Curved-Straight Lines-Analysis (CSLA) Algorithm for Handwritten Digit Recognition Enhancement

Young Suk Cho
GVU Center,
Georgia Institute of Technology
Georgia USA, 30332 Atlanta,
ycho47@gatech.edu

Hyungsin Kim
College of Computing
Georgia Institute of Technology
Georgia USA, 30332 Atlanta,
hyungsin@gatech.edu

Ellen Yi-Luen Do
College of Architecture[3]
Georgia Institute of Technology,
Georgia USA, 30332 Atlanta
ellendo@gatech.edu

ABSTRACT
In this paper, we propose a new recognition algorithm for handwritten digit recognition. This algorithm is designed to enhance the recognition accuracy of current Microsoft SDK recognizer. The algorithm recognizes the unique signature of each number by comparing curved and straight lines, and writing sequences of the stroke. Through the trial experiments, we achieved 97.67% of positive recognition accuracy.

Categories and Subject Descriptors
I.5.4 [**Pattern Recognition**]: Applications; H5.2 [**Information interfaces and presentation**]: User Interfaces. - Graphical user interfaces.

General Terms
Algorithms, Design, Human Factors

Keywords
Pen-based UIs, input and interaction technology, sketch recognition

1. INTRODUCTION
The Clock Drawing Test is one of the most effective diagnostic measures utilized for detecting people with dementia [7]. A test participant is asked to draw a clock on the paper with the time 10 past 11. When a participant finished drawing a clock, the clock drawing is manually graded by a test instructor. This is a time-consuming process that can be replaced by computational power. Also, the paper-based test only gives the data from the final result of clock drawings. It is hard for a human to capture the data produced during the test which might be the helpful information to analyze the disease. Therefore, the computerized measure to score the drawings automatically and to capture the data related to the drawing process is needed.

In this paper we first present our ClockReader system briefly then introduce a new recognition algorithm named Curved-Straight Line Analysis (CSLA) to enhance recognition accuracy. Finally, we conclude this paper and propose future research direction.

2. CLOCKREADER SYSTEM
We developed the ClockReader system to enhance the paper-based Clock Drawing Test by using Tablet PC to manage and collect useful test data [7]. When a participant draws a clock on the Tablet PC, the program collects all information from the drawn stroke. The collected information (e.g. Bezier and cusp points for each stroke, etc.) is stored in the memory. Then the processor generates a rectangle-shape dynamic recognition region for each stroke and compares it to the best-matched character. After the recognition process is over, the program analyzes the drawn clock and scores under given criteria [2]. The ClockReader system is developed in C# programming language and supported by "Microsoft Windows XP Tablet PC Edition Software Development Kit 1.7" and "Microsoft Visual Studio 2008". Figure 1 shows the User Interface of the ClockReader system for the test participants.

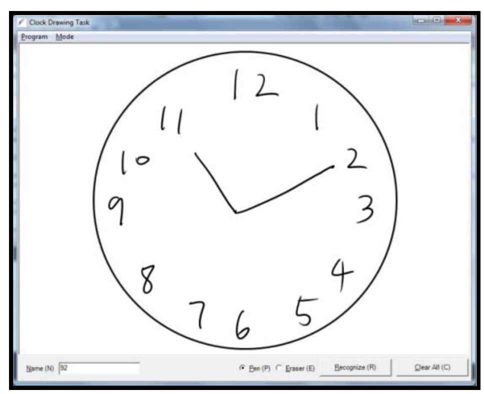

Figure 1. User Interface of ClockReader

3. RELATED WORK
Several researchers have developed various recognition algorithms [4, 8] to recognize handwritten characters using computational technology. These algorithms can be classified into two types such as offline recognition and online recognition. Offline recognition analyzes the characters from scanned raster images [5]. On the other hand, online recognition provides the

real time captured data and recognition result simultaneously [5]. Well-known online recognition algorithms include the $1 recognizer and the $N recognizer designed by J. Wobbrock et al [5, 8]. The $1 recognizer obtains over 97% accuracy with only 1 loaded template and 99% accuracy with 3+ loaded templates [8]. Even though it has significantly high recognition accuracy, the $1 algorithm only can recognize one stroke at a time. Thus, recognizing more than two-stroke digits such as 4 and 5 is not possible. The $N algorithm improves the $1 algorithm, enables multi-stroke recognition [1]. However, it also has some limitation in stroke recognition for normal handwriting.

CueTIP interface achieves some error correction progress in handwritten recognition by providing the intuitive user interface that allows error correction by simple stroke such as horizontal, vertical, or diagonal line on the result panel [6]. The system remembers each correction made by the user. It enables the user to make a correction with a small number of actions because it proposes possible correction candidates based on the stored correction data. This technology is currently adopted by the tablet pen writing panel in Microsoft Windows. However, in our ClockReader application, providing error correction suggestions may affect the performance of the patient with dementia.

Recently, we have developed a Context-bounded Refinement Filter (CRF) algorithm to enable the pro-gram to automatically correct handwritten digit error based on the learning error data from the user [3]. The CRF algorithm adopted Microsoft Tablet PC SDK (MS SDK) as the main recognizer and converts the misrecognition from the main recognizer by comparing the error database for each digit. When a misrecognized character is detected, then the program looks up that character from the database and converts it if there is a matched character. However, we have found that most of misrecognition result come from the situation in which MS SDK recognizer misrecognized a number as some of the different characters. We know that a clock have only numbers so it needs handwritten digit recognition but not handwritten character recognition. In addition, since a clock is composed of numbers and simple line drawings to represent an hour and a minute hand, we needed to separate and recognize each stroke as a drawing or a character. Unfortunately, MS SDK does not provide support for this simultaneous recognition technology.

Yin, and Sun proposed that stroke segmentation and sketch recognition in one process. [9] Their algorithm focuses on analysis on the relations between two adjacent primitives and produces multiple candidates. However generating multiple candidates still requires an additional refinement process to filter out the exact recognition result. In addition, considering the maximum number of primitives needed in writing a number from 1 to 12 is 2, such a highly sophisticated but complicated algorithm which takes up to 450ms for complex shaped multiple primitives was not recommendable.

Ultimately, since a complete clock is composed of two hands and twelve numbers, a recognition algorithm which supports simultaneous sketch and digit recognition was required. However, finding the recognizer which contains the abovementioned functionality was not easy. After we investigated several off-the-shelf recognizers, we realized that there was no appropriate ready-to-use recognizer for our ClockReader project due to the following reasons. First, we could not use the $1 recognizer as it does not support multi-stroke recognition [8]. Second, the $N algorithm has some limitations to recognize stroke accurately [1]. Lastly, we cannot adopt MS SDK since it does not support the sketch and character recognition together [3]. There-fore, rather than using these existing algorithms, we developed our CSLA (Curved-Straight Lines Analysis) algorithm to improve the performance of hand written digit recognition. The goal of the CSLA is to implement robust and accurate digit recognizer, to recognize a character regardless of its writing speed, to embrace multi-stroke character recognition, and to recognize drawings and digits simultaneously.

4. FIGURES/CAPTIONS

The CSLA algorithm adopted MS SDK to capture the stroke coordinate data. Using this SDK, we can collect various information of each stroke such as Bezier points, cusp points, and the recognition area.

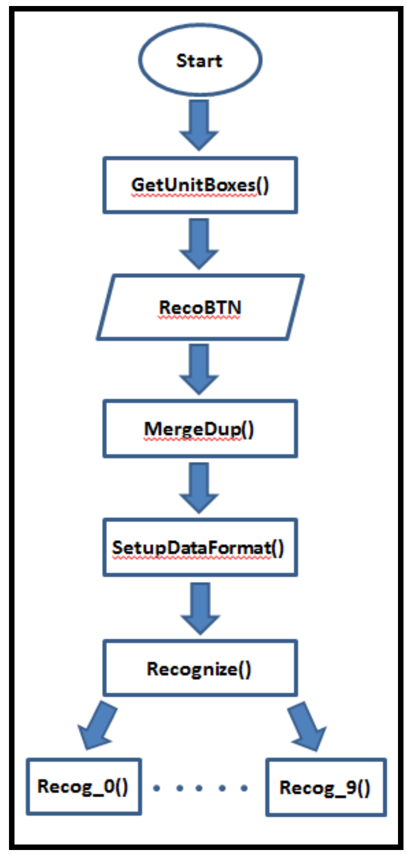

Figure 2. The flow chart of CSLA algorithm

Figure 2 illustrates the entire flow of CSLA algorithm. When the program is initiated and the user starts writing a stroke, the program stores each stroke and pass it to GetUnitBoxes(). During this phase, the program sets up a rectangle area around each stroke and defines the recognition area. This pre-processing phase decides the recognition area for each stroke before the actual recognition process is started. The recognition areas are shown as green boxes on the Tablet panel. Each recognition area will be formulated by MS SDK recognizer respectively. Figure 3 shows the pre-process result of several digits. These recognition area set up is triggered whenever the user add or erase a stroke on the panel.

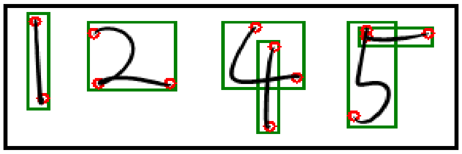

Figure 3. Pre-process result of single-stroke digits (1, 2) and multi-stroke digits (4, 5)

Once the user finishes the clock drawing, the pre-processing phase is done and the recognizer then waits for the user input (RecoBTN) to initiate the actual recognition process.

The MergeDup() process is started when the user press the "Recognize" button on the application. In this process, the processor merges multiple recognition areas into one if they are overlapped. There are some digits that are composed of more than 1 stroke such as the numbers 4 and 5, and sometimes even the numbers 7 and 8 would have multiple strokes. For these multi-stroke digits, the program checks each recognition area to see whether it is overlapped with another recognition area. When the program found two overlapped recognition area then it merges the two areas into one.

In SetupDataFormat() phase, the processor reserves the space for all necessary information which will be utilized by CSLA algorithm. It contains the storages for cusp points, Bezier points, coordinates of recognition area, and recognition result. Once all information except recognition result is gathered, SetupDataFormat() calls the actual recognizer.

When the Recognize() method is called the recognizer starts main recognition process. It has 10 recognition function for each digit and calls from Recog_1() to Recog_9() and Recog_0() sequentially. Each re-cognizing module stores the unique signature of each corresponding digit. When the stroke is delivered to one of these modules to be examined, the module checks the stroke whether it has the matching characteristics with the unique signature of the digit. If the module confirmed that the characteristics are matched, then the module determines assigns the stroke its number, deliver it to the next module otherwise.

During the main recognition process, the program subdivides the stroke into two parts - a curved line or a straight line - based on the two points of cusp points and some of Bezier points. For example, we know that number 2 is composed of a curved line in the upper bound and horizontal straight line in the lower bound. Also, we can calculate how much difference between the expected coordinates and actual coordinates of each stroke. Expected coordinate refers a coordinate of imaginary sampling point in a particular stroke if the stroke was drawn straight. Actual coordinate is a coordinate of which a line of stroke was actually passed by. That means, if there is big difference between an expected coordinate and its corresponding actual coordinate, we can decide that the stroke is a curved line. On the other hand, the stroke can be decided as a straight line if the difference between the two coordinates is nearly zero. Specifically we calculate maximum difference between expected coordinates and their corresponding actual coordinates in each segment. When the maximum difference exceeds a certain level, then the algorithm determines the segment is a curved line and a straight line otherwise.

In this way, each numerical digit has different curved line in number or position and that can be considered as a unique signature to recognize the numbers. Figure 4 shows how each numerical digit was represented by the CSLA algorithm. The upper row represents the raw strokes drawn on Tablet PC panel. The lower row shows the recognition process to find difference between the expected coordinates of imaginary straight line in a stroke (straight line drawn by red circles) and the actual coordinates of the stroke (actual stroke drawn by blue circles).

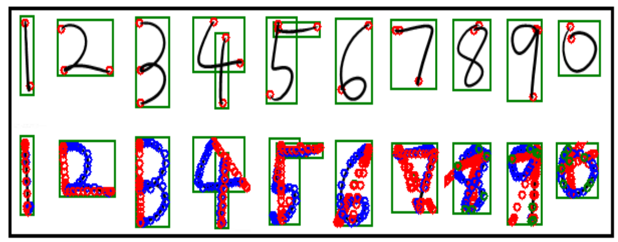

Figure 4. A screenshot of raw strokes and their recognition result

The CSLA first establishes a unique template for each digit that includes (1) the number of curved and straight line components, (2) the drawing sequence, and (3) the location of the curve or straight line components in one recognition area.

For example, the digit 1 is represented by a single vertical straight line. Thus it has a signature of one straight vertical line. The digit 2 has a curve line drawn above and a horizontal straight line below. Also the curved line of digit 2 is mostly drawn before than the horizontal straight line. Therefore the digit has the curved line above first and the horizontal straight line below next as its signature. 3 has a curve above another curve, 4 has the first left slanted curve with the next vertical straight line penetrate it. In this way we decided the unique signature of digits.

The recognition process for every stroke starts by comparing the templates from number 1 to number 9, and ending with number 0 sequentially. That is, while comparing with the template of number 5, it would already have eliminated the numbers from 1 to 4 as a candidate.

Assume that the recognizer is looking at number 2. It is consisted of one curved line in upper bound and one straight line in lower bound. It also checks the writing sequence of number 2 that the upper curved line mostly comes earlier than the lower straight line. The recognizer already knows that the stroke is not a number 1 and now it checks whether the stroke has the same signatures as which number 2 uniquely has. If the recognizer found matching unique signatures as those of number 2 from the stroke, it determines the stroke as a number 2, move on to check number 3 otherwise. Numbers 1 to 7 are recognized using this template- matching algorithm. Numbers 8, 9 and 0 can be recognized faster just by the coordinate of several sampling points alone. It is efficient and also it allows us to avoid unnecessary time-consuming comparison of many of expected and actual coordinates. When we look at the digits 8, 9, and 0, each of them has quite different shape than the others.

According to the recognition sequence of this algorithm, when it tries to compare the digit 8 and its sig-nature, it already eliminated 1 to 7 as a matching candidate. That means we only need to consider whether this digit is 8, 9, 0, or not a digit. We already mentioned that those numbers have clearly different shape so we can determine this number just using location of several sampling actual coordinates. For example, sometimes, a number 2 was misrecognized as a number 3 and a number 1 was misrecognized as a number 7 due to its similar shape. This means that it is hard to detect the unique signature of each number without using our CSLA algorithm. In short, this algorithm maximizes its recognition accuracy by detecting the unique signature of each number which is the combination and position of curved, straight lines, and writing sequence (which is presented above). In the next section, we will present the experiment result of our CSLA algorithm.

5. EXPERIMENT RESULTS

We conducted 20 trials to test the performance of our CSLA recognition algorithm. Each trial requires 15 digits to write from 1 to 12. The algorithm produced a high recognition accuracy of 97.67% (recognized 293 digits successfully out of 300) on average through the experiment as shown in Figure 5 and Figure 6 below.

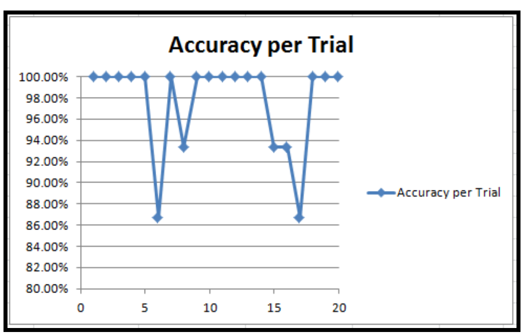

Figure 5. A recognition accuracy of each trial

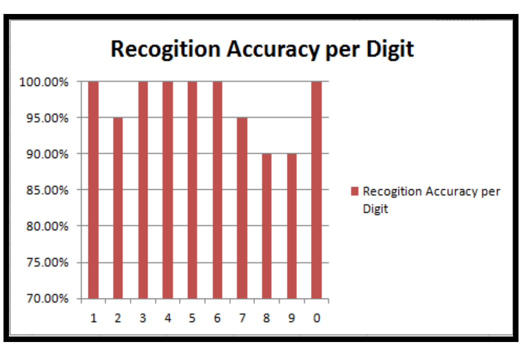

Figure 6. A recognition accuracy of each number

6. FUTURE DIRECTION

Although, we developed a decent recognition algorithm which shows high recognition accuracy, the validation of this algorithm has been conducted by the small number of people. In the future, we will conduct an experiment with a large number of participants from different ages. This will enable us to collect the information of variety of writing styles and validate whether this algorithm is still robust enough to the many different shape of digits. Moreover, the experiment may help us to determine that we need to add more criteria to recognize those digits such as writing direction for each component, more coordinate sampling, or statistical approach.

7. CONCLUSION

We have developed CSLA algorithm to enhance the recognition accuracy of MS SDK. CSLA compares the unique signature of each number by comparing every curved and straight line, and writing sequence of the stroke. Through the 20-time of experiments, we confirmed 97.67% of positive recognition accuracy on average which was around 70% produced by MS SDK recognizer. For the future, we will improve it by handling exceptions such as incorrect integration of rectangle recognition area.

8. REFERENCES

[1] Anthony, L., Wobbrock, J. A Lightweight Multi-stroke Recognizer for User Interface Prototypes, 245-252

[2] Freedman, M., et al. Clock Drawing: A Neuropsychological Analysis, Oxford: Oxford University Press, 1994

[3] Kim, H., Cho, Y., and Do, E. 2010. Context-Bounded Refinement Filter Algorithm: Improving Recognizer Accuracy of Handwriting in Clock Drawing Test, *In proceedings of Visual Representations and Reasoning of a workshop of the 24th Conference on Artificial Intelligence* (Atlanta, GA, July 11 – 15, 2010). AAAI '10, (2010), 53-60

[4] Libon, D., S. RA, et al. 1993. "Clock drawing as an assessment tool for dementia." Archives of Clinical Neuropsychology 8 (1993), 405-415.

[5] Santosh, K. C., Cholwich, N. 2009. A Comprehensive Survey on On-line Handwriting Recognition Technology and Its Real Application to the Nepalese Natural Handwriting. Kathmandu University Journal of Science, Engineering and Technology 5, 1 (2009), 31-55.

[6] Shilman, M., et al. 2006. CueTIP: A Mixed-Initiative Interface for Correcting Handwriting Errors. *In proceedings of the ACM Symposium on User Interface Software and Technology* (Montreux, Switzerland, October 15 – 18, 2006), UIST '06, 323-332

[7] Strauss, E., Sherman, E. M. S. and Spreen, O. A Compendium of Neuropsychological Tests: Administration, Norms, and Commentary. Oxford University Press, Oxford, 2006.

[8] Wobbrock, J. et. al. 2007. Gestures without Libraries, Toolkits or Training: A $1 Recognizer for User Interface Prototypes. *In proceedings of the ACM Symposium on User Interface Software and Technology* (Newport, RI, October 07 – 10, 2007), UIST '07, 159-168

[9] Yin, J., Sun, Z. 2005. An Online Multi-stroke Sketch Recognition Method Integrated with Stroke Segmentation, *In proceedings of the Conference on Affective Computing & Intelligent Interaction* (Beijing, China, October 22 – 24, 2006), ACII '05, LNCS 3784 (2005), 803-810