i and v_j. Then, the vertex v_i with the smallest $DLI(P(v_i, v_j))$ may not be a terminal of D-line. Hence, two vertices with smallest DLI may be regarded as the terminal vertices v_s and v_t.

Next, let V_D be the set of every odd degree vertices of G, where $v_s, v_t \in V_1$ are omitted. Then, it is easy to see that $|V_D|$ is even. In addition, any D-lines are the shortest path between a pair of vertices of V_D, because the path length corresponding to D-line must be very short. According to the fact that any D-line does not connect two vertices both of which have degree one, all the D-lines are identified as follows. Let G_{odd} be an edge weighted graph with a set of vertex $V(G_{odd})$,

$$V(G_{odd}) = V_D \tag{7}$$

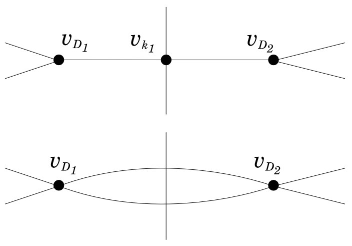


Fig. 3: The upper figure is an example of a path $P_D = \langle (v_{D_1}, v_{k_1}), (v_{k_1}, v_{D_2}) \rangle$ that represents D-line. The lower figure is the result of the path duplication method.

and a set of edge $E(G_{odd})$,

$$E(G_{odd}) = (V_D \times V_D) \backslash \{(v_i, v_j) | \deg(v_i) = \deg(v_j) = 1\}, \tag{8}$$

where $A \times A$ is the Cartesian product of A with itself. Also, we introduce the weight function $w : E(G_{odd}) \to \mathbf{R}^+$ defined as

$$w(v_i, v_j) = DLI(P(v_i, v_j)) \tag{9}$$

for each $(v_i, v_j) \in E(G_{odd})$. Then, the detection of D-lines can be regarded as the maximum weighted matching problem on G_{odd}. As is well known, the maximum weighted matching problem can be solved in polynomial time. Letting M be a maximum weighted matching of G_{odd}, we identify any path P_D corresponding to an edge of M as D-line.

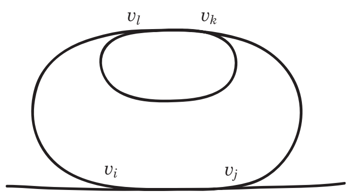
After identifying the terminal vertices and D-lines, we readily transform the graph G into a semi-Eulerian graph G_{eul} by employing the path duplication method as follows. Now, we suppose that a D-line is given as $P_D = \langle (v_{D_1}, v_{k_1}), (v_{k_1}, v_{k_2}), \cdots, (v_{k_l}, v_{D_2}) \rangle$. Then, all the edges in P_D are removed and v_{D_1} is connected with v_{D_2} by two new edges. As a result, the degree of vertex $v_{k_i}, i = 1, 2, \ldots, l$ may reduced to two. Since there is no branch of stroke at such vertex v_{k_i}, we can remove v_{k_i} by merging two edges adjacent to v_{k_i} to an edge (see the Fig. 3). Hence, we get the following lemma.

Lemma 1. *G_{eul} is a semi-Eulerian graph. Furthermore, Euler paths of G_{eul} corresponds one-to-one with strokes of the given image.*

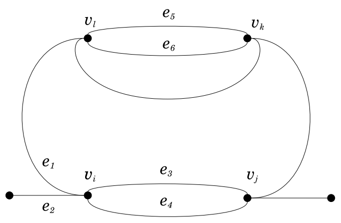
Example 1. *When a handwritten image as shown in Fig. 4 (a) is given as input image I, we get semi-Eulerian graph G_{eul} as shown in Fig. 4 (b).*

3. Constructing Optimal Euler Path

Our concern is to recovery the smoothest drawing order, which is normally produced by humans, from input image I. Then, our task is to find the smoothest Euler path from the semi-Eulerian graph G_{eul}. We first construct continuity graph from G_{eul}. Then, the smoothest Euler path is approximated by employing probabilistic tabu search algorithm.



(a) handwritten image input I



(b) corresponding semi-Eulerian graph G_{eul}

Fig. 4: An example of handwritten image input I and the corresponding semi-Eulerian graph G_{eul}.

3.1. Constructing Continuity Graph

We construct the continuity graph G_{con} from the semi-Eulerian path G_{eul} in Section 2.

Let v_i be even degree vertex of G_{eul} and let $\{e_1, \ldots, e_d\}$ be a set of adjacent edges of v_i. For each v_i, we create a complete graph C_{v_i} whose vertices are $\{v_{e_1}, \ldots, v_{e_d}\}$. Then, v_i is replaced by C_{v_i} and we connect v_{e_i} to e_i for $i = 1, \ldots, d$. In addition, the weight $S(e_i, e_j)$ is assigned for each edge (v_{e_i}, v_{e_j}). Note that the graph G_{con} is no longer Eulerian. However, by using a perfect matching of C_{v_i}, we can readily reduce the graph G_{con} to a path corresponding to an Euler path of G_{eul} as follows. Let M_p be a set of perfect matchings defined as

$$M_p = \{M_{v_i}\}_{v_i \in V(G_{eul})}, \tag{10}$$

where $M_{v_i} \in M_p$ is a perfect matching of C_{v_i}. Then, the graph G'_{con} defined by

$$G'_{con} = \left(V(G_{con}), \left(E(G_{con}) - \bigcup_{v \in V(G_{eul})} E(C_v) \right) \cup M_p \right) \tag{11}$$

is a path that correspond to an Euler path of G_{eul} whenever G'_{con} is connected. We then get the following theorem.

Theorem 1. *For a set of perfect matchings M_p in (10), G'_{con} is a path corresponding to an Euler path of G_{eul} whenever G'_{con} is connected. Conversely, for every Euler path P of G_{eul}, there exists a perfect matching M_p such that G'_{con} corresponds to P.*

Theorem 1 indicates that the problem of computing optimum Eulerian path in G_{eul} is reduced to the problem of computing a set of perfect matching $M_p = \{M_{v_i}\}_{v_i \in V(G_{eul})}$, where M_{v_i} is a perfect matching of C_{v_i} such that G'_{con} is connected. Furthermore, the total weight of M_p represents

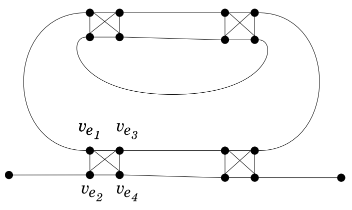


Fig. 5: A continuity graph G_{con} constructed from the semi-Eulerian graph G_{eul} in Fig. 4 (b).

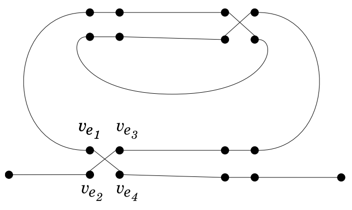


Fig. 6: The resulting graph G'_{con} obtained from the continuity graph G_{con} in Fig. 5.

smoothness of a corresponding Euler path. In order to compute such M_p, a probabilistic tabu search algorithm may be available.

Example 2. *The continuity graph G_{con} as shown in Fig. 5 is constructed from semi-Eulerian graph G_{eul} in Fig. 4 (b).*

3.2. Probabilistic Tabu Search

We develop a probabilistic tabu search in order to obtain the maximum weighted perfect matchings M_p such that G'_{con} is connected.

For this purpose, we first compute an Euler path P_e of G_{eul}. Then, a set of perfect matchings of complete graph $\{C_{v_i}\}_{v_i \in V(G_{eul})}$ corresponding to P_e is given as M_p in (10). Here, M_p is set as initial feasible solution of probabilistic tabu search. In the probabilistic tabu search, an optimal solution is obtained by iteratively modifying the initial feasible solution to better one. Then, a tabu list T of feasible solutions is used to avoid modifying to feasible solution that have been visited in the recent past. An initial setting of T is set as $T = \{M_p\}$. Then, the algorithm may find better feasible solution M'_p of which the weight is larger than that of previously visited feasible solutions by repeating the following local change: First, we choose C_{v_i} randomly from $\{C_{v_i}\}_{v_i \in V(G_{eul})}$ with probability proportional to the weight of $M_{v_i} \in M_p$. Then, two edges $e_1 = (v_1, v_2)$ and $e_2 = (v_3, v_4)$ of M_{v_i} are chosen uniformly at random. In addition, we set $E_1 = \{(v_1, v_2), (v_3, v_4)\}, E_2 = \{(v_1, v_3), (v_2, v_4)\}$, and $E_3 = \{(v_1, v_4), (v_2, v_3)\}$. If there exists $i \in \{2, 3\}$ such that $w(E_i) \leq w(E_1)$ and $M'_p = M_p \backslash E_1 \cup E_i$ is a feasible solution unlisted in T, then we update T and M_p as $T = T \cup \{M'_p\}$ and $M_p = M'_p$ respectively. This process is iteratively carried out for new M_p and T until a predefined time limit N is exceeded. Finally, the algorithm outputs the M_p that is the best feasible solution found so far as the desired drawing order of input handwritten image I.

Example 3. *We get the graph G'_{con} in Fig. 6 from the continuity graph G_{con} in Fig. 5.*

4. Experiments

We perform some experiments by employing our algorithm. Fig. 7 shows the recovering results. In these figures, the written order is illustrated by arrow marks. From the results, we may observe that our algorithm can correctly recover the drawing order of handwritten image even when the double-traced lines are included.

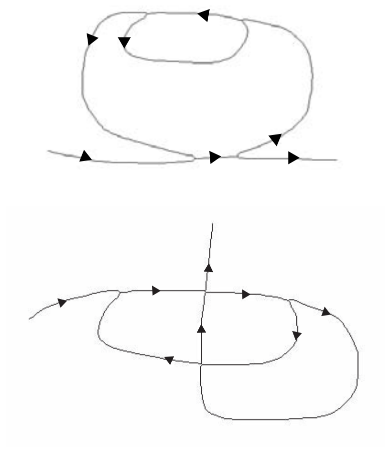


Fig. 7: Experimental results.

References

[1] S. Lee and J. C. Pan, Offline Tracing and Representation of Signatures, *IEEE Trans. Systems, Man, and Cybernetics*, Vol.22, No.4, pp.755–771, 1992.

[2] S. Jäger, Recovering Writing Traces in Off-Line Handwriting Recognition: Using a Global Optimization Technique, *Proc. of 13th Int. Conf. on Pattern Recognition*, pp. 931-935, Vienna, Aug.25–29, 1996.

[3] Y. Kato and M. Yasuhara, Recovery of drawing order from single-stroke handwriting images, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol.22, No.9, pp.938–949, 2000.

[4] H. Fujioka and T. Nagoya, Recovering Stroke Order from Multi-Stroke Character Images, *Proc. of the 2011 2nd Int. Conf. on Innovative Computing and Communication, and 2011 2nd Asia-Pacific Conference on Information Technology and Ocean Engineering*, pp.34-37, Macao, March 5-6, 2011.