Recovery of temporal information of cursively handwritten words for on-line recognition

H. BUNKE, R. AMMANN,
G. KAUFMANN, T. M. HA

University of Berne
CH-3012 Switzerland
{bunke,kaufmann,haminh}@iam.unibe.ch

M. SCHENKEL, R. SEILER
and F. EGGIMANN

Swiss Federal Institute of Technology
Zurich, Switzerland
{schenkel,seiler,eggimann}@isi.ee.ethz.ch

Abstract

On-line recognition differs from off-line recognition in that additional information about the drawing order of the strokes is available. This temporal information makes it easier to recognize handwritten texts with an on-line recognition system. In this paper we present a method for the recovery of the stroke order from static handwritten images. The algorithm was tested by classifying the words of an off-line database with a state-of-the-art on-line recognition system. On this database with 150 different words, written by four cooperative writers, a recognition rate of 97.4% was obtained.

1 Introduction

In the past, on-line and off-line character recognition technologies had evolved independently. On-line data are essentially temporal whereas off-line data are spatial. This fundamental difference led therefore to separate developments. Recently however, it has been realized that the two technologies can cooperate and eventually be combined to compensate the weaknesses of each other. The general idea is to convert the data from one representation to the other and apply the two technologies to two representations of the same data. The results can then be combined to improve the reliability.

The conversion from on-line to off-line representation is a relatively simple process (see [8, 10], for instance). Therefore the off-line techniques are almost immediately available for on-line recognition. The reverse is much more complicated. Converting off-line to on-line representation is a kind of inverse problem, which is known to be difficult and not always possible. Nevertheless, some attempts have been made [6, 2, 7].

In this paper, we present our studies of converting off-line to on-line representation. Our works consist of two parts, namely, the conversion algorithm (Section 2) and the testing of the converted off-line data by an on-line system (Section 3).

2 Algorithm

The main idea behind our reconstruction algorithm is to formulate it as a graph search problem. Singular points of the word, e.g. end points and T-joints, are associated to the nodes of a graph. Strokes connecting singular points are assigned to the corresponding edges. The reconstruction is defined as the process of building a sequence of strokes, i.e. choosing a path of the graph, which should correspond to the (natural) usual writing order. Accordingly, we have to solve the following three problems: construct the graph representation of the word (Section 2.1), define the likelihood criteria (Section 2.2), and search the best path in the graph (Section 2.3).

2.1 Graph representation of the word

Since the graph representation of the word is an intermediary step towards the reconstruction and recognition, we first perform some standard preprocessing operations. The preprocessed image is then thinned. Singular points are detected, and thin strokes between them traced. These data are finally mapped to a graph yielding a symbolic representation of the word. These operations constitute the graph building process. However, during its development, we observed that a long word can be reliably split into its components by vertical cuts. The cutting drastically reduces the complexity of the graph search procedure because only subgraphs are involved instead of the graph of the whole word.

Two main preprocessing operations are slant correction and base line detection. Slant is corrected through a shear transformation after estimating the slant angle [4]. The four baselines, that split the word into an upper, middle and a lower area, are found by analyzing the bounding box and the horizontal projection of the word. The four lines are used in a later step of the conversion process.

Thinning an image produces its skeleton but unfortunately also artefacts, see Fig. 1. Therefore, the skeleton has to be filtered to remove disturbing spikes (Fig. 1).

The resulting thinned image is split vertically at each column in the word where there is only one black pixel. At these positions we can assume that the writer passed from left to right once and then did not go back any more. With this assumption a reliable segmentation can be obtained very easyly (Fig. 2). Each of

0-8186-7898-4/97 $10.00

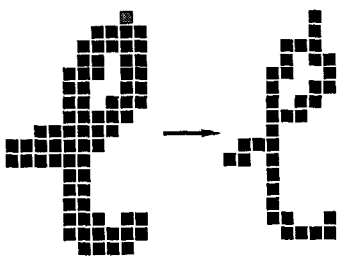


Figure 1: An example of spike added by the thinning algorithm.

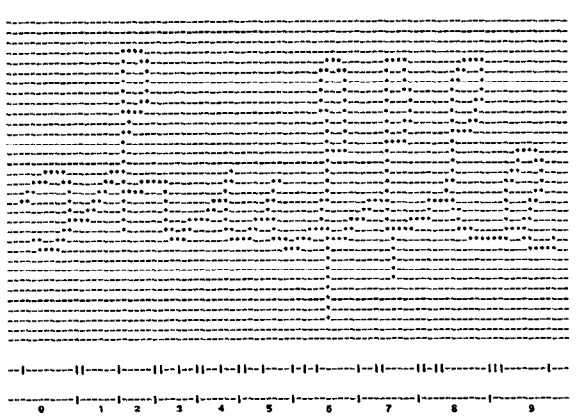
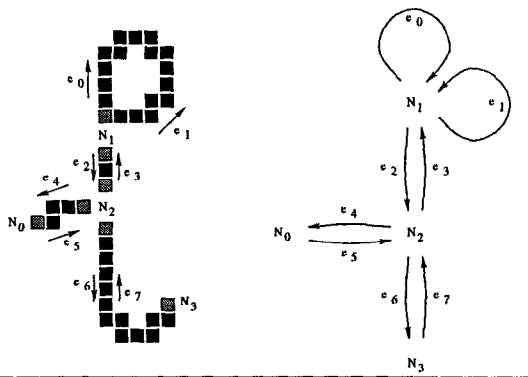


Figure 2: The word 'shuffle' is split into 10 components that can be treated separately.



edges	attributes
e_0	$N,N,N,N,E,N,E,E,S,E,S,S,S,W,S,W,W,W$
e_1	$E,E,E,N,E,N,N,N,W,N,W,W,S,W,S,S,S,S$
e_2	S,S
e_3	N,N
e_4	W,W,S,W
e_5	E,N,E,E
e_6	$S,S,S,S,S,S,E,S,E,E,N,E,N$
e_7	$S,W,S,W,W,N,W,N,N,N,N,N,N$

Figure 3: The representation of the skeletonized component by a directed, attributed graph

these new generated components can now be treated separately.

In the next step the components of skeletonized word can be represented by a *directed attributed graph*. All points with more than two or less than two neighbors are nodes while the points between two nodes are described in the graph by an edge. In this way an edge represents one single stroke without any crossing points. Again we can assume that each stroke was completely drawn before another or parts of another stroke were drawn. Since we do not know in which direction a stroke was drawn it is represented in our directed graph by two edges with opposite directions. An edge contains, for each point of the corresponding stroke, an attribute that encodes the direction of the stroke at this point. Working with a four-connected neighborhood we have four possible directions: $\{N, E, S, W\}$ (North, East, South, West). Figure 3 shows the graph and its representation for the component of Fig. 1.

2.2 Likelihood criteria

The problem of finding the drawing order of a word can now be expressed as searching, for each component of the word, the most likely path, and combining them to one single path. The paths are combined by concatenating them in the left to right order of the corresponding components.

To find the most likely path we have to define criteria that allow a measurement for a path. Many different criteria have been proposed in literature. Following an approach in [5] we distinguish between *global* and *local* criteria. The global criteria are dealing with

Figure 4: The complete stroke length of the word on the left side is smaller than the one on the right side, so this writing order is more likely.

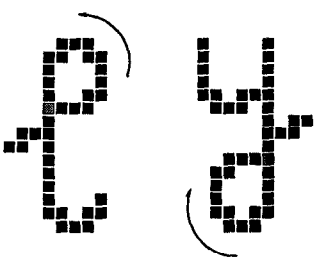


Figure 5: Loops in the middle or upper area are normally drawn in counter clockwise order while loops in the lower area are drawn in clockwise order.

the whole word.

- **writing direction:** Writing direction is from left to right, that means, writing starts in the left part of the word and ends in the right part.
- **path minimization:** The length of the complete path should be minimal. Like other human movements, the drawing movement is controlled by an attempt to minimize the energy needed to produce it (Fig. 4).

In contrast to the global criteria the local ones are only taken into consideration in small areas, for example around crossing points:

- **continuity criterion:** The continuity criterion is based on the assumption that the script features do not change rapidly following a line in the drawing order. In our work, the criterion is restricted to the new direction at a crossing point. This means that the angle between the old and the new stroke should be small.
- **direction of loop drawing:** Depending on their position, loops are usually drawn in clockwise (for loops in the lower part of the word) or in counterclockwise order (for loops in the upper or middle part of the words) (Fig. 5).

For each of these criteria we define a cost function that is minimum when the traversing path represents

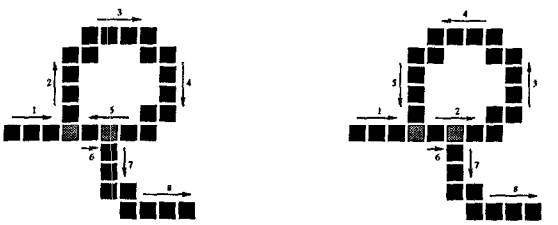


Figure 6: The traversing order with the strong restriction for a *valid* path (Fig. a) and the released (Fig. b).

a natural drawing order, and takes on large values otherwise. The cost of a path is defined as the weighted sum of partial costs that are evaluated using the attributes of the graph. For further details see [1].

2.3 Graph search

The graph search is based on the standard best-first search algorithm. However, due to the formulation of the problem, some additional restrictions on the path have to be imposed.

- A path has to be *complete*. In a path found, every line of the word has to be included. Since a stroke is represented by two opposite edges, this means that for all strokes at least one of their two opposite edges should be part of the path.
- The path has to be *valid*. On a valid path an edge is used only once. This does not mean that a stroke can not be drawn twice. Since each stroke is represented by two edges it is possible to traverse a stroke twice, but only in opposite directions. This is necessary for many letters, e.g. letter 'n' where the first vertical line is normally drawn twice, i.e. downwards and upwards. But it is not allowed to draw a stroke twice in the same direction. Because of digitalization effects, however, it can be necessary that a stroke has to be drawn twice in the same direction as Fig. 6 shows; otherwise a natural drawing order could not be recovered. Therefore this restriction is released for shorter strokes.

As *potential start nodes* for the gragh search, the nodes representing singular points at the left side of the component are taken into consideration. From these start nodes all paths are recursively built up and costs are evaluated. A path is registered as a possible solution when it is *complete* and *valid*. It is easy to see that the complexity of this graph search is exponential. Already for short words the number of possible paths can become quite huge as Table 1 shows. Therefore the graph search had to be expanded by a best- first search with integrated pruning. With the presented splitting of the words into separate components the search complexity can be decreased even more. In the

last column of Table 1 the number of possible paths for each component is presented.

3 Experimental Results

For our experiments we worked with a database we already used for the test of an off-line recognition system [3]. In this database the data was produced by cooperative writers. That is, we asked our writers to adhere as closely as possible to the writing style taught to first grade pupils at Swiss elementary schools. The vocabulary contained 150 randomly selected English words (see Table 2) and was written by five writers four times. These four sets were split into two groups where three sets of one writer were put to a training sample and the remaining one to the test sample. We made three different experiments, all of which using a state-of-the-art on-line recognition system [9], but trained on different databases:

1. Recognition of the reconstructed test data with the system that was trained with real on-line data[9] (ORIG).

2. Recognition with the same system as in experiment 1, but it was re-trained with the training samples of our database (CONT).

3. Recognition with the system that was only trained with the data from our training samples (NEW).

Additionally we worked with two different dictionaries, namely, the Unix spell checker (containing about 25'000 words) extended by 15 of our 150 words that are not in it, and our own dictionary containing exactly 150 words. They are referred to as 'Unix+' and 'our' in Table 3.

As one can see, the best result (97.4%) was achieved with the system that was trained with real on-line data and re-trained with our training samples. It is interesting to note that with our previous off-line recognition system [3], we obtained on this database a recognition rate of 98.0% which is similar. It may therefore be concluded that the conversion algorithm does preserve most of the discriminant information. Moreover, since the representation is different, it suggests that the results may favorably be combined with those obtained by the off-line recognizer to further improving the recognition rate.

4 Conclusion

We have presented a temporal reconstruction algorithm based on graph search. The input image is first preprocessed to compensate for various geometric variations. Thinning is then applied the result of which allows the construction of a graph whose nodes represent singular points and edges correspond to strokes. The best path in the graph is defined as the one that minimises the cost of both global and local criteria while satisfying some hard constraints, such as no strokes are left out. The algorithm is applied to a collection of off-line handwritten data, and the reconstructed 'on-line' data are then used in various recognition experiments. These results can be used, in combination with off-line results, to improving the recognition rate and/or reliability.

References

[1] R. Ammann, Rekonstruktion von Online-Information in der Handschrifterkennung, Master thesis, 1996.

[2] G. Boccignone, A. Chianese, L.P. Cordella, and A. Marcelli, "Recovering Dynamic Information from Static Handwriting," *Pattern Recognition*, Vol. 26, No. 3, pp. 409-418, 1993.

[3] H. Bunke, M. Roth, E.G. Schukat-Talamazzini, "Off-line Cursive Handwriting Recognition Using Hidden Markov Models," *Pattern Recognition*, Vol. 28, No. 9, 1995, pp. 1399-1413.

[4] T. Caesar, J.M. Gloger, E. Mandler, Preprocessing and Feature Extraction for a Handwriting Recognition System, *Proc. of the 2nd Int. Conf. on Document Analysis and Recognition*, Tsukuba, Japan, 1993, pp. 408-411.

[5] David S. Doermann and Azriel Rosenfeld, "Recovery of Temporal Information from Static Images of Handwriting," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Urbana, 1992, pp. 162-168.

[6] V. Govindaraju, D. Wang, and S.N. Srihari, "Using Temporal Information in Off-Line Word Recognition," *Advanced Technology Conference*, United States Postal Service, Nov. 30–Dec. 2, 1992, pp. 529-543.

[7] S. Jäger, "Recovering Writing Traces in Off-Line Handwriting Recognition: Using a Global Optimization Technique," Int. Conf. on Pattern Recognition, Vienna, Austria, August 25-29, 1996, pp. 150-154.

[8] E. Mandler, R. Oed, and W. Doster, "Experiments in on-line script recognition," *Proc. 4th Scandinavian Conf. on Image Analysis*, June 1985, pp. 75-86.

[9] M. Schenkel, I. Guyon, and D. Henderson, "On-line Cursive Script Recognition using Time Delay Neural Networks and Hidden Markov Models," *Machine Vision and Applications*, Vol. 8, 1995, pp. 215-223.

[10] R. Seiler, M. Schenkel, and F. Eggimann, "Off-Line Cursive Handwriting Recognition Compared with On-Line Recognition," *Proc. 13th Int. Conf. on Pattern Recognition*, Vol. IV, Vienna, Austria, Aug. 25-29, 1996, pp. 505-509.

Word	Simple graph search	Best-first with pruning	Pruning and segmenting ($\sum$)
car	1'040	283	1, 1, 32, 1, 1, 6, 1 (43)
edit	40'572	8'870	9, 1, 198, 1, 133 (342)
day	77'578	21'606	61, 20, 2'662 (2743)
able	186'572	21'248	4, 1, 11, 7, 13, 113 (149)
deep	246'234	58'494	41, 21, 15, 14 (91)
lazy	295'018	23'434	13, 29, 1, 80, 261 (384)

Table 1: Complexity of the graph search for a few words

Set 1			Set 2		
able	accomplish	avenue	afternoon	agree	although
balance	broom	busy	beneficial	blood	breath
car	casual	celebrate	chance	cheap	cigarette
day	deep	discover	decide	disappoint	dream
earth	edit	expensive	easy	empty	evening
follow	free	furniture	familiar	fetch	fiction
garage	general	grill	gesture	ghost	give
high	honey	hurry	harbour	help	highway
ignore	illegible	index	identical	illuminate	incomplete
jealous	jewel	joke	jacket	journey	judge
ketchup	kind	knock	kettle	key	kidnap
lazy	listen	luxury	legal	lesson	lunch
maintain	merchandise	mighty	man	moderate	morning
narrow	neglect	night	name	naughty	nobody
obsolete	office	oscillate	obtain	omit	opposite
parallel	photograph	please	paper	persuade	pretend
quarrel	queen	quick	quantum	question	quiver
ready	recommend	riddle	remember	return	rise
script	shuffle	splendid	sample	school	silent
thread	transfer	typical	telephone	this	tough
umbrella	unlock	upward	uncertain	unique	usually
vague	vegetables	vote	very	visible	vision
water	wear	wide	which	window	woman
xylonite	xylophone				
yacht	yield	young	yawn	yes	yesterday
zipper	zoom		zebra	zero	

Table 2: The 150 used words of our vocabulary

Recognizer	Training iterations	Vocabulary		
		none	Unix+	our
ORIG	-	1.4	26.1	72.7
CONT	54	28.0	79.7	97.4
NEW	260	16.9	75.6	93.7

Table 3: Word recognition rates in percentage for the original (ORIG), the re-trained (CONT), and the newly trained (NEW) recognizer networks with different dictionaries. Unix+ is the original Unix dictionary extended by 15 of our 150 words which are not contained in it.