On the Possibility of Instance-Based Stroke Recovery

Yutaro Iwakiri Soma Shiraishi Yaokai Feng Seiichi Uchida
Kyushu University
Fukuoka, Japan
{*iwakiri, shiraishi, fengyk, uchida*}*@human.ait.kyushu-u.ac.jp*

***Abstract*—This paper tackles the stroke recovery problem, which is a typical ill-posed reverse problem, by an instance-based method. The basic idea of the instance-based stroke recovery is to refer to the drawing order of a similar instance. The instance-based method has a strong merit that it can deal with multi-stroke characters and other complex characters without any special consideration. However, it requires a sufficient numbers of instances to cover those various characters. As an initial trial of the instance-based stroke recovery method, this paper describes the principle of the method and then provides several experimental results. The experimental results indicate the potential of the proposed method on recovering the drawing order of complex characters, as expected.**

***Keywords*-handwriting; stroke recovery; instance base; time series; big data;**

I. Introduction

Stroke recovery is a technique to estimate the drawing order of a given handwritten character image. For example, for a handwritten image of "x", a correct stroke recovery result is "$>$"$\rightarrow$ "$<$" (where both strokes are written in the top-to-bottom direction). Similarly, for "χ", a correct result is "$\backslash$"$\rightarrow$ "$/$". Stroke recovery is a typical inverse problem and therefore it is difficult to obtain correct results. The above examples of two similar characters "x" and "χ" clearly indicate how the problem is difficult.

For single stroke characters, like "α", the recovery method based on graph theory [1] seems promising. This method relies on a simple but effective algorithm, called basic trace algorithm. Starting from one of two end points of a single stroke character, the algorithm traces the stroke and if it encounters an intersection (a node of degree 4), it just takes the center way. Even this simple algorithm, it can recover the stroke order of single but complex stroke characters, such as "&". However, it has several limitations. First, it requires special treatments on double-traced lines, such as the vertical stroke of "d". Second, it cannot determine the global writing direction. For example, it is essentially impossible for the method to determine whether "α"-shaped single stroke is written from top or from bottom. Third, it cannot deal with multi-stroke characters. Therefore this elegant method cannot deal with "x" and "χ". Most of other existing methods, such as [2], [3], [4], have the same difficulties because they also formulated the stroke recovery problem as a trace problem.

Exceptionally, Qiao et al. [5] proposed a method which can recover the stroke order of multi-stroke characters. Their method is a kind of model-fitting methods [6], [7], where a one-dimensional stroke model is fitted nonlinearly to a two-dimensional input image. Since the drawing order of the model is known, the fitting result directly shows the recovery result. One drawback of this method [5] is its difficulty to control the relationship among strokes. (For example, any pair of two strokes should not be fitted to the same stroke on the image.) In addition, variations of stroke order and stroke number are not considered. Consequently, must be designed an individual model for each of those variations.

This paper tackles this difficult stroke recovery problem by a simple but novel approach, that is, an *instance-based approach* is proposed. The principle of this approach is to consider that the more similar two character images become, the more similar their drawing order is. Therefore, if we have a large number of character images with their true drawing order as an instance set, a reliable recovery result of an input image canbe obtained by referring to the drawing order of the most similar instance. As we will see later, the method is formulated as an optimal path problem, where a path represents a sequence of removing the black pixels one by one from the input character image.

One important merit of the instance-based approach is that it can deal with multi-stroke characters and other complex characters without any special consideration. Theoretically, just by adding instances of multi-stroke characters, we can refer to them for the stroke recovery of multi-stroke character images.

We need a sufficiently large number of instances for a good-enough recovery result. This is because the proposed method needs a similar instance even for severely deformed input image. In our experiment, about 15,000 instances of 10 digit classes were used for showing that more instances are necessary for better recovery accuracies.

This research has been inspired by recent "big data" researches in image processing and pattern recognition. Those researches have proved that even the simple nearest neighbor methods can provide very good performance for complex tasks if a large instance set is available. For example, in [8], 80-million tiny images were used in nearest neighbor-based recognition. Visual object segmentation has also been tackled with a large instance set [9]. A large

978-0-7695-4774-9/12 $26.00 © 2012 IEEE
DOI 10.1109/ICFHR.2012.248

ground-truthed video data set has also provided and utilized for scene understanding [10], [11]. The proposed stroke recovery method is considered as a method of inferring an image sequence to reach the input character image by using a large instance set.

II. Basic Idea of Instance-Based Stroke Recovery

As mentioned above, the basic idea of instance-based stroke recovery is to refer to the drawing order of a similar instance. One may simply consider to use the drawing order of the similar instance "directly" as the recovery result. However, this simple method is obviously irrelevant. This is because such a result shows the drawing order of the instance and does not indicate that of the input[1].

The above observation explains an important fact that possible drawing orders are strictly constrained by the input image. Consider an input image of "3". Our goal is to estimate the optimal sequence of "intermediate" images from an empty image to the completed image "3". Each intermediate image is constrained so that its black pixels should be a subset of the black pixels of the input image "3". Therefore, if "3" contains 10 black pixels, there are up to 2^{10} intermediate images which satisfy the constraint, and the stroke recovery result is one of the ordered sequences of those images.

Figure 1 illustrates our stroke recovery problem. This is an image space and thus an instance (a sequence of intermediate images) is represented as a continuous trajectory. The problem is to determine the optimal trajectory from the input image to the empty image (the origin) while referring to instances. The green area shows the set of all possible intermediate images for the input image. Thus, the optimal trajectory should be included in this area. Considering "instance 1" is of the most similar image, the above simple method fails because the trajectory of instance 1 is not included in the green area.

Instances are used in the criterion to determine the optimal trajectory. In fact, there are 10! possible recovery results for the input image with 10 black pixels. For selecting the best results among them, instances are used. Roughly speaking, a trajectory which always keeps some proximity to one or more instances is a good trajectory. In other words, every intermediate image of the recovery result should have a similar image in the instance set. Consequently, the proximity (or, inversely, the distance) accumulated along the recovery result is the criterion for selecting the optimal result.

[1]If we can find the exactly same instance in the database, we can use the drawing order of the instance directly as the result. Of course, we cannot expect such an optimistic situation, which requires almost infinite instances.

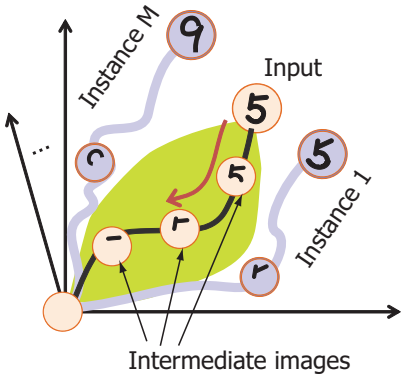

Figure 1. Basic idea of instance-based stroke recovery.

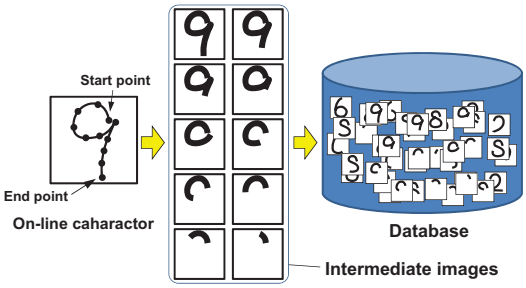

Figure 2. Preparation of instances.

III. Detailed Recovery Process

The instance-based stroke recovery method is organized in a three-step manner, that is, preparation of instances, preprocessing and optimal drawing order estimation. In the following, those steps are detailed.

A. Preparation of Instances

An instance is a sequence of intermediate images of character drawing processes and created from an on-line handwritten character data. Figure 2 illustrates a process of creating an instance "9". An intermediate image is created by so-called "inking" process, where the pen-tip movement from $t = 0$ to t' is converted into a black-ink trajectory on a bitmap image with an appropriate ink width. Hereafter, consider $t' = 0, \tau, 2\tau, 3\tau, \ldots, T-\tau, T$. Consequently, T/τ intermediate images are created with the regular interval τ. The larger τ is, the rougher the instance becomes. Figure 3 shows several examples of instance images. The instances of (a) are written by the general stroke order and (b) are written by a rare order.

B. Preprocessing

We perform preprocessing to convert the input image as a set of line segment. Letting N denote the number of segments, we will consider the stroke recovery problem as a problem to find the optimal order of removing N line segments one by one from the input image. Although

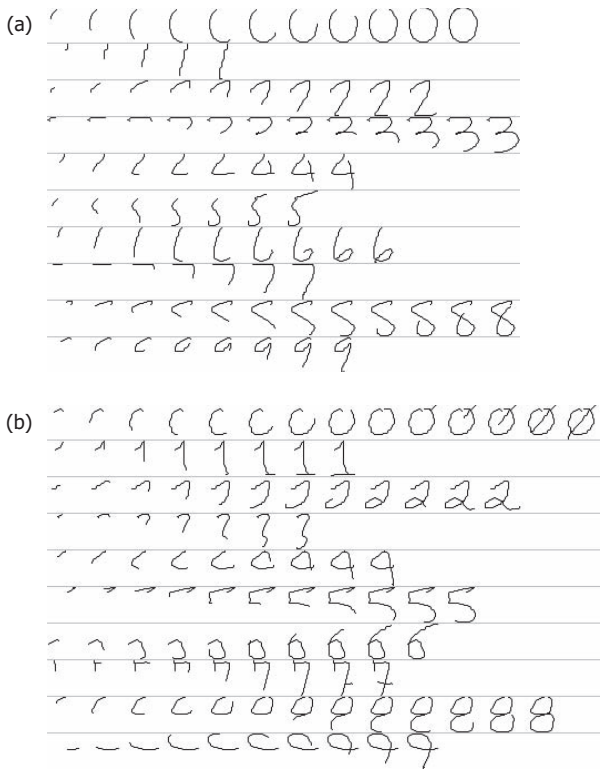

Figure 3. Sample of instances.

Figure 4. Example of segmentation.

individual pixels are considered as a unit of removal in Sections I and II, line segments are a better unit from the viewpoint of computational efficiency and accuracy.

Figure 4 shows a segmentation result ($N = 5$) provided by the following method. First, intersection points are detected and then the black ink strokes are cut at those points. Second, the obtained line segment is further divided at the halfway point if the segment has a high curvature or a long length. Finally, a very short line segment is combined with its nearest neighbor segment. Note that it is difficult to apply this combination process for all short line segments because some of them are important. Consequently, we cannot remove all noisy short line segments, that badly affect recovery accuracy as we see later.

C. Optimal Drawing Order Estimation

For determining the globally optimal order of removing N line segments, we will formulate the problem as an optimal path problem on a *cube graph*. The simple brute-force search requires $O(N!)$ computations for examining all possible permutation of N segments. In contrast, the following algorithm on the cube graph requires $O(2^N)$ computations, which is far less than $O(N!)$. This efficiency comes from the fact that the algorithm, called cube search [14], is based on dynamic programing search on the cube graph.

Figure 5 illustrates the formulation of the stroke recovery problem as an optimal path problem on the cube graph. As noted above, if we have N line segments, the number of the possible intermediate images are 2^N and thus the cube graph has 2^N nodes. Figure 5 illustrates the cube graph for "5" with $N = 4$ segments and it is comprised of 16 nodes divided into 5 layers. The k-th layer contains ${}_NC_k$ nodes, each of which corresponds to an intermediate image with k missing segments. Note the first and the last layer contains the start and the end nodes corresponding to the input image and the empty image, respectively.

Any path between the start and the end nodes represents one possible removal order. Therefore if we add any appropriate cost on every edge of the graph, the stroke recovery problem is converted into an optimal path problem on the cube graph. As noted above, the optimal path can be found with $O(2^N)$ computation.

The remaining factor to complete this optimization problem is the design of the cost, (or say the objective) function of the problem. In this paper, we add a cost to each node of the graph. As mentioned in Section II, each node, i.e., each intermediate image on the path should have a similar image in the instance images. Thus, we define the node cost as the minimum distance to the instance images. The cost becomes smaller if there is a similar intermediate image in the instance set. Consequently, we can expect the optimal recovery result, all of whose intermediate images are as much as similar to the instances.

There are two variations in the cost evaluation. Hereafter, we will call them Method A and Method B.

- **Method A** The minimum distance image is always searched in the whole instance set. Thus, the minimum distance images can come from different instances at individual nodes. For example, the upper part of an instance of "3" and then the lower part of an instance of "2" can be combinatorially used for an input image. Since any combination is allowed, Method A disregards the original writing order of the instances.
- **Method B** The minimum distance image is searched only from the intermediate images of a single instance, which is created by the most similar (completed) character image to the input images. In other words, first, the nearest neighbor instance (say an instance of "3") is searched for and then along the instance the path on the cube graph is optimized. Note that Method B is different from the simple method introduced in Section II as an inappropriate method.

Chamfer distance [15] is used for measuring the node cost, i.e., the distance of two (intermediate) images. Chamfer distance is defined as follows. First, edge and distance transformation images are created from two binary images:

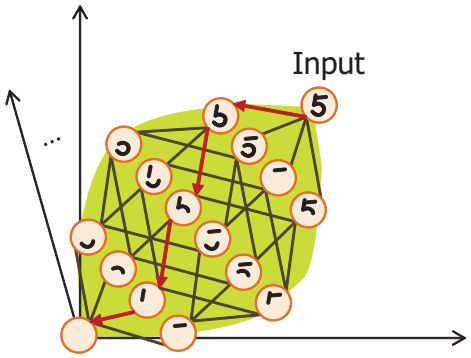

Figure 5. Instance-based stroke recovery as an optimal path problem. ($N = 4$.)

the input image and an instance image. Second, the sum of distance value on the edges is counted by overlapping them, and then the sum is divided by the number of edge pixels.

IV. Experimental Results

A. Datasets

The following experiment was conducted with on-line character patterns extracted from the Unipen database, which contains 15,953 samples in total. From those samples, 16 different instance datasets were prepared by changing a parameter τ as 2, 4, 8 and 16 at the inking process and the number of the instances M as 10, 100, 1,000 and 10,000. Specifically, the largest instance dataset ($\tau = 2$ and $M = 10,000$) contains 397,290 intermediate images, and the smallest ($\tau = 16$ and $M = 10$) contains 46 images. As test patterns, other 1,000 samples were extracted from the same Unipen database and then converted as 1,000 images through the inking process. The number of line segments per pattern, N, varied from 2 to 17 and its average was 5.9.

Each instance had no class information. Consequently, all of the instances from any class can be used for any input. This was a very general condition because it is not necessary to know the class of the input. As shown in Section IV-D, it was observed experimentally that if the instances are limited to the class of the input pattern, the recovery accuracy was increased.

B. Evaluation Criterion

Since each test pattern (i.e., the input image) was created from on-line characters, its correct drawing order has been known. If a recovery result is exactly the same as the correct drawing order, the result will be considered as successful one. This criterion is very strict. For example, if the order of two line segments is interchanged and the order of the remaining $N - 2$ segments is correct, it was counted as a failure result. Another precise evaluation criterion, called Kendall distance [16], is also used and discussed later.

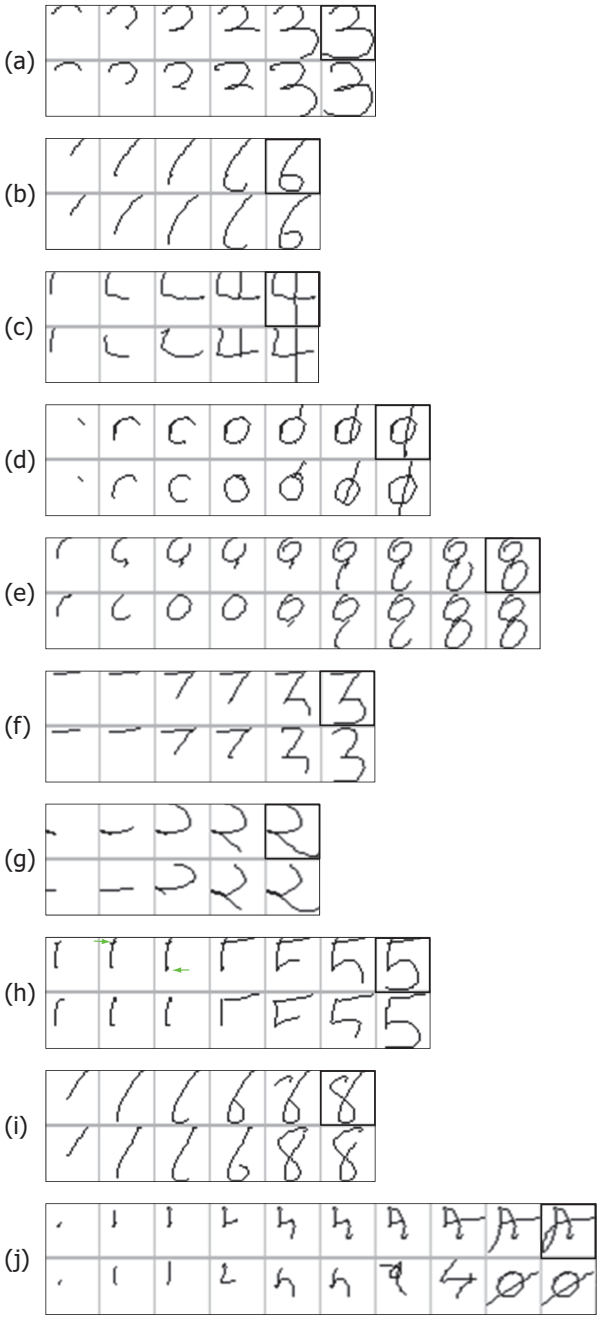

Figure 6. (a)–(f) sample of correct estimation and (g)–(j) incorrect estimation. In each lines, upper parts are recovery result and lower parts are reference images.

C. Qualitative Evaluation

1) Successful Results: Figures 6 (a)–(f) show successful recovery results under $\tau = 2$ and $M = 10,000$. In each line of this figure, the upper part is a recovery result by Method A and the lower part is the corresponding nearest images from the instance set. The input image is at the upper right end.

Figures 6 (a) and (b) show that the proposed method can

recover a drawing order using similar instance images. Even for those simple inputs, conventional method needs special considerations. In fact, the input image of (a) contains a double-traced line, and (b) has no end clear point. Conventional recovery methods need start and end points and thus have difficulty to deal with (b). The proposed method does not have such a problem.

Figures 6 (c) and (d) are recovery examples of multi-stroke characters. As mentioned before, only a few of the existing methods can deal with multi-stroke characters. Especially, (d) has only two end points and thus some conventional methods will misunderstand that it is a single stroke character. Our proposal could recover their drawing orders correctly just by referring instance without any special considerations. In addition, Figure 6 (e) shows a result of a stroke recovery for a character written by irregular order and stroke number. The proposed method is capable of recovering irregular strokes, if the dataset contains corresponding instances.

Figure 6 (f) shows an example that was recovered successfully by Method A but failed by B. This "3" has a particular shape around its upper right corner and there were few "3"s like it in the dataset. Method A, however, combined several instances ("7" and "3") and finally had the correct result. Note that Method A also combined several instances for (d) and (e). Method B uses the single (nearest) instance and thus difficult to deal with the input pattern largely deviated from the instances.

2) Failure Results: Figures 6 (g)–(j) are examples of failure recovery results. Figure 6 (g) shows a result whose recovery images are very similar to the referred instance image but its recovery result is erroneous. Although the input "2" is drawn from upper to lower like popular drawing order, the estimated order chose the stroke to start from the halfway point. Method A sometimes encounters this problem as the side effect of its ability to combine multiple instances regardless of their original drawing orders.

Figure 6 (h) shows a failure result where wrong estimation happened around a very short line segment (pointed by the green arrow). Since the cost by a short segment is small, its order often becomes unstable. The failure of Fig. 6 (i) was caused by the combination of multiple instances from different classes. An instance of "6" was partially used for "8". The failure of Fig. 6 (j) was caused by the lack of appropriate instances. The proposed method totally relies on similarity of the input image to instances and therefore cannot estimate the correct drawing order without them.

D. Quantitative Evaluation

Table I shows the successful recovery rates for each pair of M and τ. Considering the evaluation criterion of IV-B is very strict, this result suggests the possibility of the instance-based stroke recovery. Note that the latter evaluation with Kendall distance [16] reveals that recovery error often happens only locally.

Table I
RECOVERY ACCURACY(%).
EACH CELL SHOWS RATES OF METHODS A AND B.

	$M = 10$		100		1,000		10,000	
τ	A	B	A	B	A	B	A	B
2	22.1	20.7	28.8	31.8	39.8	44.7	41.8	51.7
4	21.9	21.1	29.4	32.0	39.6	44.6	42.1	51.2
8	21.0	20.8	27.8	31.0	38.1	43.0	42.5	51.0
16	20.0	17.4	25.4	25.8	34.5	33.0	38.5	37.2

Since the experimental result indicates that the accuracy increases as the number of the instances M increases, the effect of using large instance was confirmed in a quantitative viewpoint. The same effect is more significant in Method B. Method A also shows a positive but limited effect because of the increase of failures like Fig. 6 (g).

An additional experiment with Method A was also conducted, where referable instances were limited. Specifically, for example, if the input image is "0", only the instances of "0" were used. Although this limitation reduces the number of referable instances to about $M/10$, the success rates were 21, 30, 45 and 50%, respectively, at $M = 1, 10, 100$, and $1,000$, respectively , under $\tau = 2$. This improvement implies that similar instances of different classes often have a bad influence, as shown in Fig. 6 (i). Note that this condition at $M = 1$ is similar to the condition of the fitting-based recovery method like [5], because the method relies on a single model from the same class.

Table I shows that the value of τ has a little effect for Method A. Method A can compensate the decrease of instance variations according to the increase of τ by combining multiple instances. On the other hand, τ has a larger effect for Method B, which cannot combine different instances.

Figure 7 shows the distribution of the recovery cost, which is the accumulated cost along the optimal path. As expected, the cost becomes smaller as the instance set becomes larger. This indicates that a more similar instance image can be found at each node of the cube graph.

Figure 8 shows the distribution of Kendall distance [16] of each recovery result. Kendall distance (or so-called bubble-sort distance) can evaluate how the estimated order is different from the correct order. Specifically, like edit distance, Kendall distance evaluates the number of steps to convert the estimated order to the correct order by swapping the neighboring order values. Then, by normalizing the number of steps by its maximum $N(N-1)/2$, Kendall distance takes a value between 0 and 1. For example, Kendall distance is $2/(N(N-1))$ for the pair 1-3-2-4-5 and 1-2-3-4-5 and, $4/(N(N-1))$ for the pair 1-3-4-2-5 and 1-2-3-4-5.

In Fig. 8, most Kendall distances take a smaller value around $0 \sim 0.2$. Therefore the recovery error happened

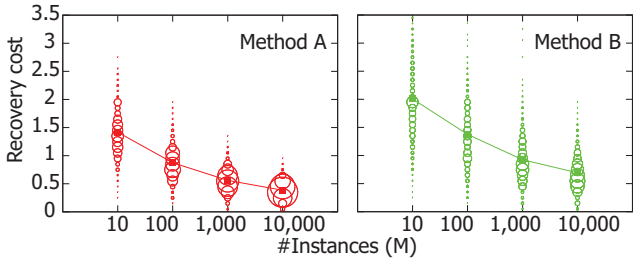

Figure 7. Recovery cost. The square symbol "■" represents the average value.

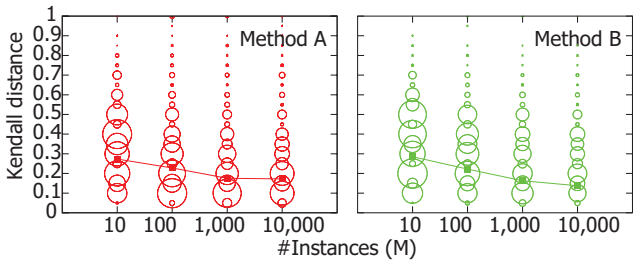

Figure 8. Kendall distance. The circles for zero Kendall distance were omitted for better visibility.

locally. (Assume a typical erroneous result 2-3-1-4-5-6 for $N = 6$. This happens especially when the first line segment "1" is very short. Kendall distance of this case is 0.13.)

V. Conclusion and Future Work

In this paper, we presented a stroke recovery method by using large instance image datasets. For more than 50% of handwritten digit character images with various drawing orders and shapes, we could have a good recovery result. Especially, it should be emphasized that the proposed method can deal with multi-stroke characters without using any extra rule or complex writing model. It was also confirmed that more instances provide better recovery accuracy.

For better recovery accuracy, future work will focus on (i) the improvement of distance metric to the instance, (ii) the use of the original temporal information of the instance, (iii) the improvement of line segmentation (suppression of noisy short line segment) and (iv) instance selection. In addition, we should not forget that there is an inevitable limitation in stroke recovery — if there are the exactly same patterns drawn in different orders, none can provide perfect estimation. This fact indicates that we need other criteria which evaluate the performance more fairly.

References

[1] Y. Kato and M. Yasuhara, "Recovery of Drawing Order from Single-stroke Handwriting Images," PAMI, vol. 22, no. 9, pp. 938–949, 2000.

[2] D. S. Doermann and A. Rosenfeld, "Recovery of Temporal Information from Static Images of Handwriting," IJCV, 15, 143–165, 1995.

[3] S. Jäger, "Recovering Writing Traces in Off-Line Handwriting Recognition: Using a Global Optimization Technique," Proc. ICPR, 1996.

[4] E. -M. Nel, J. A. du Preez, and B. M. Herbst, "Estimating the Pen Trajectories of Static Signatures Using Hidden Markov Models," PAMI, 27(11), 1733–1746, 2005.

[5] Y. Qiao and M. Yasuhara, "Recover Writing Trajectory from Multiple Stroked Image Using Bidirectional Dynamic Search," Proc. ICPR, 2006.

[6] M. Revow, C. K. I. Williams, and G. E. Hinton, "Using generative models for handwritten digit recognition", PAMI, 18(6), 592-606, 1996.

[7] T. Kato, S. Omachi, and H. Aso, H. "Precise hand-printed character recognition using elastic models via nonlinear transformation", Proc. ICPR, 2000.

[8] A. Torralba, R. Fergus, and W. T. Freeman, "80 Million Tiny Images: A Large Data Set for Nonparametric Object and Scene Recognition," PAMI, 30(11), 1958–1970, 2008.

[9] C. Liu, J. Yuen, and A. Torralba, "Nonparametric Scene Parsing via Label Transfer," PAMI, 33(12), 2368–2382, 2011.

[10] A. Karpenko and P. Aarabi, "Tiny Videos: A Large Data Set for Nonparametric Video Retrieval and Frame Classification," PAMI, 33(3), 618–630, 2011

[11] J. Yuen, B. C. Russell, C. Liu, and A. Torralba. "LabelMe Video: Building a Video Database with Human Annotations," Proc. ICCV, 2009.

[12] C. Kirsopp, M. Shepperd, and J. Hart, "Search Heuristics, Case-based Reasoning And Software Project Effort Prediction," Proc. GECCO, 2002.

[13] M. D Jære, A. Aamodt, and P. Skalle, "Representing Temporal Knowledge for Case-Based Prediction," Proc. European Conf. Advances in Case-Based Reasoning, 2002.

[14] W. Cai, S. Uchida and H.Sakoe, "An Efficient Radical-Based Algorithm for Stroke-Order-Free Online Kanji Character Recognition," Proc. ICPR, 2006.

[15] G. Borgefors, "Hierarchical Chamfer Matching : A Parametric Edge Matching Algorithm," PAMI, 10(6), 849–865, 1988.

[16] M. G. Kendall, "A New Measure of Rank Correlation," Biometrika, 30(1/2), 81–93, 1938.