Project 1 – Exploring $-family recognizers

Due: 2/8/24 11:59 pm

The purpose of this assignment is twofold. It is first designed to give you experience with simple recognizers using reliable data (touch/stylus/mouse). Second, it is to give you experience with implementing a state of the art algorithms from scratch by relying on algorithms provided in the research paper

Requirements

There are two main requirements for this assignment. First, you will implement 3 $- family recognizers and compare them. The first is the $N recognizer discussed here:

Anthony, L. and Wobbrock, J.O. (2010). A lightweight multistroke recognizer for user interface prototypes. Proceedings of Graphics Interface (GI '10). Ottawa, Ontario (May 31-June 2, 2010). Toronto, Ontario: Canadian Information Processing Society, pp. 245- 252.

The second recognizer is known as Protractor and details can be found here:
Li, Yang (2010). Protractor: a fast and accurate gesture recognizer. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10). ACM, New York, NY, USA, 2169-2172, April 2010.

The third recognizer is known as Penny Pincher and the details can be found here:
Taranta, E. and LaViola, J. Penny Pincher: A Blazing Fast, Highly Accurate $-Family Recognizer, Proceedings of Graphics Interface 2015, 195-202, June 2015.

You should test these algorithms with the following symbols: 0,1,2,3,4,5,6,7,8,9,+,-,*,t,a,n,s,c,i, and the square root symbol. I would also recommend adding some more oddball symbols of your own choosing

Train each recognizer with 1, 3, and 5 samples per symbol. Test each recognizer by writing each symbol 5 times, which should give you a good accuracy number. You should run separate experiments using the data you collected using a mouse vs a touch screen (if applicable). Please put the results of your experiment in your README file.

Strategy

The first step of this project is to create some kind of basic canvas drawing application. This can be done in any programming language of your choosing, though I recommend looking into the capabilities of C# with WPF, Python with TK, Java with Swing, or HTML/JS/CSS. No choice is inherently better than others. You will want to create buttons or widgets to enable the user to clear the canvas, trigger recognition, and undo previous written strokes. Please see

https://www.eecs.ucf.edu/isuelab/demo/stochastic-resampling/ or https://depts.washington.edu/acelab/proj/dollar/index.html for examples of similar canvas implementations which you can work from. Reach out to the instructor if more examples are needed.

To implement your symbol recognizers, there are some things you need to consider.

1. You need to find a way to invoke the recognizer. You can have it run in real time or in batch mode (by pressing a button or doing a different, heuristically determined gesture)

2. You will need to show recognition results to the user. A simple text box is fine but if you want to be more elaborate feel free to do so.

Deliverables

Upload your source code to Canvas. Please include a README file describing what works and what does not, the results of your experiments, any known bugs, and any problems you encountered

Grading

Grading will be loosely based on the following:

80% correct implementation of the recognizers
20% documentation