TOWARDS THE APPLICATION OF A HANDWRITING INTERFACE FOR MATHEMATICS LEARNING

Lisa Anthony, Jie Yang, Kenneth R. Koedinger

Human-Computer Interaction Institute
Carnegie Mellon University
5000 Forbes Avenue, Pittsburgh, PA 15217

ABSTRACT

We believe handwriting input may be able to provide significant advantages over typing, especially in the mathematics learning domain. The use of handwriting may result in decreased extraneous cognitive load on students, and it may provide better support for the two-dimensional spatial components of mathematics when compared to existing typing-based tools. Here we report progress towards the application of a handwriting interface for mathematics learning. We introduce a prototype system that allows students to use handwriting input to solve algebraic equations in an intelligent tutor. We discuss strategies to improve the existing handwriting system and apply it to math learning. Although the recognition accuracy of current handwriting engines may not be at a level suitable for use by students, we hypothesize that this may be realistically improved via advance training of the engine on a large corpus, as well as via techniques similar to co-training.

1. INTRODUCTION

Intelligent tutoring systems provide a unique opportunity to enhance the learning experience, whether in the classroom or at a distance, by providing an online environment in which students can work at their own pace and at a time convenient to them. Cognitive Tutors [6] are one type of intelligent tutoring system that focuses on *learn-by-doing*, that is, problem solving. They have been created for a variety of learning domains, including algebra, geometry, foreign languages, chemistry, computer programming and more. The best *human* tutors can achieve a two-standard deviation improvement versus standard classroom instruction alone, effectively turning *C* students into *A* students; Cognitive Tutors have been shown to raise student achievement *one* standard deviation over traditional classroom instruction [5]. This research aims to improve mathematics learning in Cognitive Tutors even further, narrowing that two-sigma gap via use of multimodal and multimedia interface technologies.

Our specific focus is on exploring the ways in which the use of handwriting-based interfaces may provide pedagogical advantages, especially for the mathematics domain. Based on preliminary evidence we have collected, and based on hypotheses grounded in learning theory, we expect that handwriting may offer advantages for learning for several reasons. First, in a preliminary study we conducted in which users copied given equations in various modalities, including handwriting and typing, handwriting was the faster, and favored, modality to typing [2]. If extended to a learning task, similar results would allow students to accomplish more problems in the same amount of time, and to become more engaged during tutoring. Second, the use of handwriting rather than a menu-based typing interface may result in a reduction of extraneous cognitive load on students during learning. Extraneous cognitive load (*c.f.*, [13]) can be thought of as a measure of how much mental overhead students experience as a result of interface-related tasks while they are also trying to learn a mathematical concept. Third, in mathematics, the spatial relationships among symbols have inherent meaning, even more so than in other forms of writing. For instance, the spatial placement of the x in these two expressions changes the meaning of the expression significantly: "$2x$" vs. "2^x". Handwriting is a much more flexible and robust modality for representing and manipulating such spatial relationships. Mathematics tools that use a typing interface often require the user to become an expert at a new programming language (*e.g.*, Maple, Matlab), or to use complex menu-based templates (*e.g.*, Microsoft Equation Editor), both of which can be difficult and inaccessible to novices.

Based on our above hypotheses, handwriting has the potential to provide significant learning advantages over typing. However, for technological reasons, most intelligent tutoring systems rely on standard keyboard-and-mouse graphical user interfaces (GUIs). This is in part because handwriting recognition has been seen as still being in its early stages of development, too inaccurate for use with real users. We present a system design that can address this concern. This paper presents the design of a prototype of an intelligent tutoring system (ITS) that will allow users to solve algebraic equations via handwriting input. The ITS foundation of our prototype is Cognitive Tutor Algebra, and the handwriting recognizer used is FFES/DRACULAE [12]. We introduce the system architecture and some means of improving the handwriting recognition accuracy such that it is usable by students. We will reduce, if not eliminate, the need for repair by collecting a large corpus of student handwriting samples in order to train the engine in advance, as well as by using techniques similar to co-training described by Blum & Mitchell [3]. We also explore the possibility that students may not require the system to immediately output a recognition hypothesis. We may be able to design an instructional paradigm based on worked examples that provides a sort of *feed-forward* rather than *feedback* to students. We expect that the system will provide a natural and easy means for students to input their problem-solving process during tutoring, thus yielding improved learning gains.
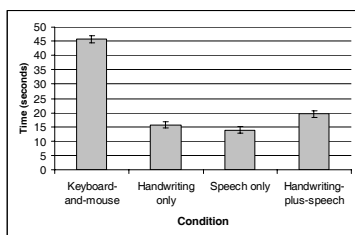
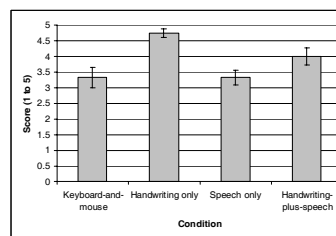**Figure 1.** Average time in seconds per equation (from [2]).

Figure 2. User ratings of each condition for entering math on the computer (from [2]).

2. Related Work

Handwriting-based mathematics interfaces are becoming a very popular research topic. Handwriting recognition research for the general domain has been an active area since the 1960s (for instance, see [1]). Computer recognition of mathematics began with the recognition of a printed page of mathematics. Now, with the prevalence of TabletPCs and PDAs which offer handwriting input as a main mode of input, real-time online human handwriting recognition is becoming more important to everyday users.

Several research and commercial systems exist that allow users to input and/or edit mathematical expressions. MathPad [7] is among the most robust and complex. In MathPad, users can write out mathematics equations and have the system animate the physical relationships given by these equations (for example, to animate a pendulum or oscillating sine curve). Other systems such as xThink's MathJournal [15] allow the sketching and writing of mathematics, but still rely on in-context menus to allow users to perform manipulations. In addition, even traditional keyboard-based math software such as Microsoft's Equation Editor and Maple 10 are now offering handwriting-based input, although limited in the amount of the equation that can be written or in what can be done as far as manipulation of the equation once it is input. Finally, inftyEditor [8] and Natural Log [10] are simple equation entry/editing programs without the added benefit of sketching or graphing on-the-fly. FFES [12], which we use, is similar to these but we chose it over the others because has a much higher base accuracy rate and because it is an open-source system, which is very important for system integration.

3. PRELIMINARY EVIDENCE IN SUPPORT OF HANDWRITING

As mentioned, we have already conducted a preliminary study that established handwriting as providing advantages over typed interfaces, at least in the area of *usability*. In that study, reported in [2], we hypothesized that handwriting would be a faster and more efficient means of input than typing, especially for mathematics input. The literature had established that the opposite was true in the domain of transcribing paragraphs of English [4], and researchers in handwriting input had accepted this for the math domain as well. We believed that there were fundamental differences in the two domains that would alter the result, however. College-aged users came to the lab in order to enter given equations in several modalities: typing in Microsoft Equation Editor (MSEE), handwriting, speaking, and handwriting+speaking.

Detailed results from this study can be found in [2]. To summarize them, we found that handwriting was indeed faster, by a factor of 3, than typing for entering mathematical equations on the computer, as well as being rated more highly in user preferences after the session. The speed effect magnified as the complexity of the equation increased: equations that included complex characters such as $\sqrt{}$ and $\sum$ were even slower in typing but the same interaction did not occur in handwriting. We have reproduced the graphs of the key findings here with permission. Figure 1 shows the average time in seconds per equation by condition. Figure 2 shows the post-session user ratings of each condition's "suitability" or "naturalness" for entering mathematics on the computer. In both cases, handwriting won over typing.

Based on the results of this study, we believed that handwriting may also have similar benefits when the domain is not just entering, but also learning, mathematics. The increased speed may have been in part due to the decreased cognitive load of the users, who were all novices to MSEE. The interaction between equation complexity and modality centered around the increased frequency of two-dimensional spatial relationships in the equations, which would become even more pertinent to students as they advance in mathematics topics from algebra to calculus. Handwriting provides a more direct means of manipulation and representation of such spatial relationships than linear text.

4. DESIGN OF HANDWRITING INTERFACE FOR MATHEMATICS LEARNING

We are in the process of developing a prototype system to allow students to solve mathematical equations via handwriting input. See Table 1 for examples of the types of equations students will solve in our system. To increase the possibility of success, we are building our system with a foundation of state-of-the-art intelligent tutoring system and handwriting recognition components, discussed in detail below. Once the prototype is complete we will conduct an evaluation of the system in a classroom setting, although the proposed system could also operate effectively as a distance learning tool. Figure 3 shows a diagram of the architecture of designed system. We employ a Java-based interface to integrate

$\dfrac{x+4}{3}=8$	$x+3=5$
$8x + 4 = 5x + 10$	$\dfrac{6x+8x}{14}=2$

Table 1. Sample equations in the algebra equation-solving domain we are addressing.
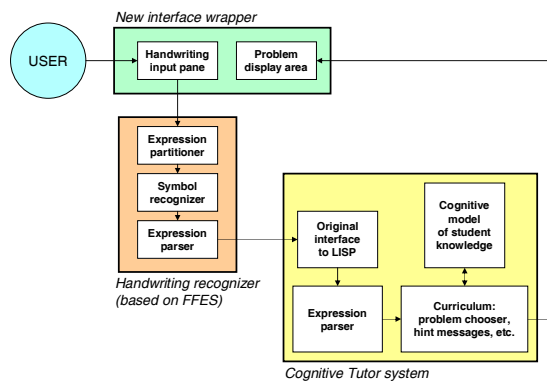
Figure 3. Architecture of prototype handwriting interface system.

FFES and the Cognitive Tutor.

The goal of this research is to use multimodal and multimedia technologies to improve mathematics learning for high school students. Besides handwriting, we consider using speech as an additional modality for error repair. Systems actually perform better when the modality of repair is different than the modality of entry, in part because users tend to over-enunciate (in speech) or trace heavily (in writing) and these patterns do not match the system's model (*c.f.*, [11]). Speech seems logical because it can be highly accurate when the symbol vocabulary is small (as in simple algebraic expressions), and because it does not require a return to the keyboard. We also consider using multimedia output. Our system may present to the student animated diagrams helping to explain the problem-solving process when the student needs a hint. Besides visual output, synthesized voice will be used when needed. Mayer's work exploring the principles of multimedia learning [9] will serve as design guidelines for when to include multimedia output and what this output should contain.

4.1. Technology Components

Cognitive Tutors have been an active area of research at Carnegie Mellon University since the 1980s [6]. They have been created for a variety of domains, including LISP programming, algebra, geometry, foreign language learning, and genetics. Cognitive Tutors for mathematics are in use in over 2000 schools in the United States. In the algebra tutoring lessons, students represent the situation algebraically in a spreadsheet-like worksheet and solve equations in a separate space with a symbol manipulation tool involving typing and menu-based operator selection. Each Cognitive Tutor is constructed around a cognitive model of the knowledge students are acquiring, and can provide step-by-step accuracy feedback and help as students solve problems.

Cognitive Tutors are implemented in LISP with a Java interface. We intend to replace this Java interface with a handwriting input space that has a handwriting recognizer behind it. We plan to use an off-the-shelf recognizer called FFES/DRACULAE because it relieves us of the technical burden of developing a robust recognizer from scratch and blazes a trail for using handwriting engines in more real-world applications. FFES has reported character recognition accuracy rates of about 77%, for both expert and novice users who had not trained the system to their style of writing [12]. With training, FFES can yield accuracy
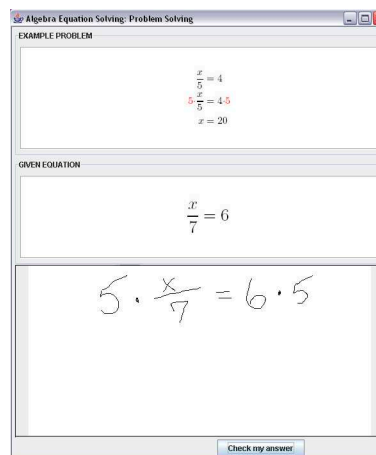

Figure 4. Screenshot of the interface of our prototype system. At the top is the worked example the student copied in the previous step; in the middle is the problem the student is solving.

rates as high as 95%. However, training handwriting recognition engines usually involves a large upfront time commitment during which the user inputs many (20+) examples of each character the recognizer is to understand (called *enrollment*). In a classroom setting, it is not very realistic to expect teachers to take valuable teaching time to allow students an hour or more to train the handwriting system. Therefore, it is imperative that we develop the system to maximize recognition accuracy while minimizing upfront training cost.

This forces us to consider a trade-off between the level of feedback we can provide to students and the accuracy of the recognition engine. To give perfect, step-targeted feedback at every step of the problem-solving process (as Cognitive Tutors currently do) may require more recognition accuracy than FFES can currently support. However, it is not clear that this level of feedback is required. We believe that there is promising evidence that the use of worked examples instruction, in which students copy and study example problems whose solutions are already worked out for them, may help mitigate the criticality of step-targeted feedback by providing a kind of *feed-forward*. Figure 4 shows a screenshot of the prototype system in which the student is solving the problem "*x/7 = 6*" by referring to the worked example at the top of the screen. The student's simple handwriting workspace is at the bottom of the screen. The use of worked example-based problem-solving has been successful in prior work on geometry [14].

4.2. Improving Handwriting Recognition Accuracy

Our long-term goal is to advance the theory of learning by fully exploring the feedback/accuracy trade-off. We may be able to improve accuracy in straightforward ways. The better the accuracy, the more feedback we can provide to students. In a step-by-step feedback paradigm, if the engine guessed wrong about the student's input, the feedback the tutor provides may be incorrect. Requiring students to correct the engine's guesses on the fly would impose *further* extraneous cognitive load, rather than reducing it.

Our main idea for how to improve accuracy is inspired by the *co-training* machine learning techniques developed by Blum and Mitchell [3]. In co-training, two independent labelers (for instance, one which reads the text on a website and one that reads the text of links pointing to a website) can each use the other's guesses to boost their own confidence, thus resulting in a classifier with greater overall accuracy than either one alone. We believe that similar techniques could be applied to our application, in which the handwriting engine is treated as one "labeler" and we combine it with various other "labelers" to achieve more accurate results. Potential labelers include the context of the problem we know the student is solving (which means we know what the student *should* be inputting), and also knowledge about common student errors as well as *this student's* current skills (which means we know what the student *might* be inputting instead). Co-training can be used as a learning technique, in which the labelers (here, the recognition engine) can improve long-term by incorporating knowledge from the other labelers, but it can also be used for *co-recognition*, as we described, improving accuracy on the fly and case-by-case.

Another important way we intend to improve recognition accuracy is through advance training of the recognizer on a large sample of corpus data from the target population. As mentioned, it is somewhat unrealistic to expect teachers to give up valuable instructional time in order for their students to spend even 10 minutes training the technology to their specific handwriting. In our preliminary studies, we have collected a corpus of over 15,000 individual online character samples from over 40 middle and high school algebra learners. We intend to batch-train the recognizer in advance on all of these samples in order to increase its *walk-up-and-use* accuracy: that is, when a new user tries the system, the accuracy should be higher than it would have been without the prior batch-train. The engine may also perform on-the-fly, student-targeted training during the copying of worked examples (effectively, labeled data), adapting the engine to the particular student as the lesson proceeds. With the use of these techniques, the resulting system should have much better performance accuracy than the raw, untrained engine on its own.

5. CONCLUSIONS

This paper has presented progress in developing a prototype system that allows students to use handwriting input to solve algebraic equations online in an intelligent tutor. We anticipate certain pedagogical advantages from the use of handwriting interfaces for learning mathematics, related to the decrease in extraneous cognitive load and the increased support for the spatial characteristics of mathematics notations, as well as evidence from prior studies that handwriting is faster and better liked than typing for this domain. While current handwriting engine accuracy may not be at a level suitable for use by students, we hypothesize that this may be improved via advance training of the engine on a large corpus of middle and high school math writing samples and via the use of techniques similar to co-training. The resulting system will be evaluated for its effectiveness in amplifying learning gains in students solving algebra equations, although the results of this work apply to other domains, such as geometry and physics.

6. ACKNOWLEDGMENTS

7. REFERENCES

[1] Anderson, R.H.: Syntax-directed recognition of hand-printed two-dimensional mathematics. In M.Klerer and J.Reinfelds, eds., *Interactive Systems for Experimental Applied Mathematics*. (1968) Academic Press, NY.

[2] Anthony, Lisa; Yang, Jie; Koedinger, Kenneth R.: Evaluation of Multimodal Input for Entering Mathematical Equations on the Computer. *ACM Conference on Human Factors in Computing Systems* (2005) 1184-1187.

[3] Blum, A. and Mitchell, T.: Combining Labeled and Unlabeled Data with Co-Training. *Proceedings of the Workshop on Computational Learning Theory* (1998) 92-100.

[4] Brown, C.M.L.: Comparison of Typing and Handwriting in "Two-Finger Typists." *Proceedings of the Human Factors Society* (1988) 381–385.

[5] Corbett, A.T.: Cognitive Computer Tutors: Solving the Two-Sigma Problem. *Proceedings of User Modeling* (2001) 137–147.

[6] Corbett, A.T., Koedinger, K.R., Hadley, W.H.: Cognitive Tutors: From the Research Classroom to All Classrooms. In: P. Goodman (ed.): *Technology Enhanced Learning: Opportunities for Change*. L. Erlbaum, Mahwah New Jersey (2001) 235–263.

[7] LaViola, J.J.: *Mathematical Sketching: A New Approach to Creating and Exploring Dynamic Illustrations*. Doctoral dissertation, Brown University (2005).

[8] Kanahori, T., Tabata, K., Cong, W., Tamari, F., and Suzuki, M.: On-Line Recognition of Mathematical Expressions Using Automatic Rewriting Method. *Proceedings of IEEE International Conference on Multimodal Interfaces* (2000) 394-401.

[9] Mayer, R.E.: *Multimedia Learning*. (2001) New York: Cambridge University Press.

[10] Matsakis, N.E.: *Recognition of Handwritten Mathematical Expressions*. Master's theses, Massachusetts Institute of Technology (1999) Cambridge, MA.

[11] Oviatt, S.: Taming Recognition Errors with a Multimodal Interface. *Communications of the ACM* 43 (2000) 45-51.

[12] Smithies, S., Novins, K., and Arvo, J.: Equation Entry and Editing via Handwriting and Gesture Recognition. *Behaviour and Information Technology* 20 (2001) 53-67.

[13] Sweller, J.: Cognitive Load During Problem Solving: Effects on Learning. *Cognitive Science* 12 (1988) 257-285.

[14] Trafton, J.G. and Reiser, B.J.: The contributions of studying examples and solving problems to skill acquisition. In *Proceedings of the Cognitive Science Society* (1993) 1017-1022.

[15] xThink.: *MathJournal*. (2003).