Aiding Human Discovery of Out-of-the-Moment Handwriting Recognition Errors

by

Ryan Stedman

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Mathematics
in
Computer Science

Waterloo, Ontario, Canada, 2009

Abstract

Handwriting recognizers frequently misinterpret digital ink input, requiring human verification of recognizer output to identify and correct errors, before the output of the recognizer can be used with any confidence int its correctness. Technologies like Anoto pens can make this error discovery and correction task more difficult, because verification of recognizer output may occur many hours after data input, creating an "out-of-the-moment" verification scenario. This difficulty can increase the number of recognition errors missed by users in verification. To increase the accuracy of human verified recognizer output, methods of aiding users in the discovery of handwriting recognition errors need to be created. While this need has been recognized by the research community, no published work exists examining this problem.

This thesis explores the problem of creating error discovery aids for handwriting recognition. Design possibilities for the creation of error discovery aids are explored, and concrete designs for error discovery aids are presented. Evaluations are performed on a set of these proposed discovery aids, showing that the visual proximity aid improves user performance in error discovery. Following the evaluation of the discovery aids proposed in this thesis, the one discovery aid that has been proposed in the literature, confidence highlighting, is explored in detail and its potential as a discovery aid is highlighted. A technique is then presented, complimentary to error discovery aids, to allow a system to monitor and respond to user performance in errors discovery. Finally, a set of implications are derived from the presented work for the design of verification interfaces for handwriting recognition.

Acknowledgements

First, I would like to thank my supervisor, Edward Lank, for his guidance and support throughout my time as a graduate student at Waterloo. I have learned a great deal from him, and because of him I have accomplished a lot in my short time here. I would also like to thank Michael Terry, who has played a big role in shaping the work presented here. Without both of their guidance and insight, this work would not exist. They also deserve thanks for creating an environment in the HCI lab where everybody can turn to each other for ideas and support. I would like to thank all the present and past members of the HCI lab during my study. Without exception, every person in the lab was always willing to provide advice, assistance, and encouragement when needed. Outside of the lab, I would like to thank Stacey Scott for the great feedback she provided on this thesis.

My parents, Larry and Becky Stedman, have always supported me in my goals, and in my graduate studies this has been no different. They have always been there with encouragement and support. My sisters, Trish and Katie, have also been supportive of me throughout my time here. Without the support of my family, I would never have been able to make it to where I am.

Finally, I would like to dedicate this work to my girlfriend, Kait, who has seen this through with me from beginning to end. Without her support and encouragement, I doubt that I could have accomplished as much as I have, and I definitely would not have had as much fun.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

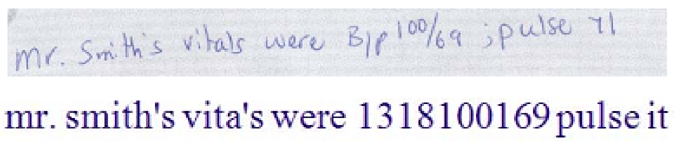
Because handwriting is an ambiguous form of input, no matter how good handwriting recognizers get they will always be prone to error in their recognition results. For handwriting recognition to be usable for applications where a high input accuracy is needed, such as in medical or research fields, user verification of recognizer output is necessary. However, humans are also error prone, and are likely to miss recognition errors in a verification task. This thesis looks at the problem of aiding humans in the verification of handwriting recognition results.

1.1 Motivation.

Handwriting recognition is used in a variety of systems as a means of natural user input. Applications range from personal note taking, such as with Microsoft Journal [13], to recording patient medical information in a health-care setting, such as with FusionForm Desktop [15]. Depending on the application, the potential severity of errors in recognition changes.

In a personal note-taking application, the information that is being recognized is meant for personal use. While errors in recognition still have some potential for some harm, for example a misrecognized appointment date, the non-critical nature of the input makes the risk of error acceptable.

In a health-care setting, however, the risk of error is much less acceptable, as errors can have a much more serious impact. For example, changes in a patient's vital signs can be an early indication of health problems, and can be used to start preventative treatment before the patient's condition worsens. If a patient's vitals are recorded into a system using handwriting recognition, there is the potential for a recognition error to occur on an irregular vital sign. Later review of that patient's vital signs will miss the early indicator of a health problem, which can seriously impact the future health of that patient. This health-care scenario was the original motivation for the work presented in this thesis.



mr. Smith's vitals were B/p 100/69 ; pulse 71

mr. smith's vita's were 1318100169 pulse it

Figure 1.1: A handwriting sample from a practicing nurse, with the result from the handwriting recognizer used with the Anoto pen underneath.

A solution to the issue of errors in handwriting recognition is to enable users to verify and correct recognizer output. However, this will not always ensure that all, or even most, of the errors in recognition get corrected. In a health-care setting, medical professionals are often very busy and pressed for time, and in these conditions it can be easy for a user to rush over verification and miss critical errors. This situation can be worsened if that user is fatigued, for example if they are at the end of a 12-hour shift, making them less careful and more prone to missing errors in verification.

Additionally, in health-care settings a majority of the medical information is input into forms, referred to as “charting” by medical professionals. This provides a clear separation of the information gathered for easier analysis, as well as acting as a reminder for what information needs to be collected. However, verifying highly structured data can quickly become repetitive as the amount of input that needs to be verified increases. This can cause users to become bored or habituated and stop paying close attention to the verification task, missing errors as a result.

There are a number of factors which can reduce the reliability of human verification, increasing the amount of residual error left behind by a user. However, creating an interface which aids a user in discovering errors can help reduce the amount of residual error left by a user, as well as reduce the impact of confounding factors on error discovery.

1.2 Error Discovery in Handwriting Recognition

Handwriting recognition systems are currently able to achieve accuracies greater than 90%. While this is a fairly high accuracy, it is not perfect. As the amount of input that a recognizer must interpret increases, the amount of errors that can be expected in the output also increases. Moreover, handwriting is an ambiguous form of input, and as such recognizers will never acheive perfect accuracy, regardless of how close they come.

Acknowledging the issue of errors in handwriting recognition, five key research areas have been defined for the handling of errors in recognition-based interfaces [12]. These are: *error reduction*, which seeks to improve recognizer performance; *error discovery aids*, which seeks to allow easier identification of errors; *error correction techniques*, which seek to speed user correction of recognition errors; *validation* *of techniques*, which studies the effectiveness of the previous research directions; and *toolkit level support*, which simplifies incorporation of recognition error handling techniques into applications. All five of these areas of research are important for creating recognition interfaces which can make reliable use of handwritten input, but existing research does not reflect this.

The primary goal of the majority of researcher in handwriting recognition has been in the first described area for error handling research: error reduction. Researchers study both off-line (scanned image) and on-line (stroke-based) algorithms, with the specific goal of improving the reliability of recognition algorithms (for a survey, see [14]). Another area of error handling research that has seen a significant amount of work in has been techniques for correcting errors. Previous work has looked at reducing the amount of input necessary from a user to correct an error, for example by providing access to n-best lists [4], as well as supporting the correction task when using stylus input. However, users still need to discover any errors in recognition output before they are able to perform any correction, a problem not represented in existing research. *This thesis explores how to design error discovery aids: techniques which can be used to aid users in identifying handwriting recognition errors.*

There are a number of factors which can affect the verification task that need to be taken into account when considering the problem of error discovery: Whether recognition is performed *in-the-moment or out-of-the-moment*, how the handwritten input is *structured*, *how much recognizer output a user must verify at a time*, if there are any *time constraints*, and whether or not the *user verifying is the user who wrote the input.* These factors are described in more detail below.

- *In-the-Moment vs. Out-of-the-Moment.* There are two different scenarios in which a user can view and verify the results of a handwriting recognizer's output. In-the-moment, where they can view the recognizer output as they write, and out-of-the-moment, where there is some delay in time between writing and when recognition can be performed and verified by a user. In an in-the-moment recognition scenario, possible with devices such as Tablet PCs, a user can verify the correctness of recognition while the intended input is still fresh in their mind, making the task of discovering errors simple. On the other hand, in out-of-the-moment recognition a user may no longer recall what the intended input was. This changes the nature of the verification task from one of verifying that the recognizer output fits the intended input, to a verification that the recognized output matches the handwritten input, a task which requires more cognitive overhead and time to complete.

- *Structural constraints on input.* There are varying levels of constraint that can be imposed on the structure of handwritten input: from completely unstructured, such as with sketch and diagram recognition; to partially structured, such as with free-form input of handwritten text, where the input follows certain constraints imposed by the language being written, yet has little constraint on the information being input; to fully structured, such as with form-filling, where the structure of the input, as well as the type of input, is constrained based on the individual form fields. Unstructured input faces the problem of recognition errors due to recognition performed on input that is not handwriting. Partially constrained input has the benefit of some context, where a user may notice a recognition error in the output due to the fact that it does not make any logical sense with regards to the surrounding context of the sentence. However, additional difficulties arise in error discovery when a large amount of input is recognized, where a user may miss errors in recognition, because of the volume of recognition to verify. Fully structured input gains even more contextual benefits, as misrecognized results may not fit in where they are recognized, such as a number in a field where only letters are expected. On the other hand, it can become monotonous to verify, as users may lose focus during the verification of large amounts of closely similar input.

- *Amount of verification.* The user may only be verifying the correctness of a small amount of recognized data, or they may need to verify a large quantity of output. As the amount of recognized data that a user must verify grows, and therefore the amount of time spent on the verification task increases, it may cause boredom, habituation or fatigue in users, causing them to miss errors.

- *Time constraints.* Any time constraint imposed on a user in the verification task may impact that user's performance in error discovery. With no time constraint a user can take their time in verification, making sure that no errors are left in the recognized results. With a time constraint a user may be forced to rush through the verification task, missing errors in recognition because they are being less careful in verification. The tighter the time constraint, the more errors a user is likely to miss.

- *User Verifying.* Whether or not the user who is verifying is the user who originally wrote the ink can impact the verification task. If the user verifying the output is not the user who originally wrote the input, errors could be missed due to misinterpretation of the original user's handwriting. With the original user verifying the output this problem is lessened, however people can sometimes misinterpret their own handwriting.

To keep this problem manageable for the purpose of this thesis, the problem of aiding error discovery has been scoped to the following three constraints:

1. *Out-of-the-moment*: The work presented in this thesis looks at the verification process as an out-of-the-moment problem, meaning that there is some delay in time between when the user writes the information and when they are able to view and verify the recognition results.

2. *Form-filling*: The verification scenario looked at will be focused on input into a multiple-field form.

3. *Batch Verification*: Finally, the verification problem will include a user having to verify a batch of documents, rather than just one.

Most of the work presented in this thesis assumes some kind of time constraint on the verification task, and that the user verifying is the user who input the handwriting, though the error discovery aids presented don't necessarily address these issues.

1.3 Goals

In this thesis I will attempt to address the problems stated above with the following goals:

Goal 1. **To explore different design possibilities for the creation of error discovery aids**. Due to the lack of prior work in aiding users in error discovery, only a single design exists for an error discovery aid (confidence highlighting) [12]. This lack of prior designs leaves little which can be used to inform further designs of error discovery aids. To achieve this goal, a design space was created which describes the different dimensions which can be used in the design of an error discovery aid. From this design space, a number of designs for error discovery aids were created.

Goal 2. **Identify methods which can be used in a verification interface to aid users in error discovery, reducing the number of residual errors – errors that remain after a user verifies results**. With an understanding of the possible designs for error discovery aids (Goal 1), the next step is to identify which error discovery aid designs provide real benefit in aiding users to discover errors. Evaluations were performed on the designed error discovery aids, as well as on the existing confidence highlighting discovery aid to achieve this goal. Additionally, a method is explored to allow the system to monitor and respond to user performance in error discovery to reduce residual error.

1.4 Contributions

This thesis contributes the following original ideas and knowledge to the field of Human-Computer Interaction.

1. *Design space for error discovery aids.* A design space is presented for the creation of error discovery aids for handwriting recognition.

2. *Design and evaluation of seven error discovery aids.* Designs for seven error discovery aids are described, and later implemented into verification interfaces and evaluated.

3. *Visual proximity technique.* Of the seven designs described, the visual proximity technique is shown to provide a significant benefit in users' performance discovering errors in handwriting recognition.

4. *Technique to measure performance of a user in error discovery.* A method is presented which can be used to allow a verification interface to keep track of a user's performance in error discovery during verification.

5. *Data on Microsoft's handwriting recognizer confidence values.* An evaluation is presented providing data on the accuracy of the recognizer's confidence values, which can be used to inform the use of the confidence highlighting discovery aid.

1.5 Organizational Overview

This thesis is divided into 8 chapters:

Chapter 2 presents background to the problem, and presents previous work. Motivation to the problem is presented with regard to the state of handwriting recognition systems. The problem is then broken down to explain the differences between in-the-moment recognition and out-of-the-moment recognition, and how the problem is more significant in the latter. Existing research on aiding users in error discovery is then presented.

Chapter 3 presents an exploration of design possibilities for the creation of error discovery aids. A design space is first laid out which describes the different possible factors that could be made use of in the creation of error discovery aids. A number of designs for error discovery aids are then described.

Chapter 4 provides an evaluation of the error discovery aids presented in Chapter 3. First, an exploratory evaluation is presented which was used to guide and refine the designs presented in Chapter 3. Then, an experimental evaluation to test the effect of three of the discovery aids on user performance in verification is presented against a control condition, and the results are discussed.

Chapter 5 explores the potential of the confidence highlighting technique, which has been suggested as an error discovery aid in prior work, and examines in which scenarios it would be appropriate to use as an error discovery aid. An evaluation of confidence values from the Microsoft handwriting recognizer is presented, and the implications from the results on the use of confidence highlighting as an error discovery aid is discussed.

Chapter 6 presents a method of injecting artificial errors into recognition results for the purpose of measuring the performance of a user in an error discovery task.

An experiment and its results are described which evaluates this method of measuring performance. The chapter concludes with a discussion of the results of the experiment.

Chapter 7 discusses the implications for the design of an error discovery interface which can be derived from the work presented in previous chapters. First, it examined when the use of error discovery aids is actually necessary. Next, the dangers of attempting to guide a user's attention toward potential errors are discussed. Results of the formal evaluation of the visual proximity technique, and its implication on the use of proximity and availability are discussed. Last, the performance measurement from Chapter 6 is discussed, and its possible uses in verification interfaces are examined.

Finally, Chapter 8 concludes this thesis by summarizing how the research goals were achieved. The contributions made by this work are re-visited, and areas for future work are suggested.

Chapter 2

Background & Related Work

This chapter provides background information on the problem of aiding users in error discovery for handwriting recognition, and an overview of related work. The problem of error discovery is discussed in an in-the-moment recognition scenario. The problem of error discovery is then looked at with respect to an out-of-the-moment recognition scenario, and the differences and difficulties between the two scenarios are discussed. Finally, a review of related research in the problem domain of error discovery for handwriting recognition is presented. First, though, the state of handwriting recognition systems are examined, and the significance of errors in recognition results, and how they are handled, is highlighted.

2.1 Errors in Handwriting Recognition

There are two classes of handwriting recognition systems: on-line and off-line recognizers. Recognition researchers define these categories based on the input provided to their algorithms. On-line systems receive stroke information for input, while off-line systems receive scanned images [14]. Within these two classes there are a number of different strategies used in the recognition algorithms between different recognizers, but these two classes describe the recognition scenario these systems are used in, which in turn affect the amount of error in their recognition results.

On-line handwriting recognizers obtain handwritten input through some form of digital pen device, such as any device which uses stylus input. Because the input from the pen is directly recorded by a computational device, attributes of the user's handwriting are available to the recognizer instead of just the image of the handwriting, such as stroke segmentation, the order in which strokes are input, as well as speed and timing information. All of this information, as well as the handwritten input itself, can be used to help recognize the input [14]. Because of the extra information available to on-line handwriting recognizers, their recognition accuracy is better on average than off-line recognizers. Academic evaluations show that the accuracy of on-line handwriting recognizers can vary between 87-96% [5, 9].

Off-line handwriting recognizers are used to obtain recognition results from images of a user's handwriting, for example, performing recognition on a scanned image containing handwriting. The accuracy of off-line recognizers are much lower than that of on-line, with academic evaluations acheiving recognition rates between 50-80% [7, 21, 25], depending on the experimental setup.

While reducing these error rates is a valuable goal, it has proven elusive in practice. Some of the major contributing factors to the error rates observed are a result of differences in writing styles between users. Geometric variations in writing – that is, the differences between users in the position, baseline orientation and slant of their handwriting – adds to the complexity of recognition, increasing the possibility of error [17]. Allograph variations for individual characters – that is, differences between users in how individual characters are drawn – are another source of complexity in recognition, and can cause errors [17]. On-line recognizers often make use of the sequence in which strokes are added when performing recognition, causing any sequencing variations of strokes between users to cause difficulties for the recognizer. Neurophysical and biomechanical variations, physical differences between users which affect their ability to write, can affect the quality of handwriting [17], making recognizers less robust for certain users.

The accuracy of academic evaluations of handwriting recognizers show relatively good recognition accuracy rates, but there are a variety of other errors that occur in real world usage that are generally not represented in academic evaluations. Shomaker [16] lists a variety of factors that can produce errors not represented in academic evaluations: discrete noises, canceled material, badly formed shapes, material legible by humans but not the algorithm, misspelled words, words not in the recognizer's dictionary and device generated errors. Additionally, most recognizers do not deal with the problems of post-hoc editing, correction of spelling errors and letter insertion or deletion [14].

As a result of the differences in handwriting between users, handwriting is an inherently ambiguous form of input. Humans will often misinterpret other peoples' handwriting, and will occasionally misinterpret their own handwriting. Because of this ambiguity in human handwriting, modern handwriting recognition systems produce significant rates of error in their output. While there is still a lot of work that can be done to improve the accuracy of handwriting recognition, because of this inherent ambiguity no recognizer will achieve perfect accuracy; they will always contain some degree of error. For handwriting recognition to be used in any system where a high input accuracy is necessary, methods need to be in place to enable handling of errors produced.

As we noted in the introduction, Mankoff and Abowd [12] have defined five areas of research for the handling of errors in recognition-based interfaces:

1. *Error reduction.* Reducing the number of errors produced by a recognizer, in other words increasing recognition accuracy, decreases the amount of errors that need to be handled by a user or system. Researchers in the fields of

document analysis and recognition, as well as pattern recognition, produce extensive research toward this goal (for examples, see [26, 5]).

2. *Error discovery.* An error must first be discovered by either the system or a user before that error can be corrected. Techniques can be made to allow the system to discover errors, or aid users in the discovery of errors, which is the focus of this thesis.

3. *Error correction techniques.* Error correction techniques are used to support users in the correction of discovered errors. The default method of error correction would be re-entry of the desired input, however methods exist to reduce the amount of work necessary, for example by providing access to n-best lists [4], as well as providing correction mechanisms with the use of pen input, which can be seen in the Cue-TIP interface [18].

4. *Validation of techniques.* To ensure that techniques in the previous areas of research are providing positive benefits for error handling, they need to be validated. Most research produced in any of the previous three areas contain some form of validation for any new technique proposed.

5. *Toolkit level support.* Toolkits with validated methods of handling errors can simplify the integration of these methods into a handwriting recognition interface, ensuring that these error handling methods are properly implemented. An example of this can be seen in OOPS, a toolkit for error correction techniques in a recognition interface [11].

For the output of handwriting recognizers to be used with a high degree of confidence in the accuracy of their results, successful methods of handling errors in recognition are needed. This has been recognized by the research community, as research exists in the areas of error reduction, error correction techniques, validation of techniques and toolkit level support. Yet no research exists on aiding users to discover errors in handwriting recognition. The reason for this omission in existing research is that the problem of handling errors in handwriting recognition has been looked at primarily in the scope of in-the-moment recognition.

2.2 In-the-Moment Recognition

The most common approach to error discovery is to rely on the user to indicate an error in the recognized text, without any system output to assist in the discovery process [12]. In fact, almost no research exists on error discovery aids in the verification of handwriting recognition results. Even in related fields such as Optical Character Recognition, where residual error rates and human correction are frequently highlighted as motivators for improved recognition, no research has been found that studies how best to support the error discovery task during human verification of recognition results.

In handwriting recognition, one reason for this lack of research may be that most systems have been designed for "in-the-moment" verification of recognition results, where a user is able to see the output of the recognizer as they write and correct recognition errors as they occur. The input panel of a Tablet PC is a typical instantiation of this in-the-moment recognition process, seen in Figure 2.1. As the user writes, the recognized text is displayed above or below the ink input and any errors can be corrected immediately. In-the-moment recognition is only possible using on-line recognition systems, as off-line systems necessitate a delay between handwriting and recognition.

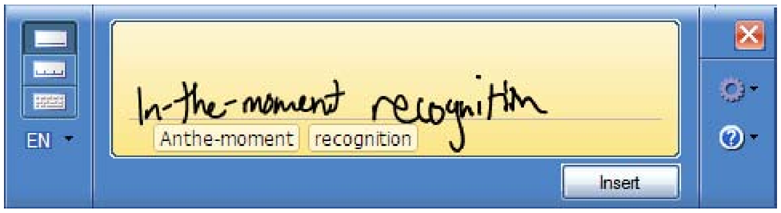


Figure 2.1: The pen input panel on a Windows XP Tablet PC. An example of in-the-moment recognition

In a system where in-the-moment recognition is used, the problem of getting a user to notice and identify an error is simplified. By being able to quickly and easily view the output of the recognizer while a user is still writing, and while the intended input is still fresh in the user's mind, discovering errors in recognition becomes a trivial task. In an in-the-moment recognition system, the more significant problem is instead providing methods methods for correcting errors using pen input. However, not all handwriting recognition systems provide in-the-moment verification of recognition results, making the problem of discovering errors much more difficult for the user.

2.3 Out-of-the-Moment Recognition

In-the-moment verification of recognition results is not always possible. Devices such as Tablet PCs, which can afford on-line in-the-moment recognition, often have applications that apply recognition to handwriting some time after the user has written the input. This behavior can be seen in Microsoft Journal [13], where a user may write a large amount of text and then apply a recognizer hours or days later. Although in-the-moment recognition is possible, it may not always be appropriate. For example, a user may be using Journal to write notes and ideas for a paper they are writing. While most of the input is unnecessary to perform recognition on, they may wish to copy parts of what they wrote into their paper, instead of having to transcribe from one source to another. In this scenario, recognition is only needed on select parts of input, making the use of in-the-moment recognition inappropriate for the entire input.

Delay between writing of input and recognition is a characteristic of off-line recognition algorithms. With off-line recognizers, a user must write the entirety of the input that is to be recognized before they are able to scan that input into the system and perform recognition. Examples of use can be seen in companies which process large amounts of hand-filled paper forms from customers, such as banks and insurance companies.

Other technologies exist for on-line handwritten input which create a similar recognition scenario to off-line recognizers, in that a delay between input and recognition is required. Examples of this are the Anoto [1] and Livescribe pulse [8] digital pens, both which allow users to record their handwriting with use of a digital pen and specially patterned paper, and download the recorded handwriting to a computer at a later time. With these devices a user writes some amount of handwritten input where no display is available to verify and correct recognition results, instead having to download the recorded input information from the device to a computer to perform recognition on, and then to verify and correct the results.

This delay between writing and recognition creates an "out-of-the-moment" recognition and verification problem, rather than in-the-moment. With out-of-the-moment recognition, recognition is a separate post-handwriting process, discontinuous from input. In other words, the capture of ink occurs, the ink is stored and saved, and the user goes on to other tasks. At some later time, a user (either the same user or a different user) returns to the digital ink and applies handwriting recognition. Out-of-the-moment verification is challenging because the information loses much of the context that is available in-the-moment. Users are no longer actively capturing information, instead, the information can be many hours old, and the intent of what was to be written is no longer fresh in the user's mind.

While out-of-the-moment verification may occur with both Anoto pen technology and Tablet PCs, the important characteristic of out-of-the-moment verification is that recognition results are not presented during capture. The raw digital ink can be an aid in verifying the recognizer output during out-of-the-moment verification, but if the ink was written quickly or by someone other than the person verifying, it may be difficult to decipher and correlate with recognizer output. If illegible ink coexists with out-of-the-moment verification, users may miss errors.

An additional problem that is a result of out-of-the-moment recognition is the possibility of there being a batch of documents that need to be recognized and then verified by a user, creating additional potential for recognition errors to be missed. Over the course of hours or days, a significant amount of data can be captured and stored by an Anoto pen or by a user of Tablet applications like Microsoft Journal. Users may then apply recognition to, and verify the correctness of, a large number of documents consecutively, and, as users examine the documents, boredom, habituation, or fatigue may cause them to miss errors made by the recognizer. While these missed or residual errors may not be significant in, for example, a classroom notetaking task, they may be very costly in a health care setting.

2.4 Error Discovery in Handwriting Recognition

Out-of-the-moment recognition can make the verification task more difficult for the user than in-the-moment, increasing the probability of errors in recognition being missed by a user. To reduce this risk of residual error, error discovery must be supported by the recognition interface. This section discusses related work in error discovery, looking at both aiding users in error discovery as well as automated error discovery by the system.

The most common approach to error discovery is to rely on the user to indicate an error in the recognized text, without assistance from the system in the discovery process [12]. Users of these systems examine the input provided to the system and compare it to recognizer output displayed on the screen. This is not sufficient to ensure that all errors in recognition are discovered, though, as users are prone to error themselves, and are likely to miss some of the errors in recognizer output. To decrease the possibility of a user missing an error in recognition, and increase confidence in user accuracy, techniques must be used which aid users in the discovery process.

The only technique found in the literature to aid users in error discovery is the use of recognizer confidence values. This can be seen exemplified in Mankoff and Abowd's PenPad system [12], which changes the colour of a word based on confidence values from the handwriting recognizer. Whether these confidence visualizations aid the discovery of handwriting recognition errors is an open question.

Displaying recognizer confidence values as an aid to error discovery have been studied in speech recognition interfaces [19, 20]. In these speech recognition interfaces, as in Mankoff and Abowd's PenPad system, recognizer output is flagged when the confidence values for the output are low. In speech recognition, these visualizations have been found to have any significant effect on a user's ability to discover errors [20]. Researchers note that the presentation of confidence values in speech recognition may have a negative effect on error discovery, causing users to miss errors that do not have low confidence values [20]. Though users are found to be more likely to discover errors that have been highlighted with low confidence, these visualizations seem to draw users' attention to flagged data, causing users to miss errors not highlighted by the system.

An alternate approach to error discovery is to try to eliminate human verification and to instead automate the process. Baber and Hone [2] experimented with both thresholding on confidence values and with the use of a rule-based system to determine if an error is present in a speech recognition interface. Thresholding suffers the same problem of recognizer inaccuracy as confidence visualization, specifically that the recognizer may have high confidence in an incorrect result. As well, a rule based system for diagnosing errors has proven extremely complex to create. A rule-based system frequently needs knowledge of the context in which something is being written. For example, '!ate' would be a less probable interpretation than 'late' in normal contexts, but in the context of taking notes in a programming course, the probability of '!ate' would increase. Any automated error discovery technique might be unaware of context-based variations in allowable output, forcing human intervention to both inform the recognizer and validate the use of appropriate context. As a result of the difficulty of automatic error discovery, Mankoff claims that only reliable source for error discovery is an explicit notification of an error by direct user input [12]. However, as noted by Bourguet [3], although the users are primarily responsible for notifying the system of errors in recognition, the system is still responsible for enabling the discovery of errors.

Despite the lack of research literature on error discovery, handwriting recognition systems do contain error discovery aids. Microsoft's Journal both displays recognizer output near handwritten text, and supports confidence highlighting for batch recognition tasks. These techniques can be used both in real time, for in-the-moment verification, and for batch validation of a large set of documents, i.e. for out-of-the-moment verification. While in use, there is no research supporting that these techniques aid error discovery.

2.5 Summary

In this chapter, the background and motivation for the work presented in this thesis has been described. While handwriting recognition systems are able to achieve high accuracy rates, there are a number of factors which will prevent handwriting recognition systems from achieving perfect accuracy; handwriting recognition will always contain some degree of error. Because of this unreliability in recognizer output, recognition interfaces need to support the handling of recognition errors for handwriting recognition to be used in any system where a high input accuracy is necessary.

While the handling of errors in handwriting recognition has been acknowledged in research, little work exists that looks at the problem of discovering errors in recognition. The reason for this lack of research is that the problem of error handling has been looked at largely in an in-the-moment recognition setting, where the user is able to view the output of the recognizer as they write any input. In an in-the-moment recognition scenario, the problem of error discovery becomes mostly trivial, as the user can quickly verify the output of the recognizer as they write, and the main problem becomes correcting the recognizer output using pen input.

The problem changes, however, in an out-of-the-moment recognition scenario, when recognition is not performed on the ink until some time after input, causing a delay between recognition and verification. In this scenario, the user is no longer in the moment that the information is being recorded while verifying, and may no longer recall what the intended input was, making errors difficult to recognize. While the presentation of the original ink input can be used to give users something to compare the results against, users may misinterpret the ink themselves, missing errors in recognition. With out-of-the-moment recognition, the problem of batch verification is also possible, where a user must verify the results of recognition performed on a large amount of data, which could cause boredom, habituation or fatigue in the user, causing errors to be missed.

Very little related research exists which examines the issue of error discovery for handwriting recognition systems. The only suggested technique found in literature is a confidence highlighting technique, where recognition results with low confidence values from the recognizer are highlighted to bring a user's attention to those areas. However, while there has been no evaluation of this technique for handwriting recognition, results of its use in speech recognition have implied that it may have negative effects on a user's ability to discover errors, and may not be suitable for achieving high accuracy in error discovery. Automated methods of error discovery have also been proposed for recognition systems, such as thresholding and rule-based systems, where the system decides if an error is present rather than relying on user verification. However, these also suffer from issues that cause them to be unreliable for use in obtaining high accuracy results.

Chapter 3

Design

Due to the lack of previous research on the subject, no proven methods exist for aiding users in error discovery for handwriting recognition. Given this lack of previous designs to inform the design of new techniques, the first goal of this chapter is to provide a framework for the exploration of design alternatives. From this framework, we create a set of designs for aiding users in the discovery of handwriting recognition errors.

First, a design space is described for the design of discovery aids. This design space describes in detail seven different factors which can be used and varied in the creation of error discovery aids, with examples given for each. Next, designs for seven error discovery aids are presented and described in detail. The error discovery aids presented are representative of a majority of the design space, and are designed to solve specific problems in recognition or shortcomings in the discovery process.

3.1 Design Space

To explore design alternatives for error discovery, we explored different features which can be varied to design new discovery aids. The set of features identified form an initial space of potential designs, commonly called a *design space* in Human-Computer Interaction work [10].

This design space is a useful tool for identifying additional factors that can be taken advantage of in the design of a solution, and in understanding how these factors can be varied. In addition, it is also a useful tool in understanding how different designs relate to one another when attempting to compare multiple designs. While our design space does not exhaustively cover all possible factors, it does create a space from which to pull ideas for designs of error discovery techniques. The different dimensions identified in this design space are depicted in Table 3.1, and described in greater detail below.

Dimension	**Description**
Separation of Error Discovery and Correction	Does a user correct an error immediately after discovery, or are discovery and correction separate tasks?
Availability	Is both the ink and recognized text available on screen? Availability can vary between static, to on-demand availability, to changing over time.
Proximity of Ink and Recognized Text	How close is the digital ink to the recognized text? Proximity can vary from ditantly separated to overlapping.
Modality	What modalities are used to present information to the user? Single and multimodal presentations are possible.
Visual Manipulation of Recognizer Output	Is the recognized output visually augmented in any way? Visual manipulations can be used to make errors in recognition stand out to a user.
Visual Manipulation of Handwritten Input	Is the display of the digital ink visually augmented in any way? Visual manipulations can be used to draw a user's attention to areas of ink which may have been misrecognized.
Presenting Hidden Information	Is any information which usually remains hidden presented to the user? Recognizer metadata, historical data, and contextual information can all be presented to the user in ways which may aid error discovery.

Table 3.1: The different dimensions in the design space for error discovery aids.

3.1.1 Separation of Error Discovery and Correction

There are two steps to handling errors in a recognition interface from the user's perspective: discovering errors and correcting errors. These steps can be performed in parallel (when a user discovers an error they immediately correct it) or they can be performed separately (a user flags recognition errors, leaving the correction of flagged errors until later). Performing the two steps in parallel may be somewhat faster overall, but separating the two steps may allow users to focus more directly on the error discovery process, allowing them to discover more errors.

In addition, in some systems it may be enough to signify that an error is present, leaving the correction step for when the information is needed again. For example, in a health-care setting, medical professionals are often pressed for time, and verifying and correcting handwritten information may take up time that they could be spending with patients. In this scenario, only having to flag errors could save a significant amount of time, especially if there is a large amount of information that needs to be recognized. As long as the original ink is stored alongside the recognized results, the next user that needs that data will know where any errors are, and can make any necessary corrections.

3.1.2 Availability

Availability describes the presence or absence of the input (ink) data or the recognizer (text) output. A static technique would display ink data continually on the display alongside the recognized text. While having all of the data constantly visible allows users freedom in their visual search, the display can become crowded. Having all of the data constantly visible also encourages skimming of the data, instead of a careful comparison of the two, which can make errors that are less apparent more likely to be missed (e.g. 'A1' misrecognized as 'Al').

An alternative to static display is to provide information on-demand. Either recognizer output or the original ink is displayed on the screen, and the corresponding information (ink or text, respectively) is hidden. When the user selects some portion of the data on the screen, the corresponding information is displayed. For example, the original ink handwriting could be displayed on the screen and, as the user mouses over the handwriting, the recognized text could be displayed in a tooltip. This on-demand mechanism has the benefit of limiting the amount of information on the display, allowing the user to focus on a portion of the data when verifying recognition. Microsoft Journal includes a type of on-demand recognizer feedback: Journal can display recognizer output just above each word, allowing a user to scan the document to identify recognition errors.

Beyond on-demand techniques, we consider animated techniques, where the state of the information displayed changes over time. For example, rather than having handwriting simply appear when the user calls up a form, the timing information from the ink data could be used to "replay" the creation of the handwritten form. As handwriting appears, the recognizer output could appear alongside the handwriting, attempting to recreate the feel of in-the-moment verification techniques where, as the ink is created, it is immediately recognized.

3.1.3 Proximity of Ink and Recognized Text

Placement of original ink data and recognizer output is another factor that can be manipulated to affect the discovery of recognition errors. At one extreme, the ink data could exist on a sheet of paper beside the computer and recognized data could be displayed on the computer screen. This approach has the benefit of preserving the original context of information capture.

However, data could be merged on the display to varying degrees. For example:

- An electronic version of the original form and the recognized data inserted into a similar form could be displayed side by side. Side-by-side presentation of full data forms can preserve the context of the original information. For example, if an error occurs in a form-filling application because the user wrote outside of the input area for a field, keeping the digital ink in its original context can cause that error to be noticed, where that error can be missed if the ink for each form field was moved to the recognized text.
- The data from the original form could be placed next to the recognized data on an on-screen form. Creating a spatial proximity between the original and recognized information can speed the scanning and comparison of the two.
- Alternatively, one type of data could be placed directly behind the other using a “shadow” style presentation. Shadowing the two data types could cause dissimilarities in character and word shapes to stand out. For example, if the ascenders and descenders of a recognized word don’t match that of the original ink, the dissimilarities between the two word shapes can stand out to a user at a glance.

3.1.4 Modality

Current verification interfaces use visual display of information: ink input and recognizer output are compared visually by a user to determine whether recognizer output is correct. For example, when converting a selection of ink to recognized text in Windows Journal, the recognized text is shown in a textbox, with the digital ink of the word under the cursor shown next to the textbox. It is, however, possible to use other modalities to communicate information to the user, such as sound or speech. Multimodal error correction techniques have been explored in literature [22], but multimodal error discovery techniques have not.

As an example of the use of alternative modalities, handwriting could be displayed on the computer screen and recognizer output could be spoken by the computer using a text-to-speech engine. Using two modalities – visual for the original ink data and auditory for the recognition results – might improve the reliability of the error discovery task by eliminating the need for the user to compare two separate pieces of information, the ink and the output, by visually scanning back and forth between the two items.

3.1.5 Visual Manipulation of Recognizer Output

Visual manipulations can be applied to the output of the handwriting recognizer to attempt to make errors in recognition, or possible errors in recognition, more apparent to the user. At a basic level, visual manipulation can include simple factors such as the use of a programmer's font (fonts designed to reduce ambiguities between characters) for displaying recognized text.

More active manipulations of output include the confidence highlighting technique, where the areas of output with low recognition confidence are visually highlighted. This attempts to bring a user's attention to these areas of low confidence as they are more likely to contain a recognition error. Presenting alternative representations of the recognizer output may also be beneficial. For example, using the health-care motivation in the introduction, one piece of medical information that is usually collected is a patient's weight. Instead of, or as well as, showing the numerical value of that weight, that value could be shown as a point on a graph against that patient's normal weight range. This would allow a medical professional to quickly interpret the output, and notice if a value is out of place, which could be a result of a recognition error.

3.1.6 Visual Manipulation of Handwritten Input

When digital ink is displayed in the recognition interface, visual manipulations to that ink can be used to draw a user's attention to areas likely to contain error. For example, occasionally a handwriting recognizer will misinterpret a stroke as noise and omit it when recognizing the input, when the stroke was intended input. If it were possible to identify the strokes omitted as noise by the recognizer, there may be a benefit in making those strokes highlighted when viewed alongside the recognized results. This could draw a user's attention to the highlighted area making them more likely to discover any recognition errors due to intentional input disregarded as noise.

Visual manipulations on the handwritten input can be varied: from no visual manipulation; to changing a characteristic of the ink, such as the size or colour of strokes; to adding or removing content to the visual presentation of the ink, for example drawing lines over the ink to show how the recognizer is segmenting words allowing users to quickly notice ink that has been improperly segmented.

3.1.7 Presenting Hidden Information

Verification interfaces often have access to a variety of information which is usually left hidden to the user. Handwriting recognizers provide a host of metadata associated with its recognition results. Historical data is often available to an interface related to the data being recognized; for example, in recording patient data in a health-care setting, prior information about that patient usually exists. Also, there is usually a lot of contextual information inherent in data being recorded, such as the language being used, or the form-field the information resides in. The presentation of hidden information to a user could be used to aid them in error discovery.

Recognizer confidence values, one of the pieces of metadata provided by the recognizer, have been used in previous systems to highlight words with low recognizer confidence values, to bring the user's attention to these words. While it may be that highlighting words with low confidence may hamper a user's overall ability to discover recognition errors [20], this has not been studied for handwriting recognition results. Additionally, even if highlighting of results with low confidence reduces a user's error discovery accuracy, there may be some other way to present or use confidence values to benefit error discovery. Besides confidence values, recognizers also contain other metadata, such as n-best lists and ink segmentation data, all of which may also be presented in ways which aid users in the discovery of errors.

Historical values of previous data entered may be accessible to the interface, which could be used to present the data recognized with some context to historical values. For example, in a numerical form-field it could be useful to highlight values which fall outside of the normal range of values entered into that field. If an irregular number is recognized due to an error in recognition, this could be used to make that stand out to a user.

Contextual information of data being entered could be extremely effective at highlighting errors in recognition. Information about the language being entered could be used to bring a user's attention to broken language rules, which could be a result of a recognition error, such as a spelling or grammar errors. In a form-filling scenario, contextual information about the type of information that is expected in the individual fields could allow unexpected values caused by recognition errors to be flagged for the user to check.

3.2 Error Discovery Aids

With the aid of the design space presented in the previous section, a host of designs for error discovery aids were brainstormed. This section presents the designs of seven error discovery aids: spellchecking, visual proximity, highlighting fused characters, highlighting disregarded strokes, animation, text-to-speech and typographic manipulation.

As was discussed in the introduction, the problem of error discovery for handwriting recognition was narrowed to an out-of-the-moment form-filling scenario for this thesis work. Therefore, the discovery aids considered are designed in the context of a verification task for recognition results in a form-filling application, where the handwritten information in the form has already been collected. Additionally, each of the discovery aids has been designed, and implemented, as extensions in functionality to a regular textbox. When a description of a technique talks about displaying ink and recognized text at the same time, the space underneath the textbox has been expanded and has an added area to display the ink. The handwriting recognizer from Microsoft's Tablet PC SDK is used for recognizing digital ink, as well as obtaining any additional information needed from the recognizer, such as ink segmentation information.

The seven error discovery aid designs are discussed in more detail below. For each design, there is a description of what the aid does and how it helps discovery, a description of any user interaction required with use of the aid, and a description of how the aid can be implemented in a verification interface.

3.2.1 Spellcheck

Modern handwriting recognition systems generally use some kind of dictionary to aid in the recognition process. Even still, there are occasions when the recognizer is unable to match the handwriting to a word in the dictionary, in which case character recognition is performed. If character recognition fails on a word, a spelling error can occur. The spellcheck discovery aid brings these kinds of errors to a user's attention through the automatic underlining of any spelling errors present in recognized text, as is common in most modern document editors.

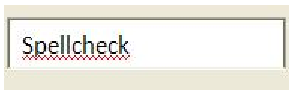


Figure 3.1: Highlighting spelling errors in recognition results.

Spellchecking is implemented using Microsoft Word's spellchecker to identify words that contained spelling errors, and then underlining those errors with a red wavy line, which is usually associated with spelling errors in modern software. An example of this can be seen in Figure 3.1. The use of Microsoft's spellchecking software ensures that a large, robust dictionary is available, making false positives unlikely.

Once an underlined spelling error is identified by a user as an error in recognition, the user can then manually correct that error. However, most document editors also provide a context menu for spelling errors which contains a list of suggested corrections. This discovery aid can provide this context menu as a means of supporting error correction.

3.2.2 Visual Proximity

Placing the digital form of the handwritten ink in close proximity to the recognized text allows users to quickly compare the two, making it easier to notice differences caused by recognition errors. However, this is not always possible. Adding fields to display the ink alongside the recognized text can quickly overload the display, which becomes more of a problem the greater the number of fields there are in the form. While only either the ink or text need be always visible, the user must be able to access the corresponding data (text or ink respectively) for verification.

The visual proximity technique hides the digital ink, only showing the form fields containing the recognized text. When a user gives focus to a form field, using either the mouse or tab, the digital ink is displayed above that field. Users can then compare the ink to recognized output, as shown in Figure 3.2, and correct any errors found.

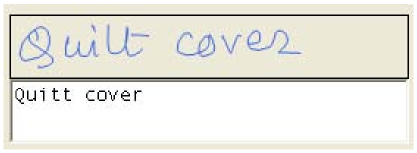


Figure 3.2: The visual proximity aid.

3.2.3 Highlighting Fused characters

A characteristic of human handwriting that causes recognizers difficulty, which can result in recognition errors, is when characters are fused together, meaning that two or more characters are overlapping or connected in some manner. This technique detects fused characters in the digital ink input, which is displayed statically next to the recognized output, and changes the colour of that ink to bring a user's attention to these areas, making them more likely to discover any errors due to fused characters, seen in Figure 3.3.

The implementation of this discovery aid identifies fused characters by analyzing how the recognizer is interpreting each individual stroke of the handwriting. If a single stroke is recognized as more than one character, it is assumed that multiple characters were written in a single stroke. If the recognizer returns more than one character, it is assumed that the stroke contains two connected characters. A shortcoming of this technique is that it will only identify characters that are fused due to being written in a connected-cursive style, and not disconnected characters overlapping one another. However, connected cursive handwriting can still be an issue for some recognizers, making this technique useful.

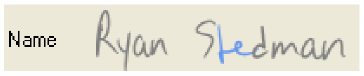


Figure 3.3: The highlighting fused characters aid.

This discovery aid could be incorporated into a verification interface in two ways. The ink displayed on the interface can be highlighted statically from the moment the interface is displayed. This could encourage users to skim over results at the start of the verification task to quickly identify errors due to fused characters. Alternatively, the highlighting could be enabled through some kind of user interaction, for example enable highlighting only when the user has given focus to the textbox containing the respective text. As opposed to a static presentation, this could discourage skimming and instead encourage users to go through recognized results one field at a time, yet still bring their attention to fused characters. In either case, once a user discovers an error with this discovery aid, they can manually correct that error.

3.2.4 Highlighting Disregarded Strokes

Handwriting recognizers must occasionally disregard strokes as random noise, as humans writers will sometimes touch the writing instrument to the writing surface without the intention of creating a stroke. Without the ability to disregard some strokes, handwriting recognition would be far more error prone, due to this artifact of human handwriting. However, due to the ambiguous nature of handwritten input, recognizers are not always accurate in determining which strokes are intended input and which are not; occasionally, intentional strokes get disregarded as noise causing the recognition to fail. As in the technique that highlights fused characters, this technique displays the handwritten ink next to the recognized text, changing the colour of the strokes that have been disregarded as noise to draw a user's attention to these areas. An example of this discovery aid in use can be seen in Figure 3.4.

Disregarded strokes are detected in the implementation by using the recognizer's segmentation information to break the ink into a series of words, and each word into a series of strokes. Each stroke is then individually recognized and compared to the recognized value for the word that the stroke belongs to. In case a stroke may be part of a multi-stroke character, each stroke is also recognized with the previous stroke in the word, as well as the next stroke in the word, and compared with the recognized result for that entire word. In both the single and multi-stroke comparisons, if the recognition of any stroke(s) is not found in the recognized result for the entire word, that stroke is assumed to have been disregarded as noise by the recognizer, and its colour is changed. This discovery aid can be used in an interface in the same ways as the highlighting fused characters aid.

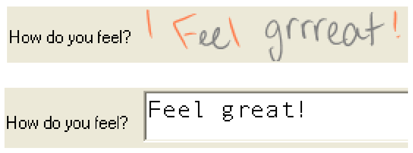
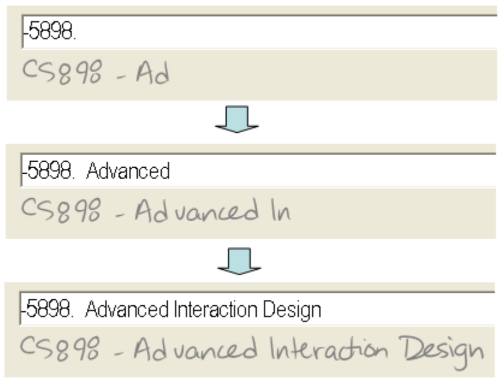


Figure 3.4: The highlighting disregarded strokes aid.

3.2.5 Animation

One of the main benefits in-the-moment handwriting recognition has in error discovery is that a user is able to view the output of the handwriting recognizer as they write. This allows them to quickly compare the recognizer output against what they are writing, while their intended input is still fresh in their mind. This can make the discovery of errors a trivial task, as long as some attention is paid by the user to the recognizer output.

The animation discovery aid attempts to recreate the benefits of in-the-moment recognition in an out-of-the-moment setting by drawing the digital ink exactly as it was written while displaying and updating the recognizer output as each additional stroke is added. This allows users the benefit of being able to compare the output of the recognizer to the ink input in much the same manner that is possible in an in-the-moment error discovery scenario, which can be seen in Figure 3.5.



-5898.
CS898 - Ad
-5898. Advanced
CS898 - Advanced In
-5898. Advanced Interaction Design
CS898 - Advanced Interaction Design

Figure 3.5: The animation discovery aid.

An additional benefit of animation over in-the-moment recognition, is that users are no longer in the state of inputting information, allowing them to focus their attention on discovering errors. For example, in an in-the-moment recognition scenario, a user may be writing information relating to a task that they are currently performing, dividing their attention between the task and writing. In this scenario, a user may be more focused on the task they are performing, not being careful in verifying recognition correctness, and missing errors as a result. With the animation discovery aid, the benefits of in-the-moment recognition are present, but a user's attention will be solely on verification.

It is necessary for the animation discovery aid to be user-activated, to ensure that the user's attention is focused, since it is not useful if the user is not looking. However, there are a couple of ways that animation could be activated in a verification interface. One method would be to allow users to activate animation on a single field at a time, in the order of their choosing, by a user giving focus to a field, using the mouse or tab. Once animation is activated on a field, it needs to play out before the user is able to correct any errors discovered, since the content of the recognized text field changes throughout the animation.

Animation can also be set to play out on all the form fields consecutively with the press of a play button. As the animation plays out on the entire form, the interface can allow users to flag fields which they have discovered errors in, highlighting those fields for correction after animation has completed. Alternatively, the animation could be paused by a user to allow correction of errors as they are discovered, either through use of a pause button on the interface, or by allowing the user to click on a field to indicate that an error is present and they want to switch from discovery to correction modes. Once the user has finished correcting, the animation can be resumed by pressing the play button again.

There is a trade-off in the speed in which users can verify recognized output with the use of this technique. Having the ink drawn out is much slower than making everything visible initially and allowing the user to scan. However, animation speeds can be increased to much greater than the original input speed, increasing verification speed. Also, a speed for accuracy trade-off is acceptable in situations where high accuracy is necessary.

3.2.6 Text-to-Speech

The text-to-speech technique speaks the recognized text using a text-to-speech engine while showing the handwritten ink. Each field is spoken serially, highlighting the ink that is being spoken. There are two potential benefits to using this cross-modal verification technique. First, the user does not need to repeatedly scan back and forth between the input and output to identify any errors. The user can focus solely on the input and listen for the output. Second, as a user reads the handwritten ink they create an expectation of what they will hear; if the speech output does not conform to that expectation, it can stand out as being incorrect, which could signify an error.

The text-to-speech technique, exemplified in Figure 3.6, is activated by the user clicking a play button, which causes the form fields to change from showing textboxes with the recognized text to showing the handwritten ink. Starting with the first field in the form, text-to-speech is performed on the recognized results of the field, and the ink for that form field changes colour so a user knows which field is being spoken. To make the speech easier to understand, numbers, combinations of numbers and letters, and non-alphanumeric symbols are explicitly read. For example, for a hyphen, the text-to-speech engine says the word “hyphen” rather than leaving a pause. Similarly, a long number such as 109183 would be read by the text-to-speech engine as “one zero nine one eight three”.

There are a two ways that text-to-speech can be implemented in a verification interface to allow users to correct errors that they discover. First, the interface can allow a user to pause the playback of text-to-speech, switching back the form fields from displaying ink to showing the textboxes with the recognized text. Pausing could be enabled with the use of a single pause button for the entire interface, or by allowing users to click an ink field to indicate that an error is present that

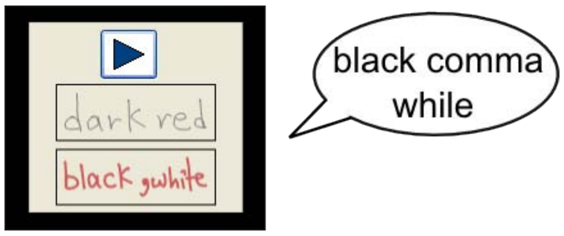


Figure 3.6: An exemplification of the text-to-speech technique.

needs to be corrected. Second, users could be allowed to flag any suspected errors that they hear by clicking on the ink that they believe contains an error. Once the text-to-speech engine has spoken all of the fields, the form fields redisplay the recognized text and the background colour of a field that has been flagged by a user is highlighted. Users can then correct any errors that exist in the recognition results.

Similar to the animation discovery aid, the drawback of text-to-speech is, potentially, speed. Despite the fact that people can understand speech at speeds of up to 300 words per minute (or 5 words per second), text-to-speech may be slower than other techniques. However, if the primary goal is to increase the number of errors identified, slower sequential techniques such as text-to-speech merit consideration.

3.2.7 Typographic Manipulation

Handwriting recognizers sometimes recognize characters as numbers, or numbers as characters. They may also incorrectly insert punctuation or symbols due to spurious ink strokes or miss decimal points due to noise filtering. These kinds of errors can be difficult for users to notice, as the misrecognized output can closely resemble the correct output. For example, if the input of the letter ”I” (upper-case i) is misrecognized as a 1 (one) or l (lower-case L), a user could easily miss this error if they are not examining the output carefully enough.

The goal of typographic manipulation is to allow these kinds of errors to stand out by changing the typographic characteristics of letters, numbers, or punctuation and symbols, making them distinct from one another. Users select one of the three character types to alter the typographic characteristics of in the recognized text, for example with use of a slider, as seen in Figure 3.7. When a character type is selected to be manipulated, the font size of all characters, across all form-fields, of that type is enlarged and the font colour is changed. This allows users to quickly scan the recognized results for any out of place characters, for example a number where no numbers are expected, as well as creating a clear disambiguation between any characters similar in shape, which could be confused.

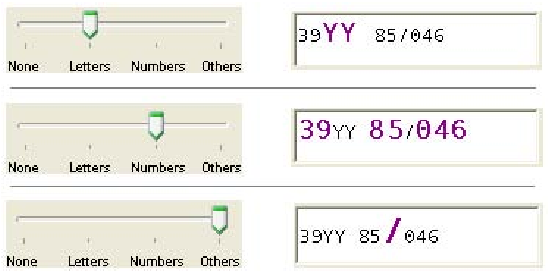


Figure 3.7: The typographic manipulation technique

3.3 Summary

This chapter explores the range of possible designs of error discovery aids. A design space was presented laying out the design dimensions which can be used to aid in the exploration of designs for error discovery aids. With the aid of this design space, several designs for error discovery aids were created for use in an out-of-the-moment form-filling verification interface. These designs represent the first set of techniques designed specifically to aid users in the discovery of handwriting recognition errors, outside of confidence highlighting.

Chapter 4

Evaluation

The designs presented in the previous chapter are all potentially useful as error discovery aids. However, evaluation is needed to determine important factors in their design and use, such as how users perceive these designs, and if the designs could be improved in a way which would improve user perception and use. Additionally, the effects of these techniques on a user's performance in error discovery need to be evaluated to determine if they are beneficial to the task of error discovery.

This chapter presents evaluations performed on the error discovery aids from the previous chapter. First, an exploratory evaluation is described which was used to guide and refine the designs. Next, a formal evaluation on three of the most promising aids against a control is presented. This formal study evaluated the effectiveness of the three techniques in aiding users to discover errors in a carefully designed out-of-the-moment handwriting recognition verification task.

4.1 Exploratory Evaluation

The final designs of the error discovery techniques described in this thesis were based on the results obtained from a iterative evaluation and refinement of these aids. While the ultimate goal of the error discovery aids is to enable users to easily and quickly discover handwriting recognition errors, the goal of our initial evaluation was to determine how users perceive these discovery aids, identify any difficulties that may arise in their use, and refine the aids to maximize their effectiveness.

An additional goal of this evaluation was to determine which of the techniques proposed are the most likely to be successful in aiding users in the discovery of handwriting recognition errors. Though each of the techniques have potential to succeed, the large number of techniques proposed creates a difficulty in attempting to evaluate the effectiveness of all of them in aiding error discovery against a control, as well as against each other. Narrowing the scope of the evaluation is necessary before embarking on any formal experimental evaluation of a set of discovery aids.

This section provides a detailed account of the formative evaluation performed on the error discovery aids, and the results obtained from the evaluation. The results are then analyzed, and their effect on the final designs are discussed.

As a result of this evaluation, changes were made to the designs of the visual proximity, animation and typographic manipulation techniques. Additionally, results of this evaluation imply that the visual proximity, typographic manipulation and text-to-speech techniques had the greatest potential for success, and should be explored further.

4.1.1 Differences in Error Discovery Aid Designs

Several of the error discovery techniques used in this evaluation were earlier design variants of the techniques presented in the previous chapter, and were redesigned as a result of this evaluation. These differences are described below.

- *Visual Proximity*: The original design of the visual proximity technique displayed the recognized ink in a tooltip when a user hovers the mouse over the recognized form field, rather than displaying it statically when the field has focus.
- *Animation*: In the animation technique, the displayed results of the recognized text were updated after each segmented word was drawn out, rather than after each additional stroke is added.
- *Typographic Manipulation*: The original design of this technique had the typographic properties of the three different character types statically set, rather than allowing them to be manipulated by the user. Letters were the default 12pt Times New Roman font, numbers were set to a 14pt Arial font, and the font for punctuation and symbols was set to 16pt Arial Black.

The "highlighting fused characters" aid changed ink colour of fused characters to light blue, and the highlighting disregarded strokes aid changed ink colour to orange. The spellchecking and text-to-speech discovery aids were the same as the currently described designs.

4.1.2 Method

Several weeks prior to the evaluation, each participant was asked to fill out a short survey using stylus input on a tablet PC, which allowed the use of samples of each participant's handwriting to recognize and use in the exploratory evaluation. By collecting the handwritten data at an earlier time, we replicate an out-of-the-moment recognition scenario. The questions asked in the survey, shown in Figure 4.1, were designed to obtain a variety of handwriting data from the participants (i.e. words, numbers and alphanumeric sequences), so that the discovery aids could be tested on a variety of data types.

The think-aloud method was used during the evaluation. Participants were asked to perform a set of tasks, and vocalize their thoughts during the task. This technique gives the benefit of allowing insight into how a user perceives any system that they are using, rather than just observing them perform the task. The results of the think-aloud are used to obtain a greater understanding of any problems encountered during the task. Comments made by participants during a think-aloud can also help identify solutions to problems encountered.

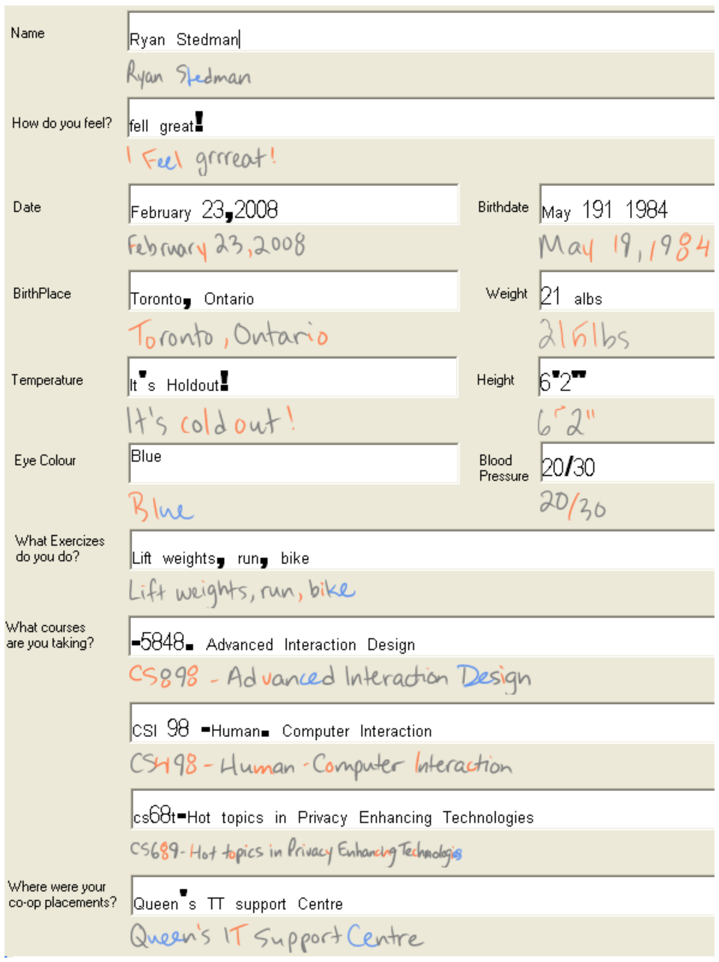
In the first task that participants performed, they were shown the interface with the participant's handwritten response to each question, and the recognized text above it (Figure 4.1). The recognized text displayed was manipulated using a typographic manipulation technique, and spelling errors were underlined. The fused character and disregarded stroke highlighting discovery aids were also used in the interface. Participants were asked to find any recognition errors and repair them as quickly as possible, thinking aloud as they did so.

In the second task, participants were asked to use the animation technique in a second experimental interface. In this interface, only a single form field was displayed at a time. The field contained random selections of ink input from the previous interface. Participants activated the animation technique on the field by pressing a button. As animation was only applied to a single field, users did not have the ability to stop playback. The goal of this task was to discover how useful participants found animation, i.e. if they found it any easier to identify any errors when animation was used. This task was repeated several times, with several different selections of the participant's handwriting.

For the final task the participants used the text-to-speech technique. Again, only a single selection of handwriting was visible at a time. However, by this point the participant had already viewed the recognized results of their handwriting twice, so handwriting from different users was shown in case a participant remembered errors in output from a previous task. The recognized text was not visible to the user during the task. Instead, for each selection of input used, only the digital ink was shown, and a text-to-speech engine spoke the recognizer output. Participants were asked whether or not they believed an error existed in the recognized output based on what they heard, as compared to the ink they viewed. The goal of this task was to determine how well a user could identify an error based only on text-to-speech output, and to determine if anything could be improved in the way the technique spoke the recognized text.

4.1.3 Participants

The evaluation consisted of three participants: two male graduate students in computer science (P1 and P2) and one female (P3), who worked as a registered nurse in a long-term care home.



Name	Ryan Stedman		
	Ryan Stedman		
How do you feel?	fell great!		
	I feel grrreat!		
Date	February 23,2008	Birthdate	May 191 1984
	February 23,2008		May 19,1984
BirthPlace	Toronto, Ontario	Weight	21 albs
	Toronto, Ontario		216lbs
Temperature	It's Holdout!	Height	6'2"
	It's cold out!		6'2"
Eye Colour	Blue	Blood Pressure	20/30
	Blue		20/30
What Exercizes do you do?	Lift weights, run, bike		
	Lift weights, run, bike		
What courses are you taking?	-5848. Advanced Interaction Design		
	CS898 - Advanced Interaction Design		
	CSI 98 -Human- Computer Interaction		
	CS498 - Human-Computer Interaction		
	cs68t-Hot topics in Privacy Enhancing Technologies		
	CS689- Hot topics in Privacy Enhancing Technologies		
Where were your co-op placements?	Queen's TT support Centre		
	Queen's IT Support Centre		

Figure 4.1: Interface used for the first task in the exploratory evaluation.

4.1.4 Results

During the first task, both P1 and P2 were confused as to what the difference was between the two highlighting colours in the ink, which slowed them down until their meaning was explained. P3, on the other hand, realized within a couple of minutes that the blue highlighting meant that characters were fused. Once explained, P1 noted he found the highlighting to be useful in discovering a couple of the errors present due to disregarded strokes. Neither P2 nor P3 were able to use the highlighting to their benefit.

P2 found the different font types very useful for quickly spotting certain kinds of errors, specifically when numbers were erroneously mixed into words, or when special characters were present that were not supposed to be (in one field it recognized a '£' where there should have been a '6'). P1 and P3 on the other hand did not find the font manipulations useful in discovering errors. In one instance it actually hindered P1, when he was uncertain if a character was a front-slash (/) or an italicized 'l'. To figure out which character it was he had to delete it and re-type the character. Afterward, P1 noted that if there were a more clear disambiguation between the different character types, he would not have had any problems.

Spellchecking was found to be useless as an error discovery aid in this task. Due to the dictionary-based fitting of recognition results in the Microsoft recognizer, discussed more in the following section, no recognizer output contained any spelling errors, even in the case of a recognition error.

For the second task, none of the participants found the animation of the ink to be of any use in discovering errors. Both P1 and P2 found that the handwriting was being drawn too quickly, hindering visual comparison of the ink to the recognized text. P2 also noted that comparing the handwritten ink to the recognized text was made even harder since individual words in the ink, and their corresponding recognized text, were not always visually aligned; a result of the size differences between handwriting and typeset text. As a result, both P1 and P2 found it easier to just look at the recognized text after it was written out.

All of the participants found that the text to speech technique to be effective in discovering errors. Without being able to see the recognized text, all participants immediately noticed if a word was misrecognized the first time they heard it spoken, though they were unable to identify exactly what the error was.

4.1.5 Discussion

Based on the results of this evaluation, a number of implications about the designs of these error discovery aids were drawn, leading to changes in several of the designs.

Ink Highlighting Techniques

Though neither P2 nor P3 were able to make any use of either of the ink highlighting techniques, the fact that P1 was able to use it to discover errors is encouraging. Part of the reason that P2 and P3 did not find it useful was that they did not have as much of a mental connection between a recognition error and a highlighting mark, where P1 had more of an understanding of the significance of a highlighted stroke. The original intention of these two techniques was to draw a user's attention to the highlighted areas without implying that an error is present. These results imply that for these techniques to be useful, the user needs to be aware that highlighting signifies potential errors.

The confusion seen by both P1 and P2 over the different colours could be solved by explaining the meaning of the two colours before using the interface. However, it is not completely necessary that a user understands the mapping of colour to error type. As long as they have a mental connection between the highlighting and the possibility of an error these discovery aids may prove effective. Their confusion over colour interpretation could instead be resolved by highlighting all errors with the same colour. A highlighted area would simply map to a potential error, rather than a specific type of potential error.

The main cause of failure of the ink highlighting techniques is due to the inaccuracy of the methods used in the implementations of these discovery aids for identifying areas in the ink, fused characters and disregarded strokes, that they are attempting to bring to a user's attention. The "highlighting disregarded strokes" aid suffers from a large number of false positives. While some disregarded strokes are correctly highlighted, the number of false positives makes the use of this technique equivalent to simply comparing ink to recognized text without any highlighting attempting to guide a user's attention. The "highlighting fused characters" aid, on the other hand, suffers from only being able to identify characters fused due to being drawn in a single connected stroke, leaving a large number of fused characters unidentified. In addition, results of the use of this discovery aid in the exploratory evaluation seem to show that characters fused in a connected stroke do not pose a problem for recognition by the Microsoft recognizer, as none of the highlighted areas represented a recognition error.

While inaccurate, the methods used by these two discovery aids to identify fused characters and disregarded strokes are useful in illustrating the potential of these techniques. Highlighting can be used as a starting point to create better aids that guide users attention to problem areas of ink. However, to be useful, improved algorithms to identify likely problem areas in ink are also needed.

Spellcheck

The absence of spelling errors in any of the recognition results made the use of the spellcheck technique useless in this evaluation, and thus no information was gained on how users were able to use it. However, the absence of spelling errors in recognition results does show that using spellchecking as an error discovery aid may not be useful at all for the Microsoft handwriting recognizer. Handwriting recognizers use dictionary-based fitting of recognition results so stringently that little to no spelling errors will ever be present in their recognition results.

However, spellchecking does still merit consideration as a discovery aid. Although spelling errors are very unlikely to occur, and did not occur during the exploratory evaluation, we have observed occurrences of the Microsoft recognizer producing spelling errors as a result of errors in recognition in our own informal observations of this discovery aid. While very infrequent, spellchecking can aid a user in quickly identifying these fringe cases, and correcting any errors that are present.

Typographic Manipulation

The typographic manipulation technique had some success, particularly for P1. However, as noted by P1, further disambiguation is required between the different character types. Even with the font differences used in the interface for this exploratory evaluation, there is still the possibility of confusion. We refined the implementation of the typographic manipulation aid to instead make the typography of the three character types change through user control, causing a selected character type to be typographically distinct from the other two. This ensures that no confusion will exist between character types, as only one character type will stand out at a time.

Animation

The animation technique failed in its intended purpose, with none of the participants being able to use it in the intended manner. One of the main reasons that it was observed to have failed was that participants were unable to quickly compare the ink words to their recognized text as the ink was being drawn. Participants instead waited for the animation to complete and then compared the input and output when they were both fully visible. While the speed at which ink was drawn may have been a factor, animation speed was not much faster than a user would have input the information. Further slowing of the animation speed would also make the technique less efficient, as users would have to wait for a slow animation to complete.

Another possible reason that the participants found performing any comparison during animation difficult was that each recognized word would only appear once the entire ink for that word had been written out. As soon as recognized text appeared, the user would be distracted by the following ink word being written out. While no users commented specifically on this distraction, it had been observed in our own personal use of this discovery aid. The current implementation of the animation technique addresses this problem by updating the recognition results after each additional stroke is drawn. This creates a smoother animation, rather than the abrupt change of an entire word appearing in the textbox, causing less visual distraction.

Text-to-Speech

The text-to-speech technique, in this limited evaluation, proved to be an effective method to aid users in the discovery of handwriting recognition errors. Participants had no trouble in understanding its use, or in identifying that errors were present in recognition results. Although the participants were unable to identify specifically what the error was, this would not be necessary in a verification scenario, as they could re-check the field that they identified an error in after the text-to-speech completed, and correct any errors present. This technique gives users a way to identify where errors are present, separating error discovery and error correction.

4.2 Formal Evaluation

While many of the error discovery aids described in the previous section may seem obvious candidates to experiment with, no comparisons exist between the various aids. Whether any of the designs, or even which dimensions described in the design space, help or hinder the error discovery process is unknown. Furthermore, none of these discovery aids have been compared to a basic error discovery scenario, such as comparing a handwritten page of notes to on-screen recognition results.

This section presents an evaluation of three of the the error discovery aids, and compares these techniques to a control condition. While there are more than three discovery aids designed, attempting to properly evaluate all of them against each other, as well as a control, would make the size of the study unmanageable. To keep the size of the study manageable, while still learning as much as possible, three techniques were chosen and implemented into separate verification interfaces based on their observed effectiveness during the think-aloud evaluations, and on the coverage they collectively provide of the design space. The error discovery aids evaluated were the visual proximity aid, the typographic manipulation aid, and the text-to-speech aid. A control condition was used as a baseline to compare the three discovery aids against.

The goal of this evaluation is to study a set of discovery aids which represent a large portion of the design space, to study the aids and their ability to aid users in the discovery of handwriting recognition errors, and to generalize from this initial set of designs. To achieve this an experiment was designed to contrast the three experimental techniques listed above with a control condition. In the control condition, recognizer output was displayed on the computer screen, and the participant would compare the output to data on the original handwritten paper form created using an Anoto pen.

As a result of this experimental evaluation, it was found that while the typographic manipulation and text-to-speech aids had no significant effects on a user's ability to discover errors, the visual proximity aid did provide a statistically significant improvement in the number of errors in recognition caught by a user. Also, results of the typographic manipulation aid imply that attempting to guide a user's attention may be detrimental to their performance in error discovery.

4.2.1 Participants

16 volunteer participants were recruited from the university campus with the use of fliers posted around the campus advertising the study. The participants consisted of 11 females and 5 males, with ages ranging from 19-55 (mean = 29.1 , sd = 9.2). Participants received \$50 (CDN) remuneration for participating in the experiment. Each participant was informed that they were free to end their participation in the study, and would still receive a prorated remuneration to the amount of \$10 per session participated in.

4.2.2 Experimental Task

The experiment involved a five-session form-filling transcription and verification task using Anoto pen and paper technology. Participants all completed five one-hour sessions in a five day period. The tasks performed by the participants on each day are as follows:

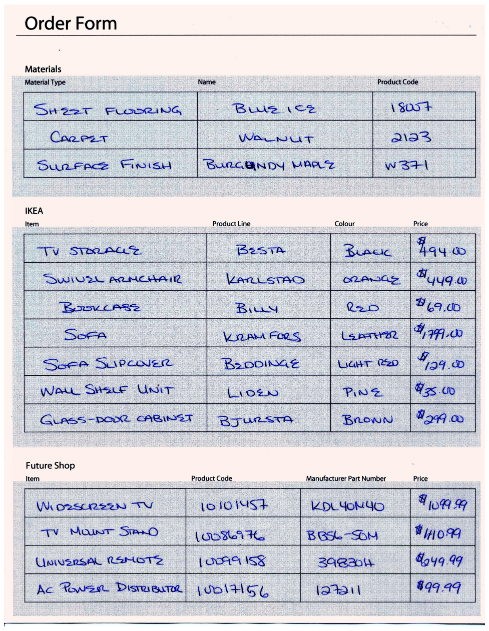
- On the first day of the experiment, participants transcribed data from a set of five forms onto blank forms printed with the Anoto pattern.
- On the second, third and fourth day of the experiment, the participants returned, verified the recognized data from the first day's forms using a verification interface which made use of one of the error discovery aids, or the control, and transcribed five new forms.
- Finally, on the last day of the experiment, participants verified the forms they recorded on day 4, were interviewed on their impressions of the error discovery aids, and were paid for their participation.

The 24 hour delay between input and verification ensured that participants would remember little about the specific content that they had transcribed, so that the verification task mimicked an Anoto-pen-based out-of-the-moment verification task.

Each verification session consisted of a separate condition, where one of four error discovery aids were present in the verification interface:

- Control. Results were shown without any error discovery aids present. Participants compared results to the ink on the original form.
- Visual Proximity. Handwritten ink is shown above the textbox containing the recognized text of that ink, when that textbox has focus.
- Typographic Manipulation. The typography of different character types in recognizer output can be altered to stand out through use of a slider bar at the top of the form verification interface.
- Text-to-Speech. By pressing a play button at the top of the form verification interface, text-to-speech is used to speak the recognition results while showing the user the ink.

To simplify the verification task for each of these conditions, we made some effort to ensure that text on the computer display was not ambiguous, so that errors in the recognized text were not missed because of any ambiguities in that text. We used 10 pt Andale Mono font on a 21" computer display (a programmer's font). This font was chosen because it is designed to reduce the ambiguities between specific



Order Form

Materials

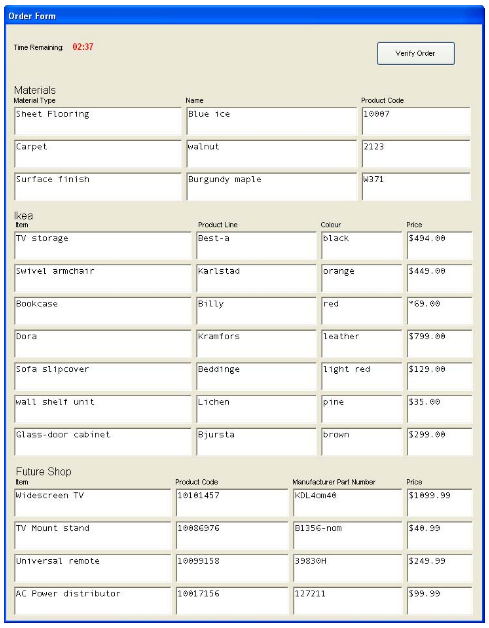
Material Type	Name	Product Code
SHEET FLOORING	BLUE ICE	10007
CARPET	WALNUT	2123
SURFACE FINISH	BURGUNDY MAPLE	W371

IKEA

Item	Product Line	Colour	Price
TV STORAGE	BESTA	BLACK	$494.00
SWIVEL ARMCHAIR	KARLSTAD	ORANGE	$449.00
BOOKCASE	BILLY	RED	$69.00
SOFA	KRAMFORS	LEATHER	$1,799.00
SOFA SLIPCOVER	BEDDINGE	LIGHT RED	$129.00
WALL SHELF UNIT	LIDEN	PINE	$35.00
GLASS-DOOR CABINET	BJURSTA	BROWN	$299.00

Future Shop

Item	Product Code	Manufacturer Part Number	Price
WIDESCREEN TV	10101457	KDL40M40	$1099.99
TV MOUNT STAND	10086976	BB56-SOM	$110.99
UNIVERSAL REMOTE	10099158	39830H	$249.99
AC POWER DISTRIBUTOR	10017156	127211	$99.99



Order Form

Time Remaining: 02:37

Materials

Material Type	Name	Product Code
Sheet Flooring	Blue ice	10007
Carpet	walnut	2123
Surface finish	Burgundy maple	W371

Ikea

Item	Product Line	Colour	Price
TV storage	Best-a	black	$494.00
Swivel armchair	Karlstad	orange	$449.00
Bookcase	Billy	red	*69.00
Dora	Kramfors	leather	$799.00
Sofa slipcover	Beddinge	light red	$129.00
wall shelf unit	Lichen	pine	$35.00
Glass-door cabinet	Bjursta	brown	$299.00

Future Shop

Item	Product Code	Manufacturer Part Number	Price
Widescreen TV	10101457	KDL4om40	$1099.99
TV Mount stand	10086976	B1356-nom	$40.99
Universal remote	10099158	39830H	$249.99
AC Power distributor	10017156	127211	$99.99

Figure 4.2: The experimental tasks. Left shows an example of the paper form filled out by participants. Right shows the verification interface used for the control condition.

characters that may be confused for one another, such as 1 (one) and l (lower-case L). As well, participants were shown the results they were to verify in a form laid out exactly the same as the physical form that they transcribed.

Sessions for days 2 through 4 of the experiment proceeded as follows. Participants were reminded that the goal of the experiment was to test techniques for finding errors in handwriting recognizer output. They were instructed to find and correct any errors present in recognizer results as quickly and accurately as possible. Next, participants were given a verbal overview of the discovery aid to be used to find errors that day, and the experimenter demonstrated the aid's use. Participants performed two blocks: a practice block on the current error discovery aid and an experimental block using the same aid. During the practice block, participants verified the recognition results of two forms filled out by the experimenter. During the experimental block, they verified the data on the five forms they transcribed on the previous day. The screen layout for the verification task is shown in Figure 4.2. The computer recorded the time taken for correction, the errors caught, and any false corrections participants made.

To further constrain the verification task, a three minute time limit was imposed on the verification of each form (shown as a count-down clock in the top left of the display). This was used to prevent participants from taking too long in the verification task, mimicking a real-life scenario where the person verifying is attempting to complete the verification task in a timely manner. Given unlimited time, a participant may spend more time verifying than a user would in a real-world time constrained task, catching more errors as a result. During all practice and experimental blocks, participants were given the original data forms that they transcribed. Though it was not necessary for users to use the original forms for the visual proximity condition, they were still provided with the original forms in case there was a recording error with the Anoto pen.

Upon completion of the experimental block on days two through four, participants transcribed five new forms for use on the following day. Participants transcribed 20 forms over the course of the experiment (five per error discovery technique). The forms were modeled to look like an order form. Each form had 53 fields with a variety of different information types, including item names, product codes and prices. The data for the forms was taken from various online product catalogs. Figure [4.2] depicts the form used in the transcription task.

At the end of each transcription task, the ink from each form was downloaded from the Anoto digital pen and saved. Although the ink was saved digitally, the recognition results displayed to the user in the verification tasks were not the result of handwriting recognition performed on participants' handwriting. Instead, the displayed results were predetermined. Recognition data for each form set was seeded with the same set of errors, regardless of the participant's writing. This allowed the experiment to be controlled for the number and type of errors for each participant for each form for each condition. Each block of five forms for each condition contained 25 predetermined errors. The set of errors was chosen at random from actual handwriting recognition errors that occurred during a exploratory evaluation.

4.2.3 Experimental Setup & Apparatus

Due to the restrictive licensing and software costs of creating custom paper applications using Anoto software and paper-space (the pattern information that is recognized by Anoto pens), the paper forms were created using Stanford's Paper-Toolkit [24]. Using papertoolkit, as well as the paper-space patterns provided with the toolkit, it was possible to create forms layered with dot-patterns recognizable by Anoto pens. A listener service was then created that would recognize when an Anoto pen containing information from the experimental forms was docked, and would start the download process.

During the download process, each stroke captured by the pen was categorized depending on which form field it belonged to, and the individual form fields were converted to Microsoft's ink serializable format (ISF) and saved. To determine which field each stroke belonged to, the first packet (point) of that stroke was compared to the predetermined bounds of the fields. By only looking at the start packet, this allowed participants to draw strokes outside of form field bounds, as would commonly happen when writing a letter with a descender. This allowed more natural, and less constrained writing, to help avoid errors in verification due to participants misreading their own handwriting due to it being overly constrained.

Ink was converted from Anoto format to ISF through a one to one mapping of stroke coordinates from Anoto coordinate space to ISF coordinate space. When displayed to the user in the visual proximity and speech conditions, the ink was scaled to fit the ink display areas, as the coordinates given by the Anoto pen are not on the same scale as ISF.

The canned recognition results used in the experiment were kept in XML. The stored data also contained information identifying recognition errors, thus allowing for automatic logging of errors caught during the experiment. Before participants performed the verification task, each form filled out by the participant was manually checked for transcription errors. If transcription errors were present, we altered the canned recognition results to incorporate these errors. This prevented an error in transcription from causing an unintentional recognition error in our canned recognizer output.

In the transcription tasks, participants wrote using a Logitech io2 digital pen on a patterned form. For the verification tasks, participants used a desktop computer, with a 21" 1280x1024 screen, and, in the text-to-speech condition, participants wore jWin JH-V100 headphones. Text-to-speech synthesis was performed using the Microsoft Sam text-to-speech engine, set at a speech rate of 5.

4.2.4 Procedure

The experiment was designed as a 4-condition (control, visual, typographic, speech) X 4-form sets (form set A, B, C, or D) mixed design, with order of condition and form set counterbalanced using a 4X4 perfect Latin square. Condition was a within subjects factor, and form set (which set of forms corresponded to which condition) and order of condition were between subjects factors. We counterbalanced form set to ensure that one set of forms did not have an "easier" set of errors than any other set of forms. We counterbalanced order of condition to ensure that inexperience during earlier sessions and/or boredom or habituation during later sessions did not bias our results. The requirement to simultaneously balance four different orderings with four different form sets resulted in a unique experimental configuration for each participant. For example, the four participants who used the discovery aids in the same order had four unique orderings of the set of forms with their attendant errors.

Dependent variables measured were the time taken to perform the verification task (maximum 15 minutes), the number of errors caught during the verification task (maximum 25 errors) for each condition, and the number of false corrections (i.e. fields that the user changed that did not contain an error). At the end of the fifth experimental session, we asked participants which of the error discovery techniques they preferred.

4.2.5 Results

Time for each condition is measured as the total time spent by a participant verifying the forms, seen in the left graph in Figure 4.3. The mean verification times were 632.6s for the control, 649.1s for the visual proximity technique, 681.9s for the typographic manipulation technique and 726.3s for the text-to-speech technique. Repeated measures ANOVA for time as a function of condition, order, and form set indicates that time differences are marginally significant ($F_{3,15} = 2.709$, $p = .056$). Order and form set were not significant.

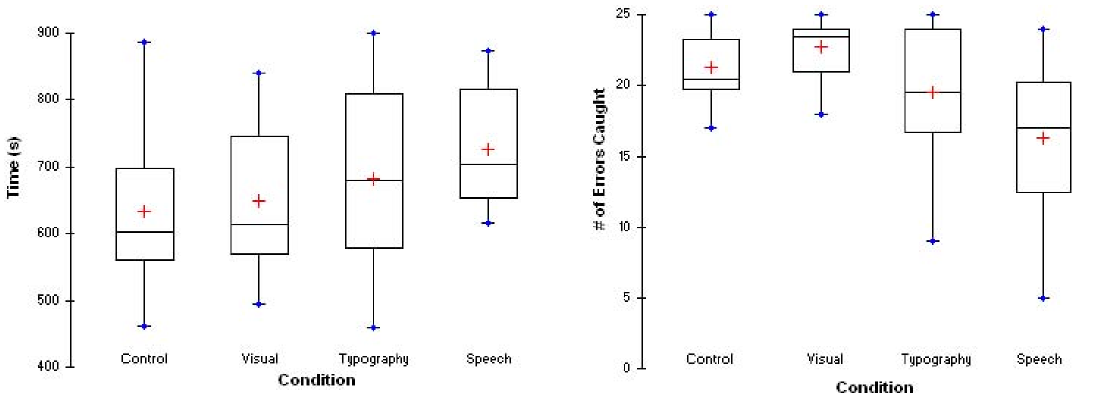


Figure 4.3: Performance results from the experimental evaluation. Left shows the time taken between conditions, and right shows the errors caught between conditions.

The left graph in Figure 4.3 shows the number of errors caught per condition. Repeated measures ANOVA indicate that the assumption of sphericity has been violated, $\chi^2 = 13.831$, $p = 0.17$, therefore degrees of freedom were corrected using Huynh-Feldt[1] estimates of sphericity ($\epsilon = 0.84$). The results show that there was a significant effect of error discovery aid used on the number of errors caught, $F_{2.52,37.798} = 10.526$, $p < .001$. These results suggest that the discovery aids used do have a significant effect on a user's performance in verification.

Because form set and order are not significant, paired t-tests with Bonferroni adjustment can be used post-hoc to contrast the number of errors caught based on the different error discovery techniques. As our goal is to understand the performance of our new discovery aids relative to a control condition, we performed three comparisons of each of our discovery aids against the control. Based on the results of these paired comparisons, we added a fourth condition that compared visual proximity to typographic manipulation. We report post-hoc testing using Bonferroni adjustment based on four comparisons: visual to control, typographic to control, text-to-speech to control, and visual to typographic.

The number of errors caught for the visual proximity technique (mean = 22.75, sd = 2.05) was found to be significantly different than that of the control (mean = 21.31, sd = 2.39), (corrected p = .047). There was no significant difference found in the number of errors caught between the control and the typographic manipulation discovery aid (mean = 19.56, sd = 4.79), (p = 0.23). The control condition was found to be significantly better than the text-to-speech condition (mean = 16.31, sd = 5.77) (p = .004).

[1]A statistical rule of thumb is that Huynh-Feldt correction, while more permissive, should be used when $\epsilon > 0.75$.

Because the visual proximity condition was found to be better than the control, and no difference was found between the typographic manipulation condition and control, a fourth comparison was added to compare visual proximity to typographic manipulation. The visual proximity technique was also found to perform significantly better than the typographic manipulation technique (mean = 19.56, sd = 4.79) (p = .037). The probabilities reported here have been adjusted to account for four comparisons.

False corrections for each condition are measured as any changes made to a form field that did not contain an error, and are displayed in Table 4.1. Capitalization changes and adding or removing spaces that did not change the meaning of the text were not counted in these false corrections. Inspection of the false corrections showed that the majority of them were caused by ambiguity in a participant's handwriting, causing them to misinterpret their own handwriting. No significance was found between the discovery aids and the number of false corrections made by the participants.

	Control	Visual	Typography	Speech
mean	1.19	1.31	1	0.75
sd	1.52	1.08	1.41	1.07

Table 4.1: False corrections for each condition

Finally, after the last session, we asked participants which discovery aid they preferred. Eight participants preferred the typographic manipulations, seven preferred the visual proximity aid, and one preferred the text-to-speech aid. Two of the participants who chose visual proximity as their preferred discovery aid stated that they also liked the control.

4.2.6 Performance of Error Discovery Techniques

Our experimental results provide contrasting results based on speed, error rate, and subjective preference. First, for mean time to complete the verification task, the control condition was fastest, followed by visual, typography, and speech. This result was surprising, as initially we had assumed that visual proximity would perform better than the control condition, with input ink and output characters spaced closer together. In analyzing our data, however, we noted that users were required to tab or click through questionable fields, and that this need to examine the fields individually slowed the users down. Despite the poor showing of our visual technique with respect to completion time, our visual technique was very effective in its ability to support error discovery. Participants were more accurate with the visual technique than with control.

The statistically significant difference between visual and control technique may be surprising, as the errors caught graph in Figure 4.3 shows little difference between the visual technique's performance (mean = 22.75, sd = 2.05) and the control technique's performance (mean = 21.31, sd = 2.39). However, because each participant uses both techniques, we can use a paired t-test which examines performance between matched participants. In all cases, participants did at least as well with the visual technique, and in most cases, their performance improved slightly. This consistent improvement in performance resulted in the statistically significant measurement. It is also interesting to note the strong performance of the control condition, both in terms of time and in number of errors caught. The control technique was the second most effective technique in mean number of errors caught, and outperformed speech by a significant margin.

The most popular technique for our participants was the typographic manipulation technique. Half of our participants preferred the typographic technique, despite the fact that it ranked third in time and in errors caught. When we asked the participants who preferred the typographic manipulation technique why they preferred it, they indicated that they liked this technique because of its ability to reduce ambiguities and its ability to divide the verification task into a set of smaller tasks.

Finally, the text-to-speech technique was clearly the worst of the techniques examined. It was found to take longer on average than the control condition, though not significantly so, and fewer errors were caught than with the control technique. We did expect that timing would be poor with the text-to-speech technique, as participants had to listen to recognizer output for each field in the form. Participants also stated that they found the voice very hard to understand; for example, one participant stated that it was like listening to somebody with a very heavy accent. This difficulty in understanding the voice produced by the text-to-speech engine may have caused the technique to perform worse than it otherwise would. Our own informal observations support this conjecture. With training, users become used to the intonations of the text-to-speech engine, and seem to go faster with higher accuracy than they do initially. It may be the case that the text-to-speech technique is more difficult to learn, but that, with mastery, it will perform on par with other techniques.

4.2.7 Limitations

Although the visual proximity aid proved successful in this evaluation, is is unclear which attribute of this aid is the most relevant to error discovery. Discussed further in Chapter 7, this discovery aid varies two factors in the design space: proximity and availability, yet the design of this experiment does not enable the effect of these individual factors to be understood. To disentangle the results of this technique, another study could be performed, similar in design to the experiment presented here, but with only two conditions. The first condition would display the recognized form and a digital version of the handwritten form side-by-side, where both the digital ink and recognized text are visible the entire time, yet visually separated. The second condition would statically display the handwritten ink above each text field containing the recognized text of that ink, creating a close proximity between the two. In each condition, the availability of the ink and text is static, but the proximity between the two conditions changes. This would allow the significance of proximity to be determined, and from that infer the effect of availability found in this evaluation.

During this study, 25 recognition errors were inserted into each set of 5 forms used by participants to test a specific condition. As noted earlier, these errors were selected from exploratory evaluations using the Microsoft recognizer bundled with the Windows XP Tablet PC SDK on a set of handwritten data forms. One concern, however, is how representative our error rate is of real world usage errors likely encountered by users of handwriting recognizer engines. It is possible that, if our error rate is too high, users may become habituated during correction with our techniques, and might begin to miss errors due to a larger number of errors being inserted. As well, if our error rate is too low, it may be that user confidence in the recognizer will be too high, and they will miss errors because they assume the recognizer is more than likely correct. Unfortunately, it is not completely clear how to properly balance error rates to account for these factors.

4.3 Summary

This chapter has presented two evaluations on the error discovery aids presented in the previous chapter. An exploratory evaluation on early designs was first presented, which revealed a number of issues in the initial designs of the discovery aids. Changes were made to the discovery aids. A subset of the most promising of these designs were selected based upon the results of our exploratory evaluation, specifically the visual proximity, typographic manipulation, and text-to-speech discovery aids.

A formal evaluation of the three selected discovery aids was presented, with the goal of evaluating the effects of the discovery aids on a user's performance in discovering errors in handwriting recognition. The evaluation used an out-of-the-moment verification task on handwriting recognition results for form-based data. It was found that, while there were no significant changes in the verification times between the discovery aid used, the visual proximity aid provided a statistically significant improvement in the number of errors found by participants over the control. No statistically significant difference was found in the errors discovered using the typographic manipulation aid, and the text-to-speech aid was found to perform significantly worse than the control.

The discovery aids evaluated in this chapter all represent techniques which have never been explored as a means of aiding users in the discovery of errors. While little work exists in error discovery for handwriting recognition, there has been one technique proposed in the literature for use as an error discovery aid – confidence highlighting. However, due to negative implications found with this technique in related research in speech recognition [20], explained further in the following chapter, we have avoided the inclusion of this discovery aid in our evaluations.

Although these negative implications exist, confidence highlighting has never been evaluated as a discovery aid for handwriting recognition, without which no real conclusions can be drawn. The next chapter examines confidence highlighting, and its potential for use as an error discovery aid in handwriting recognition.

Chapter 5

Confidence Highlighting

Confidence highlighting, the highlighting of recognizer output with low confidence values for the purpose of bringing a user's attention to that output, is the only proposed technique found in the literature which aims to aid users in error discovery [12]. While the use of confidence values has been described in the design space, the work presented in this thesis so far has not considered the use of confidence highlighting.

A goal of the research presented in this thesis, discussed in the introduction, has been to identify methods of aiding users to discover errors in handwriting recognition to reduce the amount of residual error left behind after verification. However, implications from related research in speech recognition show that confidence highlighting may have a negative impact on user performance in error discovery, due to users being more likely to miss errors that are not highlighted. If the goal is to minimize residual error, such as in the health care scenario motivating this work discussed in the introduction, confidence highlighting is not an ideal discovery aid.

However, applications likely exist which allow some tolerance for error in their input. In spite of the negative implications of the use of this discovery aid, in applications where a certain degree of residual error is acceptable, the confidence highlighting technique could still be useful.

This chapter takes a closer look at the confidence highlighting technique as an error discovery aid. First, the issues in using confidence highlighting are discussed, examining when confidence highlighting is an appropriate error discovery technique to use. Next, an evaluation of the confidence values given by the Microsoft handwriting recognizer is presented, finding that approximately 4% of recognition errors are flagged with a strong confidence, and would not be highlighted using confidence highlighting. The implications of these findings are then discussed with respect to the use of confidence highlighting as an error discovery aid.

5.1 Using Recognizer Confidence Values

The effectiveness of confidence highlighting is impacted by two values that characterize recognizer accuracy: the number of false positives, and the number of false negatives. False positives in confidence highlighting occur when a correct recognition result is given a low confidence value by the recognizer, and is therefore highlighted by the confidence highlighting technique. False negatives, on the other hand, occur when an error in recognition is given a high confidence value by the recognizer, and is therefore not flagged by the confidence highlighting technique. Both false positives and false negatives have potential impact on the effectiveness of confidence highlighting as an error discovery aid.

False positives can affect a user's verification accuracy by increasing the number of values that the user must explore. This has two possible effects on verification. The first is that the increase in the values that must be explored increases the amount of time that the verification task will take. The second, more dangerous effect is the possibility of habituation due to false positives. If a user becomes used to false positives being present, they may start to skim over highlighted results without close scrutiny, causing them to miss errors.

False negatives, errors that are not highlighted, have more serious drawbacks. While confidence highlighting may aid users in the discovery of recognition errors with low confidence values, it may hinder their ability to discover errors that have a high confidence. Previous study of confidence highlighting in a speech recognition interface has shown that, while it increased user performance in discovering highlighted errors, it decreased their performance in discovering non-highlighted errors [20]. This implies that the use of confidence highlighting as a discovery aid may result in false negatives being missed by users in verification, and left as residual error.

While confidence highlighting is likely to suffer from a residual error rate approximate to the false-negative rate, for applications where a certain amount of residual error is acceptable, confidence highlighting may still be useful as an error discovery aid. Confidence highlighting is a classic speed/accuracy trade-off: by highlighting areas that are more likely to contain errors, a user can more quickly discover and correct these errors at the expense of missing errors that are not highlighted. The benefit of speeding the verification task may be worth the cost of residual error.

However, while a designer may be willing to accept some trade-off resulting in increased residual errors, there must be some knowledge of the potential cost. To determine the cost of confidence highlighting, one must examine the number of recognition errors that receive a high confidence, resulting in a false negative. As these unflagged errors are the weakness of this technique, this rate will give an idea of how many residual errors may result.

Unfortunately, for the Microsoft handwriting recognizer, there exists no published data on the recognizer's recognition rate or confidence accuracy. Without knowledge of how well a recognizer works, no informed decision can be made for the use of confidence highlighting. To determine when confidence highlighting is an appropriate error discovery aid to used, based on what residual error rate is acceptable using the Microsoft handwriting recognizer, an evaluation on the accuracy of confidence values from the Microsoft recognizer was performed, described below.

5.2 Evaluating Recognizer Confidence Values

5.2.1 Method

To test recognition accuracy, a set of forty forms were selected, filled out by ten of the participants from the experimental evaluation in Chapter 4. For each form, an XML document was created containing the recognizer result of the digital ink, the intended input for each field and whether or not an error was present. A program was then created which read in both the digital ink and XML document, comparing the recognizer output of each handwritten word with the correct recognition result from the document, recording for each word if the recognizer made an error as well as the confidence value attributed to the recognition result.

5.2.2 Results

Overall, the recognizer examined over 2100 individual words, numbers and alphanumeric sequences to generate recognition accuracy results. The overall error rate for the Microsoft recognizer was approximately 23%. Table 5.1 breaks down the error rate with respect to the rate of assigned confidence values by the recognizer.

The Microsoft recognizer assigns confidence values to recognition results at three different values: strong, intermediate and poor. Approximately 46.5% of the recognition results returned in this evaluation were flagged with poor confidence. Of these, 41% were recognition errors, accounting for approximately 19% of the total error rate. Recognition results flagged with intermediate confidence values accounted for approximately 3% of all recognition results. Of these, 31% were recognition errors, accounting for approximately 1% of the total error rate. Finally, approximately 50.5% of the recognition results were given a strong confidence value by the Microsoft handwriting recognizer. Approximately 6% of the results given a strong confidence were recognition errors, which accounts for approximately 4% of the total error rate, rounding up.

5.3 Implications for Design

This recognizer data gives an interesting insight into the usefulness of confidence highlighting as an error discovery technique. If, using the Microsoft recognizer, poor confidence results were highlighted, most of the recognition errors would be

Confidence Value	Proportion Flagged	Error Rate
Strong	50.5%	6%
Intermediate	3%	31%
Poor	46.5%	41%

Table 5.1: Confidence values from a selected sample of forms filled out by participants. Amount Flagged represents the percentage of all recognized words flagged with a specific confidence value and Recognition Errors represents the percentage of words flagged with a specific confidence values that contain a recognition error.

flagged. If users corrected all highlighted errors and missed all non-highlighted errors, the residual error rate would be approximately 4%.

The error discovery techniques presented in this thesis can be viewed as complimentary to confidence highlighting. Using our results from Chapter 4, the expected performance for systems that use confidence highlighting and the visual proximity discovery aid together can be extrapolated. For example, the on-demand visual technique in the study allowed participants to identify 91% of the errors in the form set they examined (22.75 of 25 errors on average). Combining the visual technique with confidence highlighting might allow a user to perform to a similar level, catching 91% of flagged errors. In a worse-case scenario users may miss nearly all unflagged values. Given this, the residual error rate would then be approximately 5.8%. However, if the visual proximity technique was used and confidence highlighting was not used, the user might still identify 91% of recognition errors, yielding a residual error rate of 2%, assuming an overall error rate of 23%.

Confidence highlighting allows users to direct their attention toward recognition results more likely to be wrong, and to ignore recognition results that are more likely to be correct. The premise is not that all errors made by the recognition engine will be flagged – all researchers are aware that false positives and false negatives will occur. Instead, the goal is efficiency, i.e. users will identify most of the recognition errors faster and with less effort. What this analysis shows is that, as part of the cost of halving the amount of recognizer output that the user verifies (being able to focus on 46.5% of the values) the residual error rate will increase slightly. If a system makes use of the Microsoft recognizer and a 5-6% residual error rate is acceptable, confidence highlighting combined with the visual proximity aid would be a worthwhile technique.

There are many caveats to the numerical values calculated above. Another recognizer with better precision (fewer false positives) might provide different numbers, as might a recognizer with better overall recognition accuracy. The Microsoft recognizer was the chosen platform for evaluation because it is widely available, and this analysis provides some understanding of how error discovery techniques might combine with confidence highlighting in real world scenarios.

5.4 Summary

This chapter has examined the use of confidence highlighting to determine when it would be appropriate to use as an error discovery aid. An evaluation was presented on the accuracy of confidence values produced by the Microsoft handwriting recognizer, showing that approximately 4% of results flagged with strong confidence contain error, which would be left unhighlighted using the confidence highlighting aid. Based on implications from previous research, as well as experimental results from Chapter 4, this would imply that a 5-6% residual error rate could be expected from users making use of this technique.

Chapter 6

Measuring User Performance in a Verification Task

All of the work presented in this thesis up to this point has looked at the problem of aiding users in error discovery with the goal of reducing the number of residual errors left by a user after verification. However the residual error rate can vary greatly between users, and even a single user's performance in error discovery can vary over time due to factors such as boredom, habituation, fatigue or distraction. Without some knowledge of a user's performance in verification, it can be difficult to have a high confidence in the correctness of recognized results verified by them. If the performance of a user is known, on the other hand, measures can be taken to attempt to improve that user's performance, and a level of confidence can be gained in the correctness of verified results.

This chapter introduces a method of controlled injection of errors into recognizer output as a means to measure and track a user's performance in error discovery during a verification task. An experiment is presented to evaluate this concept, and it is found that error injection can be used as an effective measure of performance at several levels of granularity. The chapter concludes with a discussion of the implications of these results, as well as limitations in the design of the study.

6.1 Injecting Artificial Errors to Measure Performance

In the experimental evaluation of the three error discovery techniques in Chapter 4 it was found that while participants completed the verification task in the shortest time with the control condition, they found significantly more errors with the visual technique. It was also found that one of the reasons the visual technique was not faster than the control condition was that users had to tab or click through every field on a form and verify ink entry by entry. This slower field by field verification possibly led to more errors being caught.

It seems likely that any error discovery and correction task may have speed versus accuracy trade-offs, and that any technique created to aid users in this task will in some way play off of these trade-offs. An error discovery aid may attempt to decrease time taken in verification at the expense of errors caught, as with the confidence highlighting aid, increase errors caught at the expense of time taken, as with the visual proximity aid, or attempt to balance the two. However, these trade-offs may be confounded by fatigue, habituation, boredom, and distractions, particularly if the verification task is lengthy, which could cause a user's performance to vary throughout the verification task.

Beyond variations in a single user's performance, the performance in error discovery between different users can be expected to vary significantly. Some users may be more careful than others in verification, catching more recognition errors present, while others may rush verification, missing significantly more errors.

Both of these within-user and between-user variations in performance can be expected even when using discovery aids shown to provide a benefit to users in verification. This can make it difficult to accept with a very high confidence that a user's verified results are sufficiently error-free. In an application where a high accuracy of input is required, confidence in verified results is necessary.

If a measurement of a user's performance could be obtained, even an approximation, this issue of confidence could be addressed. Knowledge of how a user is performing during the task could be used to cause the system to respond to a user performing poorly to attempt to improve their performance. Also, knowledge of a user's overall performance after verification could give a degree of confidence in the number of residual errors left behind by that user.

While it is impossible to determine how many recognition errors a user is missing while verifying real handwriting recognition results, without prior knowledge of what the results should be, it could be possible to determine user accuracy by injecting artificial recognition errors into the real recognition results. Though counter-intuitive to solving the problem of errors in handwriting recognition, the insertion of a controlled number of errors into recognition results would allow a verification interface to monitor how many of the artificial errors a user is correctly identifying and correcting. This in turn could be used to get an approximate idea of how well a user is performing over time. This measurement could then be used to adapt the interface to increase a user's performance. For instance, if a user is found to be performing poorly, decrease the amount of recognition results available for verification at a time to slow the user down, which in turn would cause them to be more careful. Finally, a measurement of the user's performance could be used to get an approximation of the number of residual errors the user has left behind.

6.2 Evaluation

To determine if this method of injecting artificial errors into recognition results is a feasible method of obtaining an approximation of user performance in error discovery, an experimental evaluation of the technique was performed. As a result of this evaluation, it was found that a correlation exists between real recognition errors caught and injected errors caught, making it a suitable technique for measuring the performance of a user in error discovery.

6.2.1 Method

The experiment involved a single verification task on handwriting recognition results for completed forms. Participants were given a set of twenty paper forms, which were previously filled in using an Anoto pen, and were asked to verify, as quickly and accurately as possible, the handwriting recognition results shown to them on a computer. The application used to display the recognized results was the same as was used in the error discovery techniques experiment for the control condition, seen in Figure 4.2. Participants were not made aware that errors were being artificially injected into the recognition results.

As in the experimental evaluation of the three error discovery aids, there was also some need to simulate a real-life scenario where the user is trying to verify and correct the recognition results, but discouraged from taking too long in the task and catching more errors as a result. In the previous experiment there was a time constraint to simulate this condition with participants. However, a concern arose during the design of this experiment that a better method to motivate participants to perform in this kind of scenario would be to provide incentives based on performance. Instead of constraining the task, inform participants that two of the participants with the top performance on the task, based on a combination of speed and accuracy, will receive some reward.

As it is unclear which method of motivation is most effective, participants for this experiment were split into two condition groups; one that had a 3 minute time limit for the verification of each form, and one in which participants were told that the top two performing participants would receive an additional $15 at the completion of the study. By creating two participant groups with two motivational conditions, it would be possible to compare the performance of the two groups to get an idea of the effect of the two separate motivators. Though this comparison is not directly related to the purpose of this experiment, it can be used to validate the design of the experimental evaluation in Chapter 4.

6.2.2 Injecting Error

The main problem with creating artificial errors to inject into recognition results is making the visibility of the artificial errors equivalent to that of real errors. If the artificial errors are more or less visible than real recognition errors, the chances of a user discovering an artificial error changes in relation to a real error. This would make the use of injected errors useless in determining a user's true performance in error discovery.

With this in mind, the best resource for creating a false error for some recognized result would be from the n-best list from the recognizer for that result. Any result from the n-best list of a recognizer would be similar to any real recognition error in visibility, and thus has a similar likelihood of being discovered by a user. However, in using results from the n-best list, another possible problem occurs: what if the false error selected is the correct recognition result? When selecting fields to inject errors into, it is impossible to know whether the recognized result that is being written over is correct or not, and if it is not correct, the correct result may reside in the n-best list. An analysis of forty of the forms from the experimental evaluation in Chapter 4 showed that, of the real recognition errors present in the recognized results of those forms, 35% had the correct recognition result somewhere on the n-best list. If a correct recognition result is injected over a recognition error, a user will have no reason to perform any correction on that result, as it will appear correct to them. This could cause an incorrect analysis of how well the user is performing, as well as cause a real recognition error to be missed.

A more detailed analysis of the n-best list results for those forty forms was performed. It was found that, of those 35% of errors with correct results on the n-best list, 13% had the correct recognition result on the 2nd position of the list, 10% on the 3rd, 4% on the 4th and 8% on the 5th or lower. Knowing this, by skipping the first four results on the n-best list, an 8% chance is left that, if you are injecting an error over an existing error, the correct recognition result exists somewhere on the rest of the list. This is an acceptable chance, and is the method used in this experiment. In a real-life use of this technique, this problem could be further addressed by bringing to a user's attention any injected errors that they missed before they accept the verification as correct, in case the injected error is the actual correct recognition of the ink.

Another issue caused by selecting artificial errors from the n-best list comes when attempting to inject errors over a correct recognition result. Frequently the n-best list will contain results that are just variants of the top recognition result, with capitalization differences. To address this, any results from the n-best list was not a candidate for injection if it only contained capitalization differences, and the next result on the list was considered.

6.2.3 Apparatus

The forms used in this experiment were selected from twenty of the forms filled out by ten of the participants in the Chapter 4's experimental evaluation. Forms were selected based on the legibility of the writing on the form, to avoid the issue of recognition errors being missed due to the participant being unable to properly interpret the original handwriting. Unlike the previous experiment, however, the recognition results presented to the participants were real results from the Microsoft handwriting recognizer.

In each recognized form, ten artificial errors were randomly injected into the results. To avoid the possibility of grouping injected errors, in which a participant may catch all artificial errors because they are closely grouped together, and in case participants were more careful in certain parts of the form therefore catching all errors in those areas, the ten errors were spread out proportionally among the three different blocks of each form. Two errors were randomly injected into the materials block, five into the Ikea block, and three into the Future Shop block.

6.2.4 Participants

12 volunteer participants were recruited from the university campus (7 female, 5 male). Their ages ranged from 18-38 (mean = 22.5 , sd = 5.4). They were paid \$15 (CDN) for participating in the experiment, with the top two performing participants of the incentive block being given an additional \$15.

6.2.5 Results

The results of the experiment are summarized in Table 6.1. An error occurred in the experimental program during P10's experimental session, causing some of the data from that session to be invalid. All of P10's data was excluded from analysis.

Condition	Participant	Average Completion Time (s)	Average Injected Errors Caught (%)	Average False Errors Caught (%)
Timed	P1	143	0.951	0.965
Timed	P2	196	0.792	0.825
Timed	P3	137	0.607	0.625
Timed	P11	179	0.941	0.925
Timed	P12	138	0.791	0.82
Untimed	P4	307	0.96	0.916
Untimed	P5	176	0.967	0.935
Untimed	P6	257	0.967	0.925
Untimed	P7	185	0.97	0.96
Untimed	P8	242	0.945	0.885
Untimed	P9	211	0.961	0.98

Table 6.1: Summary of the experimental results, giving the average performance results for each participant.

Correlations between the number of recognition errors caught versus the number of injected errors caught were measured at a several levels of granularity: as a

Granularity	Correlation		
	Timed Group	Untimed Group	Combined
Individual Forms	0.58	0.11	0.59
Rolling Window	0.84	0.24	0.84
Participant Average	0.99	0.67	0.98

Table 6.2: Correlations between recognition errors caught and injected errors caught at different levels of granularity. The correlation between the two participant groups are shown separately, and combined.

measurement between the two for each form verified across all participants, as a four-form average rolling window average across each participant, shown on the left in figure 6.1, and as the average for each participant, shown on the right in Figure 6.1. These correlations are summarized in Table 6.2.

Some significant differences were observed in the performance of participants between the two conditions. Participants in the timed condition took an average of 159 seconds per form, and 229 seconds in the untimed condition. This makes an average difference of 70 seconds taken to complete the verification of a single form between the two groups. The number of recognition errors caught between the two also differed significantly, with participants in the timed condition catching on average 82% of real errors per form, and participants in the untimed condition catching 96% on average; a difference of 14% more errors caught in the untimed condition than the timed condition.

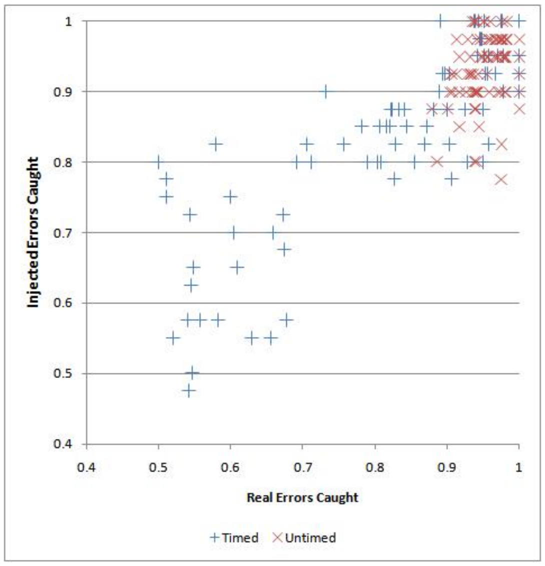
The differences in performance between the two groups also impacted the correlation between real and injected errors caught between the groups. As can be seen in Table 6.2, the correlation between real and injected errors caught is much lower for the untimed participant group than it is for the timed participant group at each level of granularity. The reason for this difference will be discussed in more detail below.

6.3 Discussion

6.3.1 Characterizing User Accuracy

The results of this experiment show that it is possible, using this method of error injection, to extract an approximation of a user's performance in a verification task, without the need of oracular knowledge in where the real errors exist. Moreover, by looking at the accuracy of a user in discovering injected errors over longer periods of time, this approximation can become much more accurate, to the point of almost perfectly characterizing a user's performance. First, though, the difference in correlations between the two participant groups must be considered.

As can be seen in Table 6.2, at each level of granularity the correlation between real errors caught and injected errors caught is much lower for the untimed par-



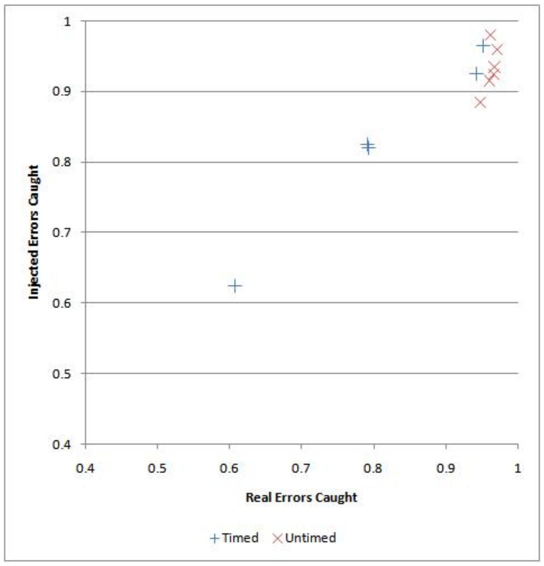


Figure 6.1: Injected errors caught vs. real errors caught as a rolling window average (left) and as an average over the entire verification task for each participant (right). Data points between the timed and untimed participant groups are labeled differently.

ticipant group than the timed group. This seems to imply that the lack of a time constraint caused a difference in how likely a user was to discover an artificial error over a real error in recognition. Another likely explanation, however, is that the number of errors injected in this experiment were not adequate to characterize the performance of consistently high-performing users.

In the untimed condition, participants were consistently discovering a high percentage of the errors present in each form, yet they still missed a small number of errors, which can be seen in Table 6.1. Because of the number of errors injected, ten per form, if a user is discovering over 90% of the errors present and more than one of the errors missed is an artificial error, the rate of real versus injected errors caught will differ significantly. This creates an outlier in the data set which will reduce the overall correlation. Looking at the left graph in Figure 6.1, this seems to be the case, as most of the variance away from a one-to-one correlation is caused by data points with high rates of real errors caught, but lower rates of injected errors caught. As the entire untimed participant group consistently discovered a large percentage of the errors present, these outliers changed the correlation substantially.

However, when the rates of injected errors caught versus real errors between both participant groups are combined, these outliers do not affect the overall correlation, as can be seen in Table 6.2. This shows that, as a measure of average performance, error injection is still a useful method of measuring user performance.

The correlation between injected errors caught and real errors caught on a form-to-form basis was found to be 0.59. While this is not a high enough correlation to be relied upon to portray a user's performance to high accuracy, it should be noted that this is a correlation on 220 data points, meaning that it is a significant correlation. This shows that the performance of a user can be characterized to some accuracy on a real-time basis.

The four-form rolling window, however, shows a much higher correlation between the two values, at 0.84, over 187 data points. This shows that, using this technique, it is possible to get a much more accurate characterization of a user's performance at any given time by looking at their average performance in discovering injected errors over some window of previously verified forms. A window of four forms was looked at in this experiment, which is equivalent to around a ten minute time span, though that could be varied depending on the desired accuracy in measurement.

Finally, this technique has proved extremely effective in characterizing users' performance on the entire verification task, with a correlation of 0.98. This could be valuable in summarizing how well a user has performed after the verification task has completed. Additionally, another valuable use of this performance characteristic would be to obtain an approximation of how many residual errors remain in the data verified by the user, a number which could be further refined with knowledge of the average error rate of the recognizer used.

6.3.2 Trade-off in Speed vs. Accuracy

The differences in performance between the two condition groups in this experiment are a perfect example of the trade-off that comes in focusing on speed over accuracy. Both participant groups were given motivation to complete their verification tasks as quickly as possible, while still maintaining accuracy in verification: the timed group by use of the time limit constraint and instruction to find all the errors, and the untimed group by use of the incentive for performing quickly and accurately. However, without the time constraint, the untimed participant group took more time on verification (70 seconds per form, on average), and as a result discovered more errors (14%).

In the experiments here and in Chapter 4 users sped up their verification of each form to be able to verify the entire form before the time ran out, understandably leaving a higher residual error rate. Though the imposed time constraints are artificial in these experiments, in real-world use of a similar system, some kind of time constraint could be expected on the completion of a verification task. Users would similarly perform the verification task quickly in order to meet any time constraint, rather than taking time in the verification task to ensure greater accuracy in verification, as seen in the untimed condition. In this sense, the artificial constraint used in these experiments accurately portrays time constraints that would occur in real-world usage.

6.3.3 Study Limitations

The results of this study have shown that it is possible to measure, to varying degrees of accuracy, the performance of a user in discovering errors in a verification task through the use of error injection throughout that task. It does not, however, show how this technique should be implemented and used in a real-world application. There are a number of factors that still need to be explored:

1. Ratio of errors per form field needed to achieve a strong correlation, while keeping the number of injected errors low.
2. Error injection's effect on user performance.
3. How to appropriately handle injected errors left uncorrected by users.
4. User acceptance of error injection, a technique that intentionally creates more errors, creating more work.

In addition, while this technique is useful in identifying when a user is performing poorly, injecting errors into the results of a user who is performing well would be less useful, as it would only serve to make more work for that user, slowing them down in their task. The number of errors injected could be varied during the course of the verification task based on the performance of the user. When a user is performing poorly, the number of errors injected is increased to an amount that will give a more precise measurement of their accuracy to enable the system to react accordingly. When a user is performing well, the number of errors injected could be decreased to enable the task to be completed faster. Of course, some baseline artificial error rate should remain, to enable the system to determine if a user's performance degrades, possibly due to distraction or fatigue.

6.4 Summary

This chapter has presented a method of measuring a user's performance in error discovery throughout a verification task by means of injecting artificial error into the recognizer's output. An experimental evaluation of this technique was described, which showed that a correlation exists between real recognition errors caught by a user and injected errors caught, which grows stronger when observed at different levels of granularity.

There are a number of factors that can affect user performance in verification, making it difficult to gain confidence in the correctness of verified results from even the most diligent user. In an application where high input accuracy is necessary, such as recording patient medical information in a health care environment, it is not enough to assume accuracy, since humans as well as recognizers are prone to failure. A performance measure, such as the use of artificial errors, addresses this issue of confidence by giving a means with which to respond to poorly performing users to attempt to increase their performance, as well as gain an approximation of the number of residual errors left by a user.

Chapter 7

Implications for Design

There are a number of implications for the design of an error discovery interface which can be derived from the work presented in this thesis. This section discusses these implications. We first discuss when error discovery aids are necessary in a verification interface. Next, results of the typographic manipulation aid from Chapter 4's formal evaluation are examined, and compared to related research in confidence highlighting, focusing specifically on the implications of attempting to guide a user's attention in a verification interface. Next, the importance of proximity and availability of digital ink and recognized text is discussed. Finally, the error injection technique is discussed, and implications of how it can be used to adapt to user performance are examined. First, however, it is discussed when the use of error discovery aids are necessary in a verification interface.

7.1 When are Error Discovery Aids Necessary?

Performance metrics in HCI generally use some combination of time, error rate, and user satisfaction to contrast competing systems. One challenge with the results of our formal evaluation of the error discovery aids is that the control, the visual technique, and the typographic manipulation technique each perform well using one of these metrics. However, there are implications for the design of error discovery aids in these results.

The effort required to engineer error discovery aids involves a trade-off between the time taken to create and properly implement these aids, and the benefits they provide to the recognizer verification task. In some situations, residual errors may be very costly. Examples of domains where residual errors are costly include health care, scientific data collection [23], and first responders [6]. In these health, safety, and research domains, the integrity of the information allows subsequent users of the data to act appropriately by prescribing, deploying, or analyzing. However, other domains, such as classroom notetaking and early design work, do not require error-free recognition. In these domains, the information being recognized was recorded and is consumed by the same person, rather than being shared with others. Between these two extremes, there are various levels of residual error that are permissible.

As seen in the experimental evaluation of the error discovery aids, the control interface, which made no use of any discovery aids, performed very well in the error discovery task. While it may not allow users to identify every recognition error, in applications where it is not critical to have highly error free data, it may not be worth any of the extra effort required to design error discovery aids, as well as any extra efforts required by the user to use any aids. Other error discovery aids were neither faster nor slower, and it had the second best error identification rate. Depending on the application, and the residual error rate acceptable for that application, the use of error discovery aids may be unnecessary.

7.2 Attempting to Guide a User's Attention

The poor performance of typographic manipulation echoes the implication from Vertanen and Kristensson's [20] evaluation of confidence highlighting in speech recognition that using confidence values to flag data draws users' attention away from other errors. Visual manipulations do highlight a certain class of error, those amenable to automatic detection, but the result of this visual augmentation is that users find unhighlighted errors more difficult to identify.

Two additional pieces of information support a concern with visual highlighting. First, with the typographic manipulation technique, participants could choose to accentuate numbers, letters, non-alphanumeric symbols, or none of the above. In the case where no characters are accentuated, our typographic interface was identical to our control condition, yet the typographic augmentations made error discovery worse than the control condition. Second, participants thought that the visual manipulation technique reduced ambiguities, but the reduction in ambiguity for a certain class of error came at the cost of participants detecting fewer errors overall.

The implication for design in this is that augmentation techniques, including typographic manipulations, confidence highlighting, and other discovery aid that might highlight likely errors must be used with care in error discovery aids. While they may aid users in the discovery of a certain class of errors, they come at the cost of an increased residual error rate for the recognition errors which fall outside that specific class.

7.3 Proximity and Availability

The visual proximity technique was found to be successful in aiding users to discover a greater proportion of errors in recognition results over the control in the formal evaluation presented in Chapter 4. The positive performance of this technique has some interesting implications for the use of proximity and availability in creating effective designs of verification interfaces for handwriting recognition.

First, the technique creates a close proximity between the recognized text and the corresponding digital ink, allowing users to perform a quick comparison between the two. This may allow users to identify less noticeable errors that may have been otherwise missed if there was more of a visual separation between the two. For example, if a number was recognized that contained a large number of repeating numbers (e.g. '111000011') and one of the repeating numbers was not recognized, this could easily be missed by a user when glancing back and forth between the screen and a paper form. However, by presenting the ink in close proximity to the recognized text, a user can perform a one-to-one comparison of the individual numbers very quickly. This implies a significant benefit in creating a close proximity between ink and recognized text, and should be considered when designing a discovery interface.

Second, the visual proximity technique reduces the availability of the digital ink to only allow the ink for a single field to be viewed at one time. By providing the ink information on-demand, users were slowed down in the verification task and forced to consider each field individually, rather than allowing them to consider multiple fields simultaneously. This technique does create a speed-for-accuracy trade-off, though a small one (only 17 additional seconds, on average, over the entire ten to fifteen minute task). In an application where verification accuracy is essential, this trade-off may be acceptable.

7.4 Adapting to User Performance

The error injection technique was found to be an effective method of measuring a user's performance, both in real-time and to a higher accuracy when measured over a longer period. While this technique has the drawback of creating more work for the user, making the verification task more difficult, the benefits of how this measurement can be used may outweigh that issue.

First, the real-time and near real-time (through use of a rolling window average) knowledge of a user's performance could be an invaluable tool to improve that user's performance. Using this measure, it is possible to tell if a user is performing poorly, and react accordingly. A simple response to poor performance, for example, would be to warn a user that errors still exist in their verified results, forcing them to re-verify. If a user was aware that the system would force them to repeat their work when they perform poorly, they may take more care in verifying results. Alternatively, an adaptive interface could be used that reacts to a user's poor performance by changing the error discovery technique used in the interface; for instance, reducing the amount of recognized results that can be verified at a time.

Another interesting idea is to present this performance measure to the user to allow that user to customize the verification interface to maximize their performance.

For example, if a verification interface had a set of discovery aids that could be used in its interface, rather than having the system manage the use of these aids, allow the user to enable or disable them. If a user was shown their performance during the verification task, they could then customize the discovery aids used to a set which improves their performance the most.

Second, this real-time knowledge of a user's performance could be used to imply the state of the user. For instance, if a user has been performing well in their verification task, and suddenly their performance rate drops off, it may just be that something distracted the user during the verification of that form, and no adaptive action is immediately necessary. Instead, a warning to the user may be suitable to regain their attention. However, if a user's performance is gradually degrading over time, it could imply that user is becoming bored or fatigued, and some kind of adaptive action is required.

Finally, knowledge of the user's average performance in discovering injected errors over a verification task on many forms can be used to give confidence in the results verified by that user. Using the performance measure, along with knowledge of the average error rates for the recognizer being used, it would be possible to determine an approximation of the residual error rate left in the verified forms from that user. If that error rate is above the rate deemed acceptable, those forms could be flagged for re-verification.

7.5 Summary

This chapter has presented a set of implications for the design of error discovery interfaces derived from the work presented in previous chapters of this thesis. First, the results of user performance in the control condition of the formal evaluation were examined, showing that in applications where a moderately low degree of residual error is acceptable, the use of error discovery aids may not be necessary. The performance of the typographic manipulation technique in the formal evaluation was next examined, and compared to implications in the use of confidence highlighting on a user's performance in error discovery. Implications were drawn about the potential dangers in attempting to guide a user's attention to a certain class of error. Next, the positive performance of the visual proximity technique was examined, finding implications that proximity and availability of digital ink and recognized text can make a significant difference in a users performance in error discovery. Finally, the performance measurement presented in Chapter 6 is discussed, and implications are drawn about how this measurement can be used in a verification interface.

Chapter 8

Conclusion and Future Work

This thesis has presented work which represents the first researched approach to creating an interface that will aid users in the discovery of handwriting recognition errors in an out-of-the moment verification task. This thesis has presented proposed interface designs for this problem, experimental results on these designs, as well as additional experimental results which provide insights into how these designs could be used. This chapter will review the contributions made by this thesis, and propose future research that arises from the work presented here.

8.1 Research Goals

Handwriting recognition, being an ambiguous form of input, will never achieve perfect accuracy. Because of this, human verification of recognizer output is necessary for handwriting recognition to be used in applications which require input accuracies greater than can be provided by a recognizer. Modern handwriting recognition systems have high enough error rates to make a verification step necessary to allow any degree of confidence in the output of the recognizer. Most research into the area of handling errors in handwriting recognition has viewed the verification process as an in-the-moment problem, making the discovery of recognition errors by users a smaller problem, and have instead focused on aiding correction of errors. However, with technology such as the Anoto pen, out-of-the-moment verification of recognition results takes place, making the problem of error discovery much more difficult. This thesis addresses the problem of aiding error discovery in an out-of the moment verification scenario, with a focus on batch verification of handwritten form data.

The first goal of this thesis was to explore design alternatives for the creation of error discovery aids, techniques which attempt to aid users in discovering recognition errors. A design space was created laying out the possible dimensions which could be used in the design of an error discovery aid. From this design space a set of error discovery aids were created.

The second goal of this thesis was to identify methods which reduce the number of residual errors left by a user after verification. Evaluations were performed on the error discovery aids created to identify which of them allowed users to discover more errors in recognition results. An evaluation was also performed on confidence highlighting, the only error discovery aid presented in the literature, examining its potential as an error discovery aid. In addition, a method was proposed and evaluated to provide a measurement of user performance in error discovery, allowing verification interfaces to react to poorly performing users to reduce the number of residual errors left in verified results.

8.2 Contributions Revisited

This thesis has presented a number of contributions toward the problem of aiding error discovery in out-of-the-moment handwriting recognition.

Design space for error discovery aids

A design space was presented in Chapter 3 for the creation of error discovery aids for handwriting recognition. A variety of different factors are described which could be varied to affect a user's ability to discover recognition errors. Due to the lack of prior research on error discovery, this design space can be used as a tool to guide the design and exploration of additional error discovery aids.

Seven error discovery aid designs and evaluation

Designs for seven error discovery aids were presented in Chapter 3, and evaluated in Chapter 4. These designs, and the results from their evaluations, can be used to inform the design of future error discovery aids and verification interfaces.

Visual proximity technique

A number of designs for error discovery aids were presented in Chapter 3, including the visual proximity technique, and a set of these were evaluated experimentally in Chapter 4. The experimental evaluation showed that the visual proximity technique, which displays the handwritten digital ink on-demand in close proximity to the recognized text, provides users with a statistically significant increase in their ability to discover errors in recognition. This is the first method that has been experimentally shown to increase a user's performance in error discovery, and is a valuable first step toward the creation of an interface designed to aid users in the discovery of handwriting recognition errors.

Error discovery performance measurement

Chapter 5 presents the use of error injection into recognition results as a means of measuring a user's performance in error discovery, as well as an experimental evaluation of this technique showing that the technique correlates well with actual performance on error discovery tasks. The use of this measurement can be beneficial in allowing systems to adapt to poorly performing users, as well as to give confidence in a user's verified results.

Data on Microsoft's handwriting recognizer confidence values

The Microsoft handwriting recognizer, a high-performance recognizer that is readily available to developers, has very little published data on its performance, making it impossible to make informed decisions on how to properly use data from it. Chapter 5 presents data on the accuracy of the recognizer's confidence values, which can be used to make informed decisions on the use of confidence values, such as what residual error rate can be expected when using a confidence highlighting technique for error discovery.

8.3 Future Work

The work presented in this thesis has a variety of useful contributions, and represents a strong starting point toward creating interfaces which will aid users in the discovery of handwriting recognition errors. However, there is still a lot more work that can be done in this area, and a lot of research that can follow directly from this thesis.

8.3.1 Limitations of Evaluation

There are a number of findings from the work in this thesis that warrant further investigation. First, the success of the visual proximity technique is a valuable contribution, however the implications of the technique would benefit from further investigation. The visual proximity technique manipulates the availability and visual proximity of the handwritten ink to the recognized text, but further evaluation could be performed to gain a greater understanding of how these two factors affect a user's performance in error discovery. For example, is limited availability or proximity of ink and text more important to error discovery. The limitations section in Chapter 4 suggests an experimental design to disentangle the results of the visual proximity aid.

Second, while the text-to-speech discovery aid failed in the evaluation presented in Chapter 3, comments from the participants on the technique, as well as informal observations on use of the technique, show that it may be improved and used successfully as an error discovery aid. One of the main issues participants had with the technique was that the text-to-speech voice used was difficult to understand, which is a likely cause of the participant's decreased performance when using that technique. Informal observations of use of this aid have shown that, with some extended use of text-to-speech, it becomes significantly easier to identify errors in recognition. Re-designing this technique with an easier to understand synthetic voice, as well as further user training, could make this technique into a successful error discovery aid.

Finally, there were a number of error discovery aids that were presented in Chapter 3 that were not evaluated, to keep the scope of the formal evaluation manageable. All of these techniques have potential for use as error discovery aids, and warrant further evaluation. The techniques which employ highlighting of sections of ink could, however, benefit from improvements to their algorithms for selecting ink to highlight.

8.3.2 Further Study into Error Discovery Aids

The use of confidence highlighting, highlighting recognizer output with low confidence values, is the only suggested technique in existing literature to aid users in error discovery. While there are negative implications of the use of this technique, both in speech recognition [20] and corroborated by the results of the typographic manipulation technique in this thesis, it still may prove to be a useful tool for aiding error discovery. The analysis of confidence values in Microsoft's handwriting recognizer, presented in Chapter 5, show that even assuming that users miss all errors not flagged with low confidence, the residual error rate would still be relatively low. However, as no experimental evaluation of this technique exists for handwriting recognition, it still needs to be evaluated experimentally before assuming how it would affect user performance and residual error rate.

Beyond the techniques presented in this thesis, and proposed in previous work, there still exists a large number of possibilities for the design of error discovery aids. The dimensions of the design space presented in Chapter 3 have not been fully explored, and can be used as an effective tool for exploring the different design possibilities, and a number of design possibilities are suggested in the description of the design space.

8.3.3 Error Injection

Using error injection to measure the performance of a user in discovering errors, presented in Chapter 5, has some interesting potential for use. However, the work on error injection presented in this thesis only serves to show that it can be used as a performance metric. A number of factors still need to be explored, such as: what ratio of injected errors for amount of recognized results is necessary for a strong correlation; how error injection affects user performance; how to handle uncorrected error injections; and user acceptance of this technique. In addition, it may be possible to vary the number of injected errors based on user performance, to either reduce the number of errors for users performing well, allowing them to perform the task faster, or increasing the number of errors for users performing poorly, to obtain a better measurement of their performance.

In addition to these factors, there is also an open question of how to make the best use of the measurement of performance obtained by error injection. It could be possible to use this metric to adapt the interface based on the user's performance, changing the error discovery aids to best suit the user's level of performance. Or it could be used simply as a measure of confidence in the number of residual errors left by the user. Access to this performance measurement has great potential in improving and maintaining the residual error rate left by users, though there is more future work that needs to be performed to identify how best to implement and use it.

8.4 Summary

This thesis has presented work on the design and evaluation of techniques to aid users in the discovery of handwriting recognition errors. A technique has been presented which gives users a significant improvement in their ability to discover errors, and implications have been drawn from the results of the other designs for the creation of error discovery aids. In addition, a method to measure a user's performance in an error discovery task has been presented, which could be useful in the creation of future interfaces for the verification of handwriting recognition results.

Handwriting is, in many cases, a convenient form of input, and modern handwriting recognition systems are able to achieve very good accuracy rates. However, since handwriting recognition is prone to error, a verification step in which a user must discover and correct errors in recognition is necessary between input and acceptance of recognizer output to allow any confidence in the accuracy of the output, a problem which is made more difficult in an out-of-the-moment recognition setting. The work presented in this thesis is the first steps toward designing interfaces which will aid users in discovering errors in recognition that need to be corrected.

Appendix A

Study Letters and Forms

The studies presented in this thesis received approval from the University of Waterloo Office of Research Ethics.

This appendix presents copies of paperwork required by the Office of Research Ethics: the Information-Consent Letter and the Feedback letter for participants in each study.

A.1 Information-Consent Letter for Chapter 4 Formal Evaluation

David R. Cheriton School of Computer Science
University of Waterloo
Student Investigator:
Ryan Stedman (rstedman@cs.uwaterloo.ca)
Faculty Supervisors:
Professor Edward Lank (lank@cs.uwaterloo.ca)
Professor Michael Terry (mterry@cs.uwaterloo.ca)

Participant Information Letter and Consent Form

Overview

You are being asked to participate in a research study as part of a research project at the University of Waterloo. The intent of this study is to determine how effective a number of techniques are at aiding users in discovering errors in handwriting recognition.

This study is being conducted by Ryan Stedman under the supervision of Professors Edward Lank and Michael Terry.

Study Details

As a participant in this study, you will be asked to participate in five experimental sessions taking place on five consecutive days. In the first session you will be asked to copy ten forms given to you using an Anoto digital pen.

In the second session you will be asked to use a computer to verify the results of handwriting recognition performed on the forms you filled out. After verifying the forms, you will be asked to complete a questionnaire designed to determine your mental workload while performing the verification of the form. Afterward, you will be asked to copy another 10 forms in the same manner as the first day.

The third and fourth session will continue as the second, each day verifying the forms you filled out the previous day, except the fifth session, where you will not be asked to copy any more forms. However, each day you will be using a different application to verify the results with, each application displaying the results in a different manner.

At the end of the final session you will be given some open-ended interview questions to determine how you found the tasks you performed.

Each session will take approximately 60 minutes.

The verification tasks will be video recorded. Additionally, screen capture software will be used to record the actions you perform on the computer system used by you during the second session. These recordings will allow us to more easily recall the facts when we later analyze the information we have collected.

You may decline to answer particular questions, if you wish, and may withdraw participation at any time.

Risks and Benefits
There are no known or anticipated risks to you as a participant, other than those associated with the normal use of computers. There are also no known or anticipated benefits to you.

Confidentiality and Data Retention
All data collected is considered confidential. Codes, rather than names or other identifying information, will be used in notes and/or recordings. Video capture equipment will not be set up to capture face shots, as this can be used as identifying information. Any incidental images of your face will be removed. Your name or any other personal identifying information will not appear in any publication resulting from this study. However, with your permission, anonymous quotations may be used. Notes, images, and/or recordings collected during this study, with identifying information removed, will be retained for a period of 1 year in a secure location in the HCI research laboratory and then confidentially destroyed.

Remuneration
As a participant in this study, you will receive \$50. In the event that you cannot complete the requirements of the study, you will receive a prorated amount at the rate of \$10/per session.

Questions
If you have any questions about participation in this study, please feel free to ask the researchers. If you have additional questions at a later date, please contact Ryan Stedman via email at rstedman@cs.uwaterloo.ca, or Professor Edward Lank at 519-888-4567 ext. 35768 or via email at lank@cs.uwaterloo.ca, or Professor Michael Terry at 519-888-4567 ext. 34528.

This project has been reviewed by, and received ethics clearance through, the Office of Research Ethics at the University of Waterloo. In the event you have any comments or concerns resulting from your participation in this study, please contact Dr. Susan Sykes at 519-888-4567, Ext. 36005.

Consent Form
I agree to participate in a study being conducted by Ryan Stedman for a research project at the University of Waterloo. The faculty supervisors are Professors Edward Lank and Michael Terry. I have made this decision based on the information I have read in this Information-Consent Letter and have had the opportunity to receive any additional details I wanted about the study. I understand that I may withdraw this consent at any time by telling the student investigator.

I am aware that my actions during the verification tasks will be video recorded.

I am aware that screen capture software will be used to record my actions on the computer during the observation portions of the study.

I am aware that excerpts from the interview may be included in any publications to come from this study, with the understanding that the quotations will be anonymous.

I was informed that I may withdraw my consent at any time without penalty by advising the students or project supervisor.

This project has been reviewed by, and received ethics clearance through, the Office of Research Ethics at the University of Waterloo. I was informed that if I have any comments or concerns resulting from my participation in this study, I may contact the Director, Office of Research Ethics at the University of Waterloo at (519) 888-4567 ext. 6005.

With full knowledge of all foregoing, I agree, of my own free will, to participate in this study.

___ YES ___ NO

I agree to the verification tasks to be video recorded.

___ YES ___ NO

I agree to the use of anonymous quotations in any presentation or report that comes of this study.

___ YES ___ NO

__________________________ (Please print)
Name of participant

____
Signature of participant

____
Date

__________________________ (Please print)
Name of witness

____
Signature of witness

____
Date

A.2 Feedback Letter for Chapter 4 Formal Evaluation

Date

Dear Participant,

We would like to thank you for your participation in this study. As a reminder, the purpose of this study is to evaluate the effectiveness of a number of methods we have developed to help people find errors in handwriting recognition. The data collected through the experimental sessions will contribute to guiding future research and designs of handwriting recognition error discovery interfaces.

Please remember that any data pertaining to yourself as an individual participant will be kept confidential. Once all the data are collected and analyzed for this project, We plan on sharing this information with the research community through seminars, conferences, presentations, and journal articles. If you are interested in receiving more information regarding the results of this study, or if you have any questions or concerns, please contact us at either the phone number or email address listed at the bottom of the page.

If you would like a summary of the results, please let me know now by providing me with your email address. When the study is completed, I will send it to you. The study is expected to be completed by January 1st 2009.

As with all University of Waterloo projects involving human participants, this project was reviewed by, and received ethics clearance through, the Office of Research Ethics at the University of Waterloo. Should you have any comments or concerns resulting from your participation in this study, please contact Dr. Susan Sykes in the Office of Research Ethics at 519-888-4567, Ext. 36005.

Ryan Stedman
University of Waterloo
Faculty of Mathematics
Department of Computer Science
519-888-4567 ext. 38318
Email: rstedman@cs.uwaterloo.ca

A.3 Information Letter for Chapter 6 Study, Timed Participant Group

David R. Cheriton School of Computer Science
University of Waterloo
Student Investigator:
Ryan Stedman (rstedman@cs.uwaterloo.ca)
Faculty Supervisors:
Professor Edward Lank (lank@cs.uwaterloo.ca)

Participant Information Letter and Consent Form

Overview
You are being asked to participate in a research study as part of a research project at the University of Waterloo. The intent of this study is to determine user accuracy in discovering handwriting recognition errors.

This study is being conducted by Ryan Stedman under the supervision of Professor Edward Lank.

Study Details
As a participant in this study, you will be asked to participate in a single session. In this session you will be given a set of 20 handwritten forms, already filled out. You will be asked to use a computer to verify the handwriting recognition results performed on these forms, correcting any errors found. You will be give 3 minutes to complete the verification of each form.

The session should take approximately 90 minutes.

You may decline to answer particular questions, if you wish, and may withdraw participation at any time.

Risks and Benefits
There are no known or anticipated risks to you as a participant, other than those associated with the normal use of computers.

From this study we hope to discover a method of modeling how precise a user is in searching for errors in recognition, which could be used in future research and user interfaces to improve user accuracy when searching for errors.

Confidentiality and Data Retention
All data collected is considered confidential. Codes, rather than names or other identifying information, will be used in notes and/or recordings. Your name or any other personal identifying information will not appear in any publication resulting from this study. Data collected during this study, with identifying information removed, will be retained indefinitely in a secure location in the HCI research laboratory.

Remuneration
As a participant in this study, you will receive $15. In the event that you cannot

complete the requirements of the study, you will receive a prorated amount at the rate of $10/per hour.

Questions
If you have any questions about participation in this study, please feel free to ask the researchers. If you have additional questions at a later date, please contact Ryan Stedman via email at rstedman@cs.uwaterloo.ca, or Professor Edward Lank at 519-888-4567 ext. 35768 or via email at lank@cs.uwaterloo.ca.

This project has been reviewed by, and received ethics clearance through, the Office of Research Ethics at the University of Waterloo. In the event you have any comments or concerns resulting from your participation in this study, please contact Dr. Susan Sykes at 519-888-4567, Ext. 36005.

A.4 Information Letter for Chapter 6 Study, Untimed Participant Group

David R. Cheriton School of Computer Science
University of Waterloo
Student Investigator:
Ryan Stedman (rstedman@cs.uwaterloo.ca)
Faculty Supervisors:
Professor Edward Lank (lank@cs.uwaterloo.ca)

Participant Information Letter and Consent Form

Overview
You are being asked to participate in a research study as part of a research project at the University of Waterloo. The intent of this study is to determine user accuracy in discovering handwriting recognition errors.

This study is being conducted by Ryan Stedman under the supervision of Professor Edward Lank.

Study Details
As a participant in this study, you will be asked to participate in a single session. In this session you will be given a set of 20 handwritten forms, already filled out. You will be asked to use a computer to verify the handwriting recognition results performed on these forms, correcting any errors found.

The session should take approximately 90 minutes.

You may decline to answer particular questions, if you wish, and may withdraw participation at any time.

Risks and Benefits
There are no known or anticipated risks to you as a participant, other than those associated with the normal use of computers.

From this study we hope to discover a method of modeling how precise a user is in searching for errors in recognition, which could be used in future research and user interfaces to improve user accuracy when searching for errors.

Confidentiality and Data Retention
All data collected is considered confidential. Codes, rather than names or other identifying information, will be used in notes and/or recordings. Your name or any other personal identifying information will not appear in any publication resulting from this study. Data collected during this study, with identifying information removed, will be retained indefinitely in a secure location in the HCI research laboratory.

Remuneration
As a participant in this study, you will receive $15. In the event that you cannot

complete the requirements of the study, you will receive a prorated amount at the rate of $10/per hour.

Two participants with the highest performance on the verification task, based on a combination of speed in completing the task and accuracy in the task, will be given an additional $15. There will be approximately six other people participating in this study; the top two performers will receive notification once all participants have completed the study.

Questions
If you have any questions about participation in this study, please feel free to ask the researchers. If you have additional questions at a later date, please contact Ryan Stedman via email at rstedman@cs.uwaterloo.ca, or Professor Edward Lank at 519-888-4567 ext. 35768 or via email at lank@cs.uwaterloo.ca.

This project has been reviewed by, and received ethics clearance through, the Office of Research Ethics at the University of Waterloo. In the event you have any comments or concerns resulting from your participation in this study, please contact Dr. Susan Sykes at 519-888-4567, Ext. 36005.

A.5 Consent Form for Chapter 6 Study, Both Participant Groups

Consent Form
I agree to participate in a study being conducted by Ryan Stedman for a research project at the University of Waterloo. The faculty supervisor is Professor Edward Lank. I have made this decision based on the information I have read in this Information-Consent Letter and have had the opportunity to receive any additional details I wanted about the study. I understand that I may withdraw this consent at any time by telling the student investigator.

I was informed that I may withdraw my consent at any time without penalty by advising the students or project supervisor.

This project has been reviewed by, and received ethics clearance through, the Office of Research Ethics at the University of Waterloo. I was informed that if I have any comments or concerns resulting from my participation in this study, I may contact the Director, Office of Research Ethics at the University of Waterloo at (519) 888-4567 ext. 6005.

With full knowledge of all foregoing, I agree, of my own free will, to participate in this study.

___ YES ___ NO

____
Name of participant (Please print)

____
Signature of participant

____
Date

____
Name of witness (Please print)

____
Signature of witness

____
Date

A.6 Feedback Letter for Chapter 6 Study

Date

Dear Participant,

We would like to thank you for your participation in this study. As a reminder, the purpose of this study is to attempt to determine user accuracy of error discovery in handwriting recognition. While most of the errors you observed were real recognition errors, some were artificially introduced into the forms, allowing us to measure how accurate a user is in discovering error. The data collected through the experimental sessions will contribute to guiding future research and designs of handwriting recognition error discovery interfaces.

Please remember that any data pertaining to yourself as an individual participant will be kept confidential. Once all the data are collected and analyzed for this project, we plan on sharing this information with the research community through seminars, conferences, presentations, and journal articles. If you are interested in receiving more information regarding the results of this study, or if you have any questions or concerns, please contact us at either the phone number or email address listed at the bottom of the page.

If you would like a summary of the results, please let me know now by providing me with your email address. When the study is completed, I will send it to you. The study is expected to be completed by March 1st 2009.

As with all University of Waterloo projects involving human participants, this project was reviewed by, and received ethics clearance through, the Office of Research Ethics at the University of Waterloo. Should you have any comments or concerns resulting from your participation in this study, please contact Dr. Susan Sykes in the Office of Research Ethics at 519-888-4567, Ext., 36005.

Ryan Stedman
University of Waterloo
Faculty of Mathematics
Department of Computer Science
519-888-4567 ext. 38318
Email: rstedman@cs.uwaterloo.ca

References

[1] Anoto Group. *Anoto Digital Pen*, http://www.anoto.com/, Accessed February 2009.

[2] C. Baber and KS Hone. Modeling error recovery and repair in automatic speech recognition. *International Journal of Man-Machine Studies*, 39(3):495–515, 1993.

[3] M.L. Bourguet. Towards a taxonomy of error-handling strategies in recognition-based multi-modal human–computer interfaces. *Signal Processing*, 86(12):3625–3643, 2006.

[4] D. Goldberg and A. Goodisman. Stylus user interface for manipulating text. *Human-computer interaction: toward the year 2000 table of contents*, pages 500–508, 1995.

[5] S. Jaeger, S. Manke, J. Reichert, and A. Waibel. Online handwriting recognition: the NPen++ recognizer. *International Journal on Document Analysis and Recognition*, 3(3):169–180, 2001.

[6] Xiaodong Jiang, Jason I. Hong, Leila A. Takayama, and James A. Landay. Ubiquitous computing for firefighters: field studies and prototypes of large displays for incident command. In *CHI '04: Proceedings of the SIGCHI conference on Human factors in computing systems*, pages 679–686, New York, NY, USA, 2004. ACM.

[7] G. Kim, V. Govindaraju, and S.N. Srihari. An architecture for handwritten text recognition systems. *International Journal on Document Analysis and Recognition*, 2(1):37–44, 1999.

[8] Livescribe. *Livescribe Pulse Smartpen*, http://www.livescribe.com/smartpen/index.html, Accessed March 2009.

[9] I.S. MacKenzie and L. Chang. A performance comparison of two handwriting recognizers. *Interacting with Computers*, 11(3):283–297, 1999.

[10] Jock Mackinlay, Stuart K. Card, and George G. Robertson. A semantic analysis of the design space of input devices. *Hum.-Comput. Interact.*, 5(2):145–190, 1990.

[11] J. Mankoff, S.E. Hudson, and G.D. Abowd. Interaction techniques for ambiguity resolution in recognition-based interfaces. In *International Conference on Computer Graphics and Interactive Techniques*. ACM New York, NY, USA, 2007.

[12] J.C. Mankoff and G.D. Abowd. Error Correction Techniques for Handwriting, Speech, and Other Ambiguous or Error Prone Systems. 1999.

[13] Microsoft Corporation. *Windows Journal*, http://www.microsoft.com/windowsxp/tabletpc/default.mspx, Accessed February 2008.

[14] R. Plamondon, SN Srihari, E. Polytech, and Q. Montreal. Online and off-line handwriting recognition: a comprehensive survey. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(1):63–84, 2000.

[15] Satori Labs. *FusionForm Desktop*, http://www.satorilabs.com/products/ FusionFormDesktop.html, Accessed March 2009.

[16] L. Schomaker. User-interface aspects in recognizing connected-cursive handwriting. *Handwriting and Pen-Based Input, IEE Colloquium on*, page 8, 1994.

[17] L. Schomaker. From handwriting analysis to pen-computer applications. *Electronics & Communication Engineering Journal*, 10(3):93–102, 1998.

[18] Michael Shilman, Desney S. Tan, and Patrice Simard. Cuetip: a mixed-initiative interface for correcting handwriting errors. In *UIST '06: Proceedings of the 19th annual ACM symposium on User interface software and technology*, pages 323–332, New York, NY, USA, 2006. ACM.

[19] B. Suhm, B. Myers, and A. Waibel. Multimodal Error Correction for Speech User Interfaces. *ACM Transactions on Computer-Human Interaction*, 8(1):60–98, 2001.

[20] Keith Vertanen and Per Ola Kristensson. On the benefits of confidence visualization in speech recognition. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems*, pages 1497–1500, New York, NY, USA, 2008. ACM.

[21] A. Vinciarelli, S. Bengio, and H. Bunke. Offline Recognition of Unconstrainted Handwritten Texts Using HMMs and Statistical Language Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(6):709, 2004.

[22] Xugang Wang, Junfeng Li, Xiang Ao, Gang Wang, and Guozhong Dai. Multimodal error correction for continuous handwriting recognition in pen-based user interfaces. In *IUI '06: Proceedings of the 11th international conference on Intelligent user interfaces*, pages 324–326, New York, NY, USA, 2006. ACM.

[23] R. Yeh, C. Liao, S. Klemmer, F. Guimbretière, B. Lee, B. Kakaradov, J. Stamberger, and A. Paepcke. ButterflyNet: a mobile capture and access system for field biology research. In *Proceedings of the SIGCHI conference on Human Factors in computing systems*, pages 571–580. ACM New York, NY, USA, 2006.

[24] Ron B. Yeh, Andreas Paepcke, and Scott R. Klemmer. Iterative design and evaluation of an event architecture for pen-and-paper interfaces. In *UIST '08: Proceedings of the 21st annual ACM symposium on User interface software and technology*, pages 111–120, New York, NY, USA, 2008. ACM.

[25] M. Zimmermann and H. Bunke. Optimizing the Integration of a Statistical Language Model in HMM based Offline Handwritten Text Recognition. *money*, 2044(8):8, 1990.

[26] M. Zimmermann, J.C. Chappelier, and H. Bunke. Offline grammar-based recognition of handwritten sentences. *IEEE transactions on pattern analysis and machine intelligence*, 28(5):818–821, 2006.