

**Entering Mathematical Equations  
Multimodally: Results on Usability and  
Interaction Patterns**

**Lisa Anthony    Jie Yang    Kenneth R. Koedinger**

March 15, 2006  
CMU-HCII-06-101

School of Computer Science  
Carnegie Mellon University  
Pittsburgh, PA 15213

**Keywords:** pen-based interfaces, multimodal interfaces, mathematics interfaces, equation entry and editing, usability, evaluation

## **Abstract**

Current interfaces for entering mathematical equations on computers are arguably limited and cumbersome. Mathematical notations have evolved to aid symbolic thinking and yet text-based interfaces relying on keyboard-and-mouse input do not take advantage of the natural two-dimensional aspects of mathematical equations. Due to its similarities to paper-based mathematics, pen-based handwriting input may be faster, more efficient, and more preferred for entering mathematics on computers. This paper presents an empirical usability study that tests this hypothesis. We also explored a multimodal input method combining handwriting and speech because we hypothesize it may aid both computer recognition and user cognition. Novice users were indeed faster in handwriting and enjoyed the handwriting modality more than a standard keyboard-and-mouse mathematics interface, especially as equation length and complexity increased. The multimodal handwriting-plus-speech method was faster and better liked than the keyboard-and-mouse method and was not much worse in these aspects than handwriting alone. In our domain, users' self-correction of errors tends to be easier in handwriting than with the keyboard; however, the overall average error rate was slightly higher in these modalities. In addition, users' speech while writing differed from when speaking alone. Finally, user errors in handwriting and speech were non-overlapping, a fact which a multimodal recognition system could use to improve performance.



# 1 Introduction

Current interfaces for entering equations are largely limited to keyboard- and mouse-driven windows-icons-menus-pointing (WIMP) interfaces. Advanced input consists of mark-up languages, such as LaTeX, and programming languages, such as Mathematica and Maple. Both methods require learning a new language and syntax, and can be difficult for novices to grasp and slow for experts to use.

Paper-based mathematical notations have evolved to represent and aid mathematical thinking and visualization. It is therefore natural and convenient for users to communicate with computers in the same way [4]. Because pen-based input can use traditional paper-based notations, it may be better with respect to speed, efficiency and user satisfaction for entering mathematics on computers. Several systems exist for handwriting-based input both online and offline (*e.g.*, [17]), but are not widely available to most novices. In addition, no rigorous studies have been done to demonstrate the effectiveness of such interfaces over more available keyboard-and-mouse interactions. This paper presents an empirical study to do so that motivates further investigation of multimodal interfaces for mathematics.

An important note is that this study focuses on the input experience decoupled from the problems inherent in automatic recognition technologies. Users were not given feedback about how the computer may or may not recognize their input, but rather were encouraged to assume recognition would be near-perfect. In this way, we expected to see the effects of various input modalities from the user perspective rather than from a technological perspective.

In data analysis, we evaluate user efficiency and user errors in different modalities and their combinations. We also report user preferences regarding various modalities both before and after experience. Our analyses indicate that novice users are faster, more efficient and prefer the handwriting modality more than a standard keyboard-and-mouse mathematics interface, especially as equation length and complexity increase. Furthermore, the multimodal handwriting-plus-speech method was faster and better liked than the keyboard-and-mouse method and was not much worse in these aspects than handwriting alone. In our domain, users' self-correction of errors tends to be easier in handwriting and speech than with the keyboard; however, the overall average error rate was slightly higher in these modalities. In addition, users' speech while writing differed from when speaking alone. Moreover, user errors in handwriting and speech were non-overlapping, a fact which a multimodal recognition system could capitalize upon to improve overall performance. Further stages of research will investigate recognition engine errors in this domain.

Although the multimodal interface results reported here will have applications to benefit general mathematical software package interfaces, another important application, and our area of interest, is in intelligent tutoring systems for math. Students using these tutors are novice mathematics learners and therefore are not likely to have used more complex tools for entering mathematical equations on computers. The minimization of excess cognitive load during the learning process is an important pedagogical principle for instructional design (*c.f.*, [18], [2]). If students must learn a specific system interface while also learning mathematical concepts, they may experience undue cognitive load and their targeted learning may suffer. We hypothesize that removing as much interface "overhead" as possible by increasing resemblance to paper will improve learning.

The remainder of this paper is organized as follows. We first present some relevant prior work

in the areas of multimodal interfaces, interfaces for mathematics, and pen-based input for math. We then detail our experimental design and present the results and discussion of our experiment. Finally, we discuss our current and future work in this area, as well as a proposed application for this line of research, and present our conclusions.

## 2 Related Work

### 2.1 Modality Advantage

Given prior studies where typing was found to be faster than handwriting (*e.g.*, [6]), one might ask why handwriting would ever be used instead of typing. In point of fact, studies favoring typing over handwriting with respect to speed may not apply to equation entry because they have focused on entering paragraphs of English text. Standard keyboards do not allow users to easily type complex mathematical expressions such as fractions, exponents or special symbols like  $\sum$  and  $\sqrt{\quad}$ . It is possible that for simple linear equations, the keyboard may be faster. However, for longer, more complex equations, we hypothesize that handwriting would be faster and more efficient than typing.

Although some systems that can recognize handwritten equations have reported evaluations (*e.g.*, [17], [13], [10]), they have all looked at user interactions in contexts which currently have recognition errors. None of them have reported an evaluation of the handwriting modality decoupled from technological limitations with respect to recognition accuracy and correction of errors. We believe usability and user preference concerns are critical to the success of a system. However, evaluating usability of a modality in the company of a system's recognition errors will not give a pure measure of the usability of that modality for the chosen domain, in this case, algebraic equations, because recognition accuracy and correction mechanisms vary across systems. Therefore, in the study reported in this paper, the users did not perform tasks in the context of any system, but instead input the given equations in an interface for storage rather than interpretation.

In addition, while efforts have been made to develop a grammar for users to express mathematics verbally to computer systems [8], these efforts are not based on empirical user or human factors evaluations. While we recognize the fact that technological achievements to date create limitations on the types of spoken or written input a machine can handle properly, we believe that ignoring the human question is a mistake. Human factors concerns should be the first motivation for technical development.

In fact, the reasons to use a multimodal interface over a unimodal one are on both the user side and the technology side. Almost effortlessly, humans are capable of fusing multiple channels of information and shifting attention amongst them, thereby coordinating both the content and process of human communication [7]. Oviatt et al found that users actually tend to generate more multimodal output under circumstances of increased cognitive load [15]. In addition, among other technological motivations, Oviatt [14] reported that users tend to select the input mode that they have found to be less error-prone within a system. This error avoidance helps the system perform better overall and be more preferred by users. Users also tend to simplify their language when interacting multimodally, which again helps recognition [14]. Designers of systems that use

multimodal recognition can benefit from understanding the ways in which humans generate and interpret multimodal input.

## 2.2 Computer Mathematics Interfaces

The tools available for entering and manipulating mathematics on the computer are sharply divided between those intended for expert mathematics users and those intended for novices. For our purposes, novice users are defined as those who are still learning mathematical concepts and notations, or who need to use or include mathematics within text documents, but who are not professional mathematicians. We focus here on entering algebraic equations, but similar limitations exist for other, more complex mathematical operations.

One difficulty of using mathematics on the computer is that handwritten or typeset mathematics often appears in higher-dimensional layouts, enabling, for example, the representation of both superscripts and subscripts. However, most computer interfaces are optimized for entering linear text [17]. Input techniques often used by expert mathematicians include mark-up languages, such as LaTeX and MathML, and special-purpose programming languages, such as Mathematica, Maple, or Matlab. Both methods require learning a new language and syntax. Learning math and its notations has been likened to learning a second language itself. Therefore, using a mathematics program that requires learning yet more unfamiliar syntactic structures and entry/edit methods only complicates the learning process. These programs have a large learning curve, even for mathematics experts, and therefore can be not only difficult for novices to grasp but also slow for experts to use.

Mathematics interfaces that do *not* require users to linearize their input also exist. Current interfaces for novices entering equations into other documents are largely limited to keyboard- and mouse-driven WIMP interfaces, for instance, the most commonly accessible: Microsoft's Equation Editor. These are called template-based editors, in which users select pre-defined mathematical structure templates (*e.g.*, fractions, superscripts, subscripts) from a menu or toolbar and then fill the templates in with numbers and operators by typing on the keyboard. These editors allow the user to construct a representation similar to the natural higher-dimensionality of mathematics, but require the user to construct such expressions in a top-down manner, making later structural changes difficult [17].

## 2.3 Pen-Based Input for Math

In contrast, paper-based mathematical notations have evolved to represent and aid mathematical thinking and visualization. It is therefore natural and convenient for users to communicate with computers in the same way [4]. Because pen-based input can use traditional paper-based notations, it may be more natural and suited for entering mathematics on computers. Several systems exist for handwriting-based input of mathematics both online and offline (*e.g.*, FFES [17], Infty Editor [10], Natural Log [12]), but are not widely available to most novices, as they require a tablet computer and are research-level systems. MathJournal [19] is a commercial product for the Tablet PC that was recently released; it allows pen-based entry of mathematics, but falls back on menu-based interactions for manipulating and editing these mathematics.

## 3 Experimental Design

We present here a user study we conducted in order to capture the usability of various modalities for entering mathematical equations on the computer. As discussed above, the study was not performed in the context of any one system, which would have technological limitations and interface idiosyncrasies. It was instead decoupled from such concerns in an effort to capture a “pure usability” measure for the given domain: algebraic equation entry.

### 3.1 Description

In our study, users were asked to enter mathematical equations of varying complexity using four different modalities: (1) traditional keyboard-and-mouse (**KB**) using Microsoft Equation Editor, (2) pen-based handwriting entry (**HW**), (3) speech entry (**SP**), and (4) handwriting-plus-speech (**HWSP**). Microsoft Equation Editor was chosen as a representative tool for novice users because it is in wide use and is a prime example of a WIMP interface for entering mathematics. During the study, users input equations in each of the four modalities. Users were not given feedback about computer recognition or interpretation of their input. Although this is clearly an important factor in determining the usability of specific *systems* that try to support this domain, the focus of our study was the usability of the various *modalities*.

Pairing handwriting and speaking may not immediately seem like a natural choice. We explored a multimodal input method combining handwriting and speech because we hypothesize such a combination of inputs might enhance computer-based recognition of equations [14] and could aid user cognition. Research has shown that people speak in an “inner voice” (subvocalization) while reading or writing [11]. We saw several users during the sessions who, in the speaking-only condition, wrote in the air with their hands while speaking the equation out loud. Exploring the pairing of these two modalities is important to supporting user cognition during handwriting input on the computer.

### 3.2 Participants

Forty-eight paid participants (27 male, 21 female), both graduate and undergraduate students at our university, answered an ad to participate in this study. All participants were fluent English speakers with unaccented speech. Participants ranged in age from 18 to 55, with a mean age of 22. The participants were primarily (24) Caucasian; 16 were Asian or Asian American and 8 were other ethnicities. We saw no effects of age or ethnicity in our analyses. Most (33) had no experience with Microsoft Equation Editor before the study. Of those who knew of it or had used it, only 2 classified themselves as knowing it “very well.”

### 3.3 Procedure

The experiment was a within-subjects design wherein participants came to the lab for a 45-minute session and entered mathematical equations on a Tablet PC in four different conditions. There was a list of 36 equations (9 per condition) whose order remained constant for all participants; order



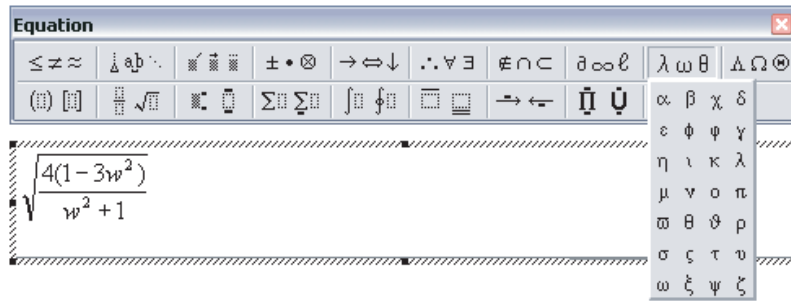


Figure 1: Microsoft’s Equation Editor toolbar provides menus to allow users to enter special symbols, fractions, exponents, etc.

of presenting each condition was counterbalanced across all possible orderings. Participants answered a questionnaire before the session in which they rated their impressions of the “suitability” or “naturalness” of each modality for entry of mathematical equations, whether or not they had used the modalities before.

Before performing each condition, participants were instructed as to how to enter equations in that condition. For instance, in the HW condition, the experimenter explained that the stylus could be used like a regular pen on paper. The experimenter did not tell the participants in what format to write the math, or how to find certain symbols in MSEE or express things verbally. Participants were given a 5-minute “practice” period before the KB condition to familiarize themselves with the Microsoft Equation Editor toolbar (Figure 1). During this time, they explored the interface on their own with no feedback or input from the experimenter. Although there was no exploratory period for the other three conditions, the first two equations in each condition were considered practice and were not included in our analyses. When participants finished all four conditions, they answered a questionnaire again rating their preferences for entering equations in each condition.

### 3.4 Stimuli Design

The experimental stimuli (36 equations) were designed with two components in mind: (1) the number of characters in the equation, and (2) the number of “complex” symbols appearing in the equation such as fractions, exponents, special symbols, and so on. The equations were all presented in standard typeset form as might appear in a mathematical textbook. This presentation format was chosen for conformity’s sake and because it can be considered a sort of “de facto” standard. Linearized versions of equations, such as “ $(1 - x)/(x^2) = y$ ”, are nonstandard, and can be difficult for humans to interpret [17]. Figure 2 shows three sample equations from left to right in increasing complexity. The first equation has 10 characters and has no special symbols that do not appear on the keyboard. The second equation has 17 characters and no non-keyboard symbols. The third equation has 14 characters, two of which are special symbols. We expected

$f(x) = x^4 - 31$	$\frac{2x^2}{x+1} <  y  < \frac{x+5}{2}$	$\int \left( \frac{u^7}{2} - \frac{\pi}{u^5} \right) du$
SIMPLE EQUATION	LENGTHY EQUATION	COMPLEX EQUATION

Figure 2: Experimental stimuli as users saw them.

that both design components would have an effect on user performance. Increased length should increase time because we hypothesized that additional characters in any modality would require more time to enter. Adding symbols that do not appear on the keyboard, such as  $\sum$  and  $\sqrt{\quad}$ , should only have a significant effect in the KB condition, because we hypothesized that special symbols are no more difficult than normal symbols when speaking or writing, but Microsoft’s Equation Editor requires finding them in menus (as shown in Figure 1). The length of each equation ranged from 10 to 18 characters, and 0, 1 or 2 special symbols.

### 3.5 Measures

The data from each session were collected at 30 frames per second by capturing the screen output and audio on a DV recorder. We measured time for each participant to enter each equation in each modality. We also analyzed the final transcripts of the output in each condition for errors. Errors were user actions in the following two categories: (a) mistakes—omitted, superfluous, or incorrect characters; and (b) self-corrections—scratched out, reworded, respoken or rewritten characters. Less than 5% of all equations were lost due to technical difficulties. Finally, we collected user preference ratings from both before and after the session by asking users to respond on a Likert-scale questionnaire as to their impression of the “naturalness” or “suitability” of each condition for entering math on the computer.

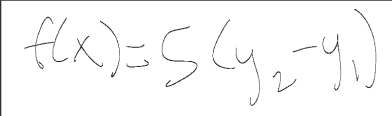
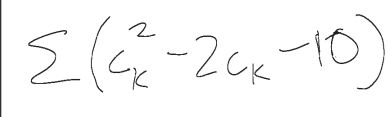
## 4 Results and Discussion

### 4.1 Qualitative Examples

Here we present examples of an equation from each condition and one particular user’s response to that equation.

**Keyboard-and-mouse (KB).** Row 1 in Table 1 shows an example of a user’s input for an equation with 14 characters. The left-hand side shows the correct typeset version; the righthand side shows the user’s input. Note that it contains an error: the user typed a “1” for the “x” in the denominator. The “/” characters were not counted as an error but instead were considered a notational difference. In this equation, which took the longest to complete out of all equations (209 seconds), the user painstakingly reconstructed the typeset version using the templates provided in MSEE. Note that this is not the longest equation users ever saw. A minority of users sometimes

Table 1: A set of selected samples, from different users, from each modality in the study illustrating typical responses.

Modality	Given Equation	Sample	Notes
KB	$\frac{1}{ x +1} - \frac{x^2}{2} \leq y$	$\frac{1}{ 1 +1} - \frac{x^2}{2} \leq y$	At 209 seconds (in the keyboard-and-mouse condition), this equation took the longest to complete.
HW	$f(x) = 5(y_2 - y_1)$		A sample of a user's input for an equation in the handwriting only condition.
SP	$\frac{y-4}{y^2-5y+4} = 9$	“y minus four over y squared minus five y plus four equals nine”	A transcript of a user's input for an equation in the speech only condition.
HWSP	$\sum [c_k^2 - 2c_k - 10]$	 “sum of c subscript k squared minus two c subscript k minus 10 close parentheses”	A sample of a user's input for an equation in the handwriting-plus-speaking condition, both the handwriting component and the transcription of the speech component.

linearized equations by typing fractions on one line separated by “/” or by using the “^” notation for exponents, rather than using the provided templates.

**Handwriting only (HW).** Row 2 in Table 1 shows an example of a user's input for an equation with 13 characters. Note that the handwritten version does not appreciably differ in structure from the typeset version. In the HW condition, users tended to write the equation exactly as it appeared in the typeset version without linearizing or altering it. This is tied to the fact that pen-based input provides better affordances for higher-dimensional representation used in traditional mathematics notations.

**Speech only (SP).** Row 3 in Table 1 shows an example of a user's input for an equation with 13 characters. Note that the user simply reads the symbols without attention to mathematical precision and notions of “quantity.” In the speech modality, users tended to express equations without inserting ambiguity control techniques, especially if they were not physically present in the typeset version. Users' spoken equations tended to be highly variable and disfluent, likely due to the lack of visual feedback in this condition, but potentially normal for spoken input. We briefly discuss quantitative results on this subject later in this paper.

**Handwriting-plus-speech condition (HWSP).** Row 4 in Table 1 shows an example of a user's input for one equation with 13 characters. Note the semantically-equivalent substitution of “( )” for “[ ]”, which was not counted as an error. In this example, unlike in the SP example, the user

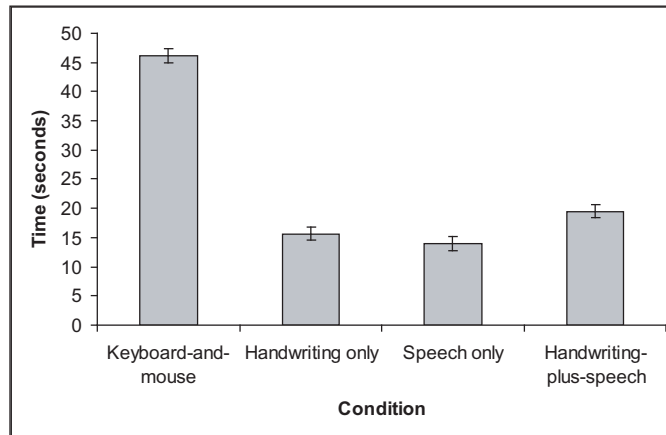


Figure 3: Average time in seconds per equation by condition. Error bars show 95% confidence interval (CI).

*did* insert a notation for “quantity” in his speech by adding the phrase “close parentheses” at the end of his speech. While not a common practice, more than one user tended to insert ambiguity control phrases at the *end* of the mathematical quantity, rather than the beginning or at both the beginning and the end. We briefly discuss quantitative results on the amount of ambiguity control users exhibited in both the SP and HWSP conditions later in this paper.

## 4.2 Speed of Input

We ran a univariate ANOVA on time per equation considering the following factors: (1) participant as a random factor to account for the correlations between datapoints<sup>1</sup>, (2) input condition and appearance of non-keyboard characters as fixed factors, and (3) the number of characters in each equation as a continuous covariate. This analysis yielded a significant interaction between condition and appearance of non-keyboard characters ( $F_{3,1090} = 20.21, p < 0.0005$ ). The KB condition was slower than the other conditions when non-keyboard characters appeared than when they did not, which we expected. There was also a significant main effect of the number of characters in the equation ( $F_{1,1090} = 64.90, p < 0.0005$ ). Longer equations took more time to enter. A planned contrast comparing the KB condition to the other three conditions showed a significant difference ( $t(1090) = 40.98, p < 0.0005$ ), KB being slowest. Figure 3 shows the overall mean time in seconds for each condition.

The HWSP condition was slower than the SP and the HW condition. This may have been due to the fact that some participants performed the HWSP condition in parallel, while some did it in series (first one modality and then the other). Thus, HWSP may not actually differ in speed from the HW or SP conditions. Therefore, we ran another univariate ANOVA on a subset

<sup>1</sup>Treating participant as a random factor in this way is equivalent to a repeated-measures analysis.

Table 2: Distribution of errors per equation by condition.

Condition	% of Equations with No Errors	% of Equations with 1 Error	% of Equations with > 1 Errors
KB	94%	3%	2% (4)
HW	80%	17%	3% (3)
SP	82%	14%	4% (4)
HWSP	70%	20%	10% (4)

of the data containing only the HWSP condition, with the same model as above, except we did not include participant because it was no longer a within-subjects contrast. This analysis yielded no interactions, but significant main effects of parallel vs. serial performance ( $F_{1,321} = 244.52$ ,  $p < 0.0005$ ), of equation length ( $F_{1,321} = 68.86$ ,  $p < 0.0005$ ), and of appearance of non-keyboard characters ( $F_{1,321} = 41.23$ ,  $p < 0.0005$ ). The mean difference between HWSP in parallel (*mean*: 17.12, *sd*: 5.62) and in series (*mean*: 27.71, *sd*: 5.30) was significant ( $t(321) = -25.33$ ,  $p < 0.0005$ ). The HWSP in parallel users were about as fast as HW users, whereas the HWSP in series were slower.

Because non-keyboard characters showed a significant effect on speed in non-keyboard modalities (here, HWSP), it may be more accurate to refer to them as “complex mathematical symbols.” Characters such as  $\sum$  and  $\sqrt{\quad}$  only appeared in the later equations users saw in each condition. They therefore may have had lower cognitive activation when they did appear, leading to a decrease in performance speed for all conditions rather than just KB.

## 4.3 User Errors

### 4.3.1 Error Rates

Table 2 gives a summary of the distribution of errors per equation across the four conditions. We defined “errors” as user actions in the following two categories: (a) mistakes—omitted, superfluous, or incorrect characters; and (b) self-corrections—scratched out, reworded, respoken or rewritten characters. User errors occurred on 19% of all equations. The KB condition was the least error-prone with errors on only 6% of equations performed in this condition, compared to 30% in the HWSP condition. The most errors a user made on any one equation was approximately consistent across conditions (3 to 4 errors).

In [3] a preliminary error analysis on the same dataset was reported that found KB to be the *most* error-prone condition. In that paper, a slightly different definition of “error” was used, which may have caused the number of errors in KB to appear inflated. Errors were coded in that paper by watching the videotape of each user’s session to find in-process self-corrections in addition to just the output of the session, as reported here. However, it is likely that the fact that the KB condition had the most errors per equation was an artifact of the input modality itself, based on this

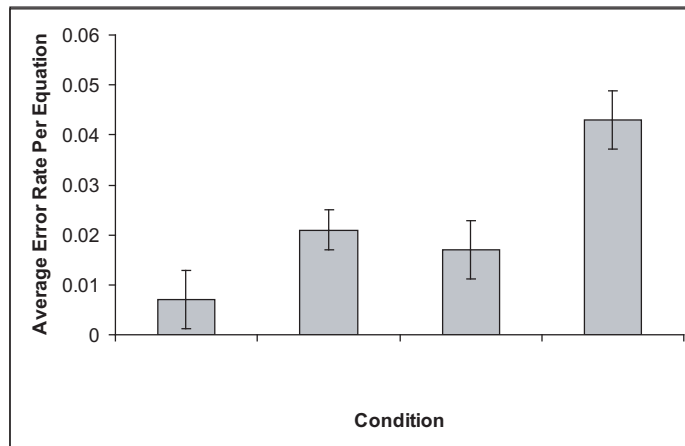


Figure 4: Mean number of errors made per equation by condition. Error bars show 95% CI.

error coding. Because the keys are close together on a keyboard, people often make typographical errors and must use the backspace key to correct them, which showed up in the video and was counted in the original error metric. Because these are not cognitive errors on the part of the user, we decided against counting them here. In addition, the error metric in [3] was not normalized for equation length.

Therefore, we revisited the problem of error analysis. It is important to note that the errors coded in each condition are still not completely symmetric. In the KB condition, we had no artifact remaining to show self-corrections, whereas in the other condition, these “errors” showed up as scratch-outs or verbal corrections. This lack of self-corrections is a consequence of our re-coding of errors. Because we did not want to artificially inflate errors reported for the KB condition, we sacrificed being able to count (rarer) cognitive errors in order to also not count (far more common) typographical ones.

We performed a univariate ANOVA on error rate per equation with the same factors as for speed. We found that length of the equation was not significant ( $F_{1,905} = 0.82, n.s.$ ) because the error rate normalizes for equation length, so we removed it from the model. We then found a significant interaction between condition and appearance of non-keyboard characters ( $F_{3,406} = 7.36, p < 0.0005$ ), in which HWSP had a greater increase in error rate than the other conditions when complex math symbols appeared in the equation than when they did not. This was a surprise to us, because we would have expected the KB condition to behave this way instead. This effect could be caused by the fact that the HWSP condition, already heavier cognitive load than the other conditions because users had to perform two tasks at once, suffered more under the additional increased cognitive load caused by the introduction of lower-activation, complex mathematical symbols. Figure 4 shows the overall mean error rate per equation by condition. The KB condition has the lowest error rate of the four conditions. It was also the slowest, which implies that there is a speed/accuracy trade-off in effect. We discuss this possible trade-off below.

The HWSP condition has the highest error rate of the four conditions, only slightly less than the number of errors in HW and in SP combined—this is not surprising because users have twice as many opportunities to make errors in this condition since they are doing the same equation twice. A more fair way to look at this metric is by modality. A univariate ANOVA on error rate per equation by modality (HW or SP; alone or together; KB not included) rather than condition showed no significant effect of either modality ( $F_{1,1246} = 0.23, n.s.$ ) or presence of a companion modality ( $F_{1,1246} = 0.06, n.s.$ ). Thus, it appears that the presence or absence of another modality does not hurt user performance. Multimodality, therefore, does not seem to make this task more difficult for users to perform; errors are simply a sum of the component modalities rather than there being a further complication as a result of having to do two modalities at once.

### 4.3.2 Speed/Accuracy Trade-off

Figures 3 and 4 seem to indicate the existence of a speed/accuracy trade-off. The KB condition takes the longest time, but also has the lowest overall error rate. Such trade-offs can be very important to the design of a system, because the specific needs of the end-user will determine which aspect of the trade-off to favor in the design. Because our measure for errors does not take time into account, the relationship between errors and speed is not clear from this analysis. However, we do not have a good estimate of how long it takes to correct an error in each condition, so adding time to the error rate is non-trivial. Instead, we consolidated the time measure to take into account equation length by analyzing a “data entry” metric, “characters per second.”

We ran a univariate ANOVA on characters per second with the following factors: (1) participant as a random factor to account for the correlations between datapoints, (2) input condition and appearance of non-keyboard characters as fixed factors, and (3) the number of characters in each equation as a continuous covariate. We found significant main effects of all three factors: input condition ( $F_{3,140} = 192.17, p < 0.0005$ ); appearance of non-keyboard characters ( $F_{1,1093} = 239.37, p < 0.0005$ ); and equation length ( $F_{1,1093} = 10.18, p < 0.001$ ). Figure 5 shows the means of characters per second in each condition. The SP condition is the fastest, HW a close second, and KB is significantly slower than the other three ( $t(1093) = -48.0, p < 0.0005$ ).

As mentioned, we do not know the exact amount of time it takes to correct an error in each modality. However, we do know that repair of user-generated errors should be easier in handwriting than in the keyboard modality. For instance, in handwriting, if users catch errors they make once they have finished writing out the equation, they only have to scratch something out or write over the erroneous character. In the keyboard condition, however, they must move the cursor back to the appropriate spot, retype in place, and may even have to alter the templates and move elements of the equation around to get it to display correctly. Therefore, we believe this speed/accuracy discrepancy does not favor KB but rather would favor HW.

### 4.3.3 Error Locations

We were additionally interested in determining whether or not errors in different modalities tend to overlap, that is, do the same errors appear in the same locations within equations in different modalities? If this is the case, then support for multimodality may be difficult. To the extent

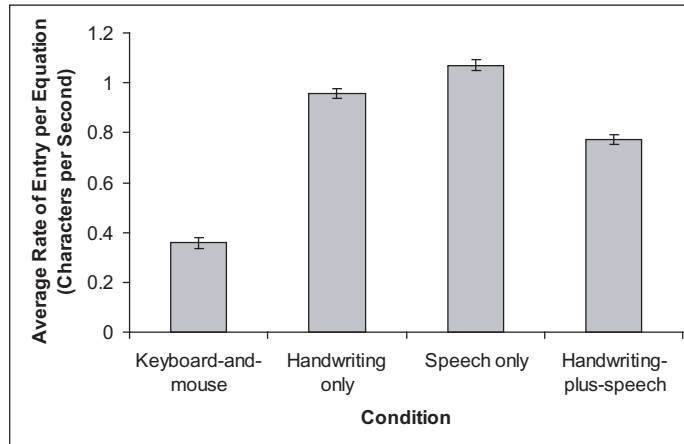


Figure 5: Average rate of entry (characters per second) per equation. Error bars show 95% CI.

Table 3: Correlation results for error locations within equations across modalities. HS:HW means the handwriting modality of the HWSP condition; similarly for HS:SP.

Condition	SP	HS:HW	HS:SP
HW	-	-	-
SP		-	n.s.
HS:HW			-

that errors are independent, two independent recognition engines may be able to improve overall system performance (*c.f.*, [5], [14]). To investigate this feature, we ran a bivariate correlation of the total number of errors in each position of each equation by modality.

Table 3 shows the positive (+) and negative (-) correlations of locations of errors in each modality (except KB). All correlations shown are significant at the  $p < 0.05$  level (*n.s.* means not significant). Correlation sizes were in the “weak” range of 0.20 to 0.30. The fact that the pen and speech modalities have significant negative correlations means that the errors were non-overlapping. When errors in one location and modality were high, they were low in the other modality. Interestingly, there was no correlation between SP and HS:SP but there *was* one between HW and HS:HW. This implies that, although the presence of another modality does not increase the error *rate* (as shown in the section on error rates), it does change the *types* of errors users make in the handwriting modality. Users make not more but *different* errors in handwriting when they are also speaking versus when they are just handwriting. To the extent that a multimodal interface supporting both handwriting and speech seems plausible, this is an important implication for multimodal interface design.



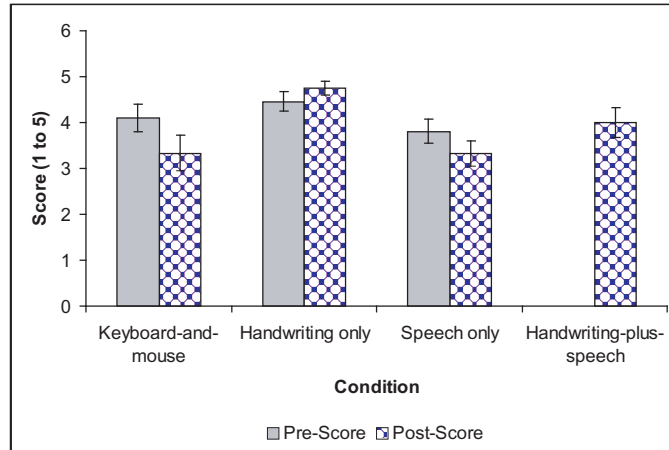


Figure 6: Pre- and post-test questionnaire rankings of each condition on a 5-point Likert scale. Users did not answer about the HWSP condition on the pre-test, because they had likely never experienced it before. Error bars show 95% CI.

#### 4.4 User Preferences

We also asked users both before and after the session to rate the “suitability” and “naturalness” of each modality in the session for entering mathematics on computers. Users rated each modality’s suitability on a 5-point Likert scale. On the pre-test questionnaire, there was no significant difference between the KB condition and the HW condition ( $t(47) = -1.61, n.s.$ ). However, on the post-test questionnaire, users rated the HW condition higher than the KB condition ( $t(47) = 4.49, p < 0.0005$ ). Figure 6 shows the mean pre-test and post-test questionnaire ratings of the four conditions.

Note that speech had the second-highest accuracy of the four conditions, as noted in Table 2 and Figure 4. In spite of this, participants rated speech quite low in user preferences. They said later in informal interviews that they did not like the lack of feedback in the SP condition—with no visual reminder of what they had said, they did not feel confident in that modality. This is a limitation of this study, in that providing feedback might have been more “realistic” with respect to how speech systems work today. In addition, teachers and students in classrooms rarely speak mathematics without a visual component. We address this limitation and discuss potential solutions later in this paper.

During the experiment, users were not restricted on how to perform the HWSP condition. The instructions stated that they could do both modalities at once, or do one modality and then the other, and that they should choose whatever felt most natural for them. Most users (77%) chose to use the two modalities in parallel for each equation and remained consistent throughout the study. This is in contrast to Oviatt’s work [16], which found that the majority of adults tend to be sequential during multimodal interactions. That study was in a speech and pen-gesture interface for map interactions; thus, the domain may be sufficiently different from ours that the behaviors will differ.

In the present study, of those who did *not* use the modes in parallel, they were split evenly between starting with handwriting and starting with speech. The way users performed the HWSP condition did not significantly affect their rating of that condition post-session ( $F_{2,45} = 0.34, n.s.$ ).

## 4.5 Speech Modality Effects

Preliminary data analysis showed potential differences in the ways users expressed themselves in speech between the two speech modality conditions, SP and HWSP. If these differences do exist, they will be important to future efforts toward making a language model for algebraic equations that a speech engine can use during recognition. Therefore, we explored three aspects of the speech modality: utterance length, pausing, and degree of ambiguity control.

### 4.5.1 Length of Utterances in Speech

We ran an analysis to determine if the length of utterance differed when the user was speaking the equation alone (SP) or while also writing (HWSP). A univariate ANOVA with participant as a random factor, condition and appearance of non-keyboard characters as fixed factors, and equation length as a continuous covariate revealed no main effect of condition ( $F_{1,587} = 2.81, n.s.$ ; SP *mean*: 18.28, *sd*: 4.03; HWSP *mean*: 17.77, *sd*: 4.00). However, we did find both a significant main effect of equation length ( $F_{1,587} = 273.8, p < 0.0005$ ) and appearance of non-keyboard characters ( $F_{1,587} = 35.72, p < 0.0005$ ). The non-keyboard characters that appeared were characters such as  $\sum$  and  $\sqrt{\quad}$ , which, as mentioned above, may have had less cognitive activation when the users encountered them, therefore prompting more phrases such as “uh,” “um” and so on, or which perhaps are inherently expressed in lengthier verbal descriptions. The average utterance length across all equations (not accounting for number of characters in the equation) was 17.6 words (including conversational phrases such as “uh” or self-corrections such as “oops”), with a range of 9 to 42 words per utterance.

### 4.5.2 Pauses in Speech

The number of pauses in spoken utterances between the SP and HWSP condition did differ significantly, however. This difference is likely due to the effect of synchronization of speech and writing in the HWSP condition. In Figure 3, we showed that HW was slower than SP; this means that the user must pause in speech while his or her hand “catches up” to his or her thought. Equation length may also affect the number of pauses. Natural speech consists of pauses for breath and punctuation and beat markings; people tend to pause after every few words. As above, we ran a univariate ANOVA and found that equation length does have a significant effect on the number of pauses per equation ( $F_{1,494} = 95.05, p < 0.0005$ ). In addition, both condition ( $F_{1,47} = 28.88, p < 0.0005$ ) and appearance of non-keyboard characters ( $F_{1,50} = 41.00, p < 0.0005$ ) had significant main effects. The effect of non-keyboard characters supports our earlier argument that these are more accurately thought of as “complex mathematical symbols” that have inherent complexity beyond not appearing on standard keyboards. The overall mean number of pauses per equation was 3.0,

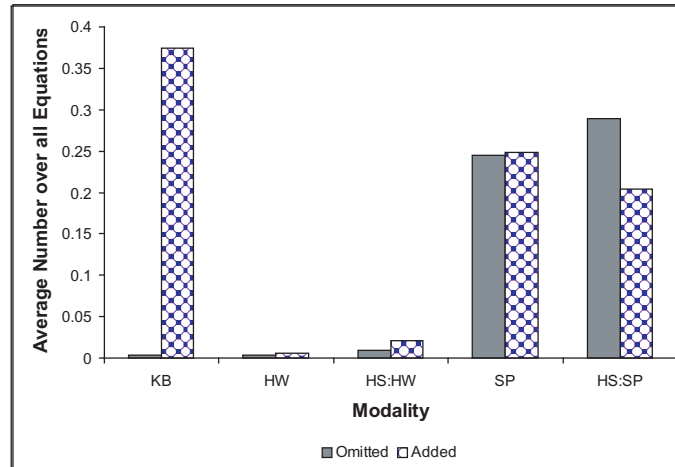


Figure 7: Average number of parentheses omitted and added per equation in the five modalities.

with a range of 0 pauses to 13 pauses per equation (SP *mean*: 2.76, *sd*: 1.49; HWSP *mean*: 3.6, *sd*: 1.46).

### 4.5.3 Ambiguity Control in Speech

During preliminary data analysis and as seen in the examples in the qualitative results, we noted that participants generally did not seem to be very precise in their speech with respect to ambiguity of expression. For instance, they often left out phrases such as “quantity of,” especially in complex equations where it might become difficult to keep all of the open quantities in short term memory. We therefore looked at the number of parentheses that actually existed in the typeset equation (*e.g.*, with physical parentheses), the number of these that were omitted, and the number that were additionally added, in the four conditions, as a measure of ambiguity control. Figure 7 shows the average number of omissions and additions per equation by condition. From this graph one can see that participants were most likely to omit parentheses in the speech modality, but were also more likely to add in their own, different parentheses in those conditions containing speech (HWSP and SP). In addition, they also tended to add more parentheses in the KB condition. This may be due to the fact that some of the participants linearized the typeset version of the equation while typing it in, thus requiring the addition of parentheses to be interpretable correct. Note that in the analyses of error rates and locations presented in this paper, ambiguous mathematics caused by the omission of parenthetical expressions were not counted as user errors. We believe that the natural expressions of users should be what drive the system’s recognition. For teaching purposes, tutoring systems can correct students if the teacher desires them to be mathematically precise in speech, but the system must be able to recognize the common patterns of speech.

## 5 Current and Future Work

### 5.1 Application: Intelligent Tutoring Systems

We envision that general mathematics software packages can benefit from the incorporation of interfaces using handwriting by improving the naturalness of the interaction. In addition, there is reason to believe that interfaces in intelligent tutoring systems for mathematics could benefit in important ways from the incorporation of multimodal interfaces, and we intend to focus our efforts in this area. A key pedagogical motivation for this is related to the instructional principle to minimize working memory or cognitive load (*e.g.*, [18], [2]). Because most current mathematics software rely on standard WIMP interfaces, learning how to use and focusing on this interface must take up part of the student's cognitive processes. Issues of cognitive load caused by such resource-consuming interfaces may interfere with learning of the goal concept. Simplifying the interface to rely solely on the "language of mathematics" that the students use in class and on paper may help alleviate these cognitive load issues and therefore could potentially improve learning.

### 5.2 Future Work

The data gathered in this study is a rich corpus which can yield several further benefits. In particular, the building of a language model for people's natural expression of mathematical equations may be of interest both to linguists and to designers of systems allowing users to speak mathematics (potentially in combination with other modalities). We have presented here data on the length of utterances, number of pauses, amount of synchronization, and degree of ambiguity control in speech when speaking alone or in combination with writing. Further questions include the amount of user variability in expression among modalities and the extent to which users' natural mathematical speech can be modeled for a recognizer. Having shown that a multimodal interface for mathematics can provide gains both from a usability perspective as well as a technological perspective, such a language model would be invaluable for the design of such systems.

A further area of future work would be to test how user preferences and performance change over long-term exposure to each modality. Users in this study only experienced each modality for an average of 2 to 6 minutes. Although we did allow for practice effects in the first few equations the users performed by not including them in the analysis, it is possible that users did not reach the plateau of the learning curve in the time they experienced each modality. Asymptotic performance data would be useful information, especially for the design of systems that will be used by students in a year-long curriculum or long-term users of a particular math software package. In addition, the learning curve of some modalities may differ from that for others and this would be further evidence in the case for or against the use of handwriting instead of keyboard-and-mouse in mathematics interfaces.

One limitation of this study is the lack of visual feedback in the speech-only condition. Users did not feel comfortable in this modality because they could not keep track of what they had said. We may be able to address this limitation in future studies that include a speech-only condition, perhaps by using a Wizard-of-Oz setup in which a human interprets what the user said on the fly and provides visual feedback. The human playing the Wizard, however, may not perform at 100%

recognition accuracy, which re-introduces the problems associated with testing modalities in the context of non-user-generated errors. In addition, the format of the visual feedback to provide is unclear. Is it better to display simply a written transcript of the words the users speaks, or should the system perform the additional step of parsing to display legal mathematical notation? Both approaches lead to such a study being performed in some context, rather than allowing a free and pure usability measure to be obtained. To the extent that a speech-based interface seems promising, this is an important avenue to explore.

Apart from the technological benefits of co-recognition mentioned in this paper, we hypothesize that multimodality may also have pedagogical benefits. The use of a robust multimodal interface that allows students to write out the math by hand naturally and that includes speech recognition for explanation of problem-solving steps may be a winning combination. Meta-cognitive activities such as self-explanation have already been shown to improve learning in the Cognitive Tutor (*e.g.*, [1]), and other studies have shown that spoken self-explanations may be more beneficial to learning than typed ones [9]. The study reported here encouraged shallow “shadowing” in the HWSP condition (the same content was presented in both modalities in the HWSP condition), rather than elaborated explanations. We drew this paradigm from real-world examples of professors teaching at a blackboard who both write and speak equations at the same time because our study was about equation *entry*. For students learning how to *solve* equations, however, this explicit redundancy does not seem to occur as often. In future studies we hope to support command-based spoken input, such as “subtract from both sides,” to accompany the handwriting of equations, in which the students will explain the mathematical operation they took from step to step. This needs further exploration before we can draw conclusions.

The next stages we will begin to address will include technological concerns such as recognition performance. We intend to test alternative automated recognition methods based on co-recognition or co-training [5]. In addition, we plan to evaluate the impact of the use of different interface modalities on learning for students solving problems on the computer.

## 6 Conclusion

We have performed an empirical study on inputting mathematical equations using different modalities. The results from this study have indicated that the keyboard-and-mouse condition was significantly slower than the other three conditions. In addition, users preferred the handwriting modality for entering equations to the standard keyboard-and-mouse interface they used in the study. In our domain, users’ self-correction of errors tends to be easier in handwriting and speech than with the keyboard; however, the overall average error rate was slightly higher in these modalities. The multimodal handwriting-plus-speech method was faster and better liked than the keyboard-and-mouse method and was not much worse than handwriting alone. In addition, users’ speech while writing differed from when speaking alone. Finally, user errors in handwriting and speech were non-overlapping, a fact which a multimodal recognition system could capitalize upon to improve performance. The results from this study can help inform future design of more natural and efficient interfaces for mathematics.

## 7 Acknowledgments

The authors would like to thank Sharon Oviatt, Chris Atkeson, Shelly Evenson, Carolyn P. Rosé, Darren Gergle, Jiazhi Ou, Jacob O. Wobbrock, Cristen Torrey, Aaron O. Bauer, Sonya Allin, Datong Chen, Adam M. Fass, Laura Dabbish, Ryan S. Baker, Angela Wagner, Anupriya Ankolekar, Scott Davidoff, Elsa Golden, Amy Hurst, and Ido Roll for their invaluable equipment, time, and support.

## References

- [1] V.A.W.M.M. Aleven and K.R. Koedinger. An effective metacognitive strategy: Learning by doing and explaining with a computer-based cognitive tutor. *Cognitive Science*, 26:147–149, 2002.
- [2] J.R. Anderson, A.T. Corbett, K.R. Koedinger, and R. Pelletier. Cognitive tutors: Lessons learned. *Journal of the Learning Sciences*, 4:167–207, 1995.
- [3] L. Anthony, J. Yang, and K.R. Koedinger. Evaluation of multimodal input for entering mathematical equations on the computer. In *Proceedings of the ACM Conference on Human Factors in Computing*, pages 1184–1187, 2005.
- [4] D. Blostein and A. Grbavec. Recognition of mathematical notation. In P.S.P Wang and H. Bunke, editors, *Handbook on Optical Character Recognition and Document Analysis*, pages 557–582. World Scientific Publishing Company, 1996.
- [5] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the Workshop on Computational Learning Theory*, pages 92–100, 1998.
- [6] C.M.L. Brown. Comparison of typing and handwriting in “two-finger typists”. In *Proceedings of the Human Factors Society*, pages 381–385, 1988.
- [7] H. Clark and S. Brennan. Grounding in communication. In L.B. Resnick, J. Levine, and S. Sag, editors, *Perspective on Socially Shared Cognition*, pages 127–149. APA Books, 1991.
- [8] R. Fateman. How can we speak math? <http://www.cs.berkeley.edu/~fateman/papers/speakmath.pdf>.
- [9] R.G.M Hausmann and M.T.H. Chi. Can a computer interface support self-explaining? *Cognitive Technology*, 7:4–14, 2002.
- [10] T. Kanahori, K. Tabata, W. Cong, F. Tamari, and M. Suzuki. On-line recognition of mathematical expressions using automatic rewriting method. In *Proceedings of the IEEE International Conference on Multimodal Interfaces*, pages 394–401, 2000.
- [11] J.L. Locke and F.S. Fehr. Subvocalization of heard or seen words prior to spoken or written recall. *American Journal of Psychology*, 85:63–68, 1972.

- [12] N.E. Matsakis. Recognition of handwritten mathematical expressions. Master's thesis, Massachusetts Institute of Technology, 1999.
- [13] E.G. Miller, N.E. Matsakis, and P.A. Viola. Learning from one example through shared densities on transforms. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 464–471, 2000.
- [14] S. Oviatt. Mutual disambiguation of recognition errors in a multimodal architecture. In *Proceedings of the ACM Conference on Human Factors in Computing*, pages 576–583, 1999.
- [15] S. Oviatt, R. Coulston, and R. Lunsford. When do we interact multimodally? cognitive load and multimodal communication patterns. In *Proceedings of IEEE International Conference on Multimodal Interfaces*, pages 129–136, 2004.
- [16] S. Oviatt, A. DeAngeli, and K. Kuhn. Integration and synchronization of input modes during multimodal human-computer interaction. In *Proceedings of the ACM Conference on Human Factors in Computing*, pages 415–422, 1997.
- [17] S. Smithies, K. Novins, and J. Arvo. Equation entry and editing via handwriting and gesture recognition. *Behaviour and Information Technology*, 20:53–67, 2001.
- [18] J. Sweller. Cognitive load during problem solving: Effects on learning. *Cognitive Science*, 12:257–285, 1988.
- [19] xThink. Mathjournal. <http://www.xthink.com/MathJournal.html>.