Aiding Human Discovery of Handwriting Recognition Errors

Ryan Stedman, Michael Terry, Edward Lank
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, Ontario, Canada
{rstedman, mterry, lank}@cs.uwaterloo.ca

ABSTRACT
Handwriting is becoming an increasingly used method of input into computer systems, but as handwriting is an inherently ambiguous form of input, there will always be some degree of error in handwriting recognition systems. Most approaches assume input and recognition occur simultaneously, making human error detection simultaneous with input. This is not always the case; even less so with the use of digital pens, which introduce a delay between when a user enters the information and when they are able to validate the correctness of the recognition. We introduce CorrectionPanels, an interface which provides a series of techniques designed to aid users in discovering and correcting handwriting recognition errors in form-filling applications.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

General terms: Design, Human Factors.

Keywords: error discovery, pen computing.

INTRODUCTION
As human handwriting is an inherently ambiguous form of input, handwriting recognition systems will always contain some degree of error. Because of this, there is a clear need for a human medium between the recognizer and the acceptance of the recognized text to be able to catch the errors in recognition. Most systems rely on the fact that they are able to display recognized text as the user inputs handwriting [4], allowing the user to validate the recognition as they write. However, technologies such as the Anoto pen [1] introduce the possibility of a delay between when user enters the information and when they are able to validate it, making in the moment validation unusable.

Previous solutions to this problem have visualized recognizer metadata, such as confidence values and alternative recognitions, to aid in the discovery of errors [4, 3]. However, although this approach can help in discovering errors, since the recognizer is prone to error, any additional information from it would also be error prone, making these techniques unreliable. An alternate approach has been to automate the error discovery process instead of relying on the user input [2, 4], but these methods are also prone to error, and can not be used as the only source of error detection.

We present our ongoing work on CorrectionPanels, an interface which provides a series of techniques with a variety of features designed to aid users in discovering handwriting recognition errors, and correct them in a form-filling application. CorrectionPanels do not rely on recognizer metadata, or any automated techniques, instead providing a number of techniques to re-render how the recognized text is displayed to the user to help the discovery of errors, as well as providing aids to make the task of verification simpler for the user.

DESIGN
Each CorrectionPanel consists of a textbox, which contains the recognized text for the ink associated with that panel, as well as an area displaying the ink. The following is a list of the features implemented into CorrectionPanels to aid users in discovering handwriting recognition errors, divided into verification aids and re-renderings of recognized text. Verification aids are features that improve a user’s ability to view and mark both the recognized text and the handwritten ink, while re-renderings of recognized text alter the way in which recognized text is presented to the user to help them discover errors in recognition.

Re-renderings of Recognized Text
Emphasizing Character Classes with Typography Recognizers will sometimes recognize a type of character (letter, number or punctuation), where no characters of that type were written. This often will stand out to a user, as the inclusion of that character type does not make any logical sense where it is. CorrectionPanels allow users to enlarge the font size of a specific character type, causing that character type stand out so that it can be easily identified and removed from misrecognized fields that it does not belong in (Fig. 1).
Figure 2: A user creates an expectation of what to hear when reading the ink. A recognition error will be spoken as something other than expected, signaling an error.

**Text-to-Speech** The use of a text-to-speech engine is provided to read the text of that CorrectionPanel. When activated, the ink of the CorrectionPanel currently being spoken is highlighted red, allowing the user to determine which panel is being spoken when a number of panels are spoken serially (Fig. 2). Preliminary use has shown that this can be used to great effect in finding recognition errors, as a user seems to create an expectation of what they hear as they read the ink while it is being spoken. If a word is spoken that goes against that expectation, as would happen in the event of a recognition error, it stands out to the user as incorrect. Coupled with the ability to tag panels for later correction, described further on, a user can quickly listen through an entire form, marking fields off that they think may contain error for later inspection and correction.

**Spellcheck** Though recognizers do not normally produce spelling errors, as they will have a strong preference toward words in their dictionaries, there are some situations in which dictionary words will not fit the handwriting, and a spelling error may occur which could be missed by a user. CorrectionPanels automatically place red underlines under spelling errors, as is common in word processors, to bring a user’s attention to them.

**Verification Aids**

**Viewing Ink and Text** Although it would be optimal to display both the handwritten ink and the recognized text to allow users to quickly compare the two, many times this is not possible due to screen real-estate constraints. Instead, only the text or the ink can be displayed at one time, though each panel can be ‘flipped’ back and forth to display one or the other.

**Chunking Correction Tasks by Tagging** To avoid users from having to switch back and forth from the ink and text views when using the ink view to discover errors, while viewing the ink a user can tag the panel for later correction by clicking on the ink, causing the text field to be highlighted.

**Ink+Text Tooltips** While being able to switch back and forth between the ink and text allows both to be viewed, and is useful for some error discovery methods, it does not enable quick comparison. To allow quick comparison, each panel contains a tooltip showing the information that is not currently visible. When the text is visible, the tooltip displays the ink, and when the ink is visible the tooltip displays the text (Fig. 3).

**Unambiguous Font** Many of the common fonts used to display text in computer programs contain ambiguities between specific character types; for example, the capitalized letter ‘O’ and the number zero. This problem is made more prominent when writing non-words that have numbers and characters mixed together, such as course codes (ex. CISC101). To address this problem, CorrectionPanels use a programmer’s font; a type of font designed to resolve any ambiguities between characters.

**IMPLEMENTATION** CorrectionPanels have been implemented as a custom control for C#, using the .NET 3.0 framework. Additionally, an enumerator has been implemented to smoothly handle interactions with multiple instances of CorrectionPanels, such as having the text-to-speech technique used serially.

**CONCLUSION AND FUTURE WORK** We have described the design of CorrectionPanels, an interface that allows handwritten ink and the recognized text of that ink to be displayed in a variety of ways meant to aid users in the discovery of recognition errors. While there are a number of techniques we wish to add to the design of CorrectionPanels, the next step is to evaluate the effectiveness of the techniques currently used. We are currently in the process of evaluating the effect that the techniques presented here have in aiding users to discover handwriting recognition errors.

**REFERENCES**