A System for Recognizing and Beautifying Low-level Sketch Shapes Using NDDE and DCR

Brandon Paulson
Sketch Recognition Lab
Texas A&M University
TAMU 3112
College Station, TX 77843-3112 USA
bpaulson@cs.tamu.edu

Tracy Hammond
Sketch Recognition Lab
Texas A&M University
TAMU 3112
College Station, TX 77843-3112 USA
hammond@cs.tamu.edu

ABSTRACT
Sketching has been identified as a natural means for human interaction and thus has become commonly incorporated into various user interfaces. Current low-level sketch recognizers have produced good accuracy but recognize only a small set of basic shapes. We propose a low-level sketch recognition and beautification system that uses a hierarchical approach that is capable of recognizing eight primitive shapes, along with complex fits, with preliminary recognition rates around 98.8%. These accuracy rates are comparable to current state-of-the-art recognition systems which recognize a lesser number of primitives. Furthermore, we introduce two new metrics, normalized distance between direction extremes (NDDE) and direction change ratio (DCR), which help aid in distinguishing between polylines and other low-level primitives.

Categories and Subject Descriptors: H5.2 [**Information interfaces and presentation**]: User Interfaces. - Input devices and strategies

Additional Keywords and Phrases: Human computer interaction, intelligent user interfaces, pen based computing, sketch recognition

INTRODUCTION
Sketch and gesture recognition have become important technologies in the user interface community, particularly with the use of Tablet PCs. It has been shown that gestures are typically easier to remember than basic textual commands [5], therefore tools have been developed that allow gestures to be easily incorporated into many user interfaces [3][7]. Simple low-level, single-stroke recognizers have already been developed using linear classification techniques [4][6]. However, a major downfall of using feature-based, linear classification is that these systems require training and are heavily user-dependent. The accuracy of these systems also becomes strongly determined by the given feature set, making recognition only as good as the chosen features.

Different sketch recognition systems use a more geometric and hierarchical approach, but face trade-off issues between the number of primitive shapes it is able to recognize and accuracy [1][8][9]. In order to create usable and accurate high-level recognizers it becomes imperative that the low-level recognizer be capable of identifying a variety of primitive shapes while still maintaining viable accuracy.

The goal of this work is to create a recognition system that is capable of recognizing a robust number of primitive shapes without sacrificing accuracy. We chose to use a geometrical approach; as such an approach does not require user training and is more robust to various drawing styles. Because we are simply creating a low-level recognizer, this work focuses on recognizing shapes that can be drawn using a single stroke. The shapes capable of being recognized include:

- Line: a stroke with a relatively constant slope between all sample points
- Polyline: a stroke consisting of multiple, connected lines
- Circle: a stroke that has a total direction close to 2π, constant radius between center and each point, and whose major and minor axes are close in size; can be overtraced
- Ellipse: a stroke with similar properties of a circle, but whose major and minor axes are not similar in size; can be overtraced
- Arc: a stroke whose shape is part of an incomplete circle
- Curve: a stroke whose points can be fit to up to a fifth degree curve
- Spiral: a stroke that can be seen as a series of circles with descending (or ascending) radii but a constant center
- Helix: a stroke that can be seen as a series of circles with similar radii but with a linearly moving center

The system uses a hierarchical approach to determine the best fit for a stroke while maintainig a list of other reasonable interpretations. We have conducted preliminary studies which compare our results against another commonly used low-level recognizer [8] and have seen promising results.

IMPLEMENTATION
Recognition of strokes is modeless, requiring no user interaction other than drawing the stroke itself. Strokes are typically input using pens on Tablet PCs, but can also be captured using basic mouse input. Strokes begin with a pen/mouse down event and end with a pen/mouse up event.

Our recognizer begins by first performing a series of pre-recognition operations on the input stroke. During pre-recognition, a series of values and graphs are computed for the stroke, in-

UIST '07, October 7–10, 2007, Newport, Rhode Island, USA
ACM 978-1-59593-679-2/07/0010

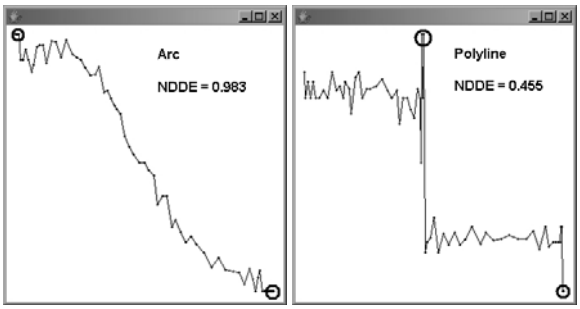

Figure 1: Direction graphs showing minimum and maximum direction values for an arc (left) and a polyline (right). While polylines may not necessarily always have an "average valued" NDDE, curved shapes such as an arc should always have high (or low) NDDE values since the maximum and minimum directional values should always be near the stroke's endpoints.

cluding number of corners, direction graph, speed graph, and curvature graph. These values are computed using methods similar to those found in [8][9].

In addition to these values, two new values which are unique to this work are computed. The first value is the normalized distance between direction extremes (NDDE). This value is used to access the extremity of the maximum and minimum direction values and is computed by first taking the difference between the lengths traveled in the stroke at the indices of the maximum and minimum direction values. This difference is then divided by the stroke length and is helpful in distinguishing between polylines and curved shapes.

The second value is the direction change ratio (DCR) which is computed as the maximum change in direction divided by the average change in direction. Like NDDE, we use this value to help distinguish polylines or shapes with distinct corners (see Figure 1). These two metrics are used not only in each of our shape recognizers, but are also used in our hierarchy function to help sort interpretations.

Once all pre-recognition values have been computed, the stroke is then sent to various low-level shape tests. Each of these tests return two important pieces of information: a Boolean flag stating whether the test passed or failed and a beautified shape object (typically an extension of Java2D shapes) that best fits the input stroke. Once the tests for all possible shapes are executed, the results of each test are then sent to a hierarchy function which sorts the interpretations in order of best fit. Such a function is necessary because each shape test returns an error value that may not be on the same scale as other tests. While most of the primitive shapes use common error metrics like least squares error and feature area, others like spiral and helix use a different set of error tests. Because the difference between a drawn spiral or helix and their beautified counterpart is typically quite extreme, least squares and feature area errors are not as helpful for these shapes.

PRELIMINARY RESULTS

After building our recognizer, we tested it by collecting a total of 450 shapes from 5 different users. Each user was asked to draw 10 examples of each one of the eight primitive shapes, along with 10 examples of a complex shape. In order to test the accuracy of complex shape approximations, we asked users to draw a complex shape consisting of one line and one arc, an example which some recognizers have difficulty in interpreting [9]. For our experiment, we wanted to test how often the correct interpretation was among the listed interpretations, as well as, how often the correct interpretation was the top or best interpretation. We found that the correct interpretation was present among the list of interpretations 99.8% of the time. The average size of this interpretation list was 2.99 with polyline always returned as one of the interpretations. The correct interpretation was found to be the best interpretation (as determined by the hierarchy) 98.8% of the time.

CONCLUSION/FUTURE WORK

We have described a low-level sketch recognition and beautification system capable of recognizing eight primitive shapes, along with complex shapes, with accuracy rates close to 98.8%. We have integrated our low-level recognizer into a higher-level sketch recognition system, LADDER [2], which has allowed us to do preliminary testing. We have seen promising results for high degree complex fits; however, it has yet to be formally tested and evaluated.

ACKNOWLEDGMENTS

This research was supported in part by CISE/IIS.

REFERENCES

1. Arvo, J. and Novins, K. Fluid sketches: continuous recognition and morphing of simple hand-drawn shapes. In *Proceedings of UIST '00* (November 6–8, San Diego, CA), ACM, NY, 2000, 73–80.

2. Hammond, T. and Davis, R. LADDER, a sketching language for user interface developers. *Computers & Graphics 29*, 4 (2005), 518–532.

3. Long, A.C., Landay, J.A., and Rowe, L.A. T̈hose look similar!̈ issues in automating gesture design advice. In *Proceedings of PUI '01* (November 15–16, Orlando, FL), ACM, NY, 2001, 1–5.

4. Long, A.C., Landay, J.A., Rowe, L.A., and Michiels, J. Visual similarity of pen gestures. In *Proceedings of CHI '00* (April 1–6, The Hague, The Netherlands), ACM, NY, 2000, 360–367.

5. Morrel-Samuels, P. Clarifying the distinction between lexical and gestural commands. *International Journal of Man-Machine Studies 32*, 5 (1990), 581–590.

6. Rubine, D. Specifying gestures by example. In *Proceedings of SIGGRAPH '91*, ACM, NY, 1991, 329–337.

7. Saund, E., Fleet, D., Larner, D., and Mahoney, J. Perceptually-supported image editing of text and graphics. In *Proceedings of UIST '03* (November 2–5, Vancouver, B.C., Canada), ACM, NY, 2003, 183–192.

8. Sezgin, T.M., Stahovich, T., and Davis, R. Sketch based interfaces: early processing for sketch understanding. In *Proceedings of PUI '01* (November 15–16, Orlando, FL), ACM, NY, 2001, 1–8.

9. Yu, B. and Cai, S. A domain-independent system for sketch recognition. In *Proceedings of GRAPHITE '03* (February 11–14, Melbourne, Australia), ACM, NY, 2003, 141–146.