A model for effective real-time entry of mathematical expressions

Marco Pollanen[*,1] and **Michael Reynolds**[2]

[1] Department of Mathematics, Trent University, Peterborough, ON, K9J 7B8, Canada
[2] Department of Psychology, Trent University, Peterborough, ON, K9J 7B8, Canada

Many science and engineering courses contain substantial mathematical content. In addition, the rapid development of online learning, in both traditional and Web-based courses, and the ubiquitous presence of laptop computing, is placing increased demands on students to enter mathematical expressions into a computer. Furthermore, it is often required that the expressions are written in real-time (e.g., in class note-taking, automated assessment in a timed testing environment, or for a live online discussion). Technology-based difficulties that affect the writing of expressions will result in inaccurate assessment or may impede the transfer of knowledge.
Obstacles to the easy input of mathematical expressions include the large number of specialized symbols in mathematics and the two-dimensional layout structure of many mathematical expressions. Most software packages, such as Microsoft Word, that allow complex mathematical content to be input employ a structure-based approach, where symbols and structural elements are chosen from palettes and input separately.
In this paper, we argue that structure-based expression editors, which were largely designed for document creation, are not appropriate for real-time tasks in education. They force a user to write expressions differently than they would on paper, and so interfere with normal thought processes. As an alternative, we introduce a new model for expression entry that, like a structure-based approach, uses palettes for symbol selection, but for structures allows users to freely place symbols as they would on paper. Supported by usability studies, we argue that this hybrid approach allows for the much more effective entry of mathematical content.

Keywords Formula Input; Mathematical Communication; Mathematical User Interfaces

1. Introduction

Digital communication tools are evolving at an astonishing pace due to the growth of the Internet. Indeed, recent evidence suggests that 98% of U.S. post-secondary faculty use the Internet to communicate with their students and that many are quick to embrace new digital communication tools [1]. The adoption of digital communication tools is widely seen as a an important pedagogical development since it is known that increased outside-the-classroom student-teacher contact (e.g., office hours, e-mail) in a post-secondary environment correlates positively with key educational indicators (e.g., academic performance, student retention, and student satisfaction) [2]. The use of text-based communication technologies is ubiquitous (e.g., e-mail, discussion groups, chat rooms, and instant messaging). In contrast, none of the faculty in the survey used video or audio conferencing to communicate with students. The heavy reliance on text-based communication technologies is problematic for academic subjects that make extensive use of mathematical expressions or other non-text content. The purpose of the present paper is to examine different mathematics user-input models and to determine which one would be most effective for real-time communication, especially by novice users (e.g., students).

The most common platform for accessing the Internet is the personal computer equipped with a keyboard and mouse (or equivalent pointing device). The hardware user-input interface for the common personal computer, which has evolved from the English-language QWERTY typewriter, originally relied solely on a keyboard and later on a keyboard/mouse combination, with digital pens and speech recognition playing a small but growing role. The QWERTY keyboard and mouse interface is not well suited for languages that do not use the alphabet-based writing systems. This is true both for logographies (e.g., Kanji) and mathematics, as can be seen by considering the following expression from a first-year calculus course:

$$\Delta \geq \int_{\alpha}^{\beta} \sqrt[3]{\frac{x^2-1}{x-1}}\,dx. \qquad (1)$$

From Expression (1) it is apparent that the digital communication of mathematics faces several major challenges not present in text-based communication. Firstly, there many more symbols than there are simple key combinations on a keyboard (e.g., “ctrl r” for √) and many of the symbols are not visually similar to alphabet characters, resulting in unintuitive mapping of symbols (the symbol problem). The second major challenge is

[*] Corresponding author: e-mail: marcopollanen@trentu.ca

that many mathematical expressions have a non-sequential layout (the layout problem). A third factor that compounds the challenges of writing mathematics is that, it is difficult to communicate expressions in everyday language; even "oral" mathematical communication relies on a facilitating interface, such as a chalkboard, for effective communication.

To date, most digital writing methods for mathematics have focused on document creation, predominantly for mathematically sophisticated users. Consequently, their primarily goal is to produce mathematical expressions with a clear, idealized, visual presentation. It is currently unclear whether traditional mathematical input methods are well-suited for real-time communication applications. Arguably, real-time communication interfaces need to balance speed and ease of communication with readability and usability for novices (i.e., students) if they are to be adopted in an educational setting. For instance, communicating with a student in real-time when they made an error requires user behaviours not present during document creation. This should influence interface design choices.

2. Software Methods for Mathematical Expression Input

During the past several decades a number of approaches have been developed for the digital input of mathematical expressions. Before the widespread use of graphical user interfaces (GUIs), text-based languages such as TeX [3] were developed for mathematical input. With the advent of the GUI, graphical editors were developed (e.g., Microsoft Equation Editor). In the this section we provide an overview of several methods of mathematical expression input. In Section 3 we discuss which of these models is most effective and efficient for communication applications for novice users.

2.1 Text-Based Communication of Mathematics

As most tools for communicating over the Internet are text-based, it is common for students to communicate mathematics via text-based technologies such as e-mail. For simple expressions, this can be a quick and efficient method for communicating mathematics. However, more complex expressions may contain mathematical symbols and have a two-dimensional layout.

For many symbols it is possible to write text-based synonyms that would be universally known. For example, most post-secondary students would known that alpha and α describe the same character. However, many students would not know that Ξ is capital xi, while other symbols, such as ∟, have no commonly accepted name. Even common symbols such as ≥ can present difficulties. The description "greater than or equal to" is long, and so is sometimes written as >=, which may not be clear to somebody who has never seen it before. Writing it as => is also possible, but could be misinterpreted visually as an arrow.

The layout problem represents an even greater challenge for text-based mathematics. It is hard to unambiguously describe the structure of many mathematical expressions using natural language. This is due, in part, to the two-dimensional layout of mathematical expressions resulting in nested substructures. Structural ambiguities are often overcome by introducing parentheses or braces to indicate the precedence of structures. However, many students in introductory university mathematics courses have difficulty with precedence and the proper use of parentheses. In other cases, ambiguities can be introduced by representation styles. For example, consider the expression 1/2x. Interpreting this sequentially would lead one to believe that it represents 0.5*x. However, some readers could interpret the division sign pictorially as a structural element and assume it represents a horizontal division bar, in which case the expression is 0.5/x. Without a pre-defined standard, the communication of mathematics using text can be problematic.

There are many text-based languages for describing mathematics. For example, using the popular typesetting language TeX, Expression (1) can be written as **"\Delta \ge int_{\alpha}^{\beta} \sqrt[3]{\frac{x^2-1}{x-1}} dx**". The difficulty of using this approach is apparent; the reader(s) and writer(s) of such an expression need to (1) know the text-based synonyms, and (2) be able to parse this expression to appreciate its two-dimensional layout. While TeX is text-based, many TeX editors allow synonyms to be inserted by selecting symbols from graphical palettes. This alleviates the symbol problem for the writer(s), and so allowing symbol synonyms to be inserted from palettes should be a design consideration for any text-based environment designed specifically for mathematics.

2.2 Structure-Based Graphical Input

The most common non-text method for digitizing mathematical expressions is the graphical structure-based editor. An editor of this type (e.g., Microsoft Equation Editor) is found in most office productivity software suites.

Graphical structure-based editors (see Fig. 1), typically require that a user enter a mathematical expression by selecting separate structural and symbol elements from palettes. The structure elements subdivide the writing

space into regions (often represented by boxes in the element) that can then contain a sequential expression or further nested structural elements.

One fundamental problem with these editors is that, like text-based methods, they force the writer of the expression to pre-parse their expression and change their natural writing order [4]. For example, the expression $\frac{\sqrt{x}}{y}$ would normally be handwritten in the order √, x, −, y, however, structure-based editors typically force the user to first input the fraction bar and then the root before populating the structure with the x and y.

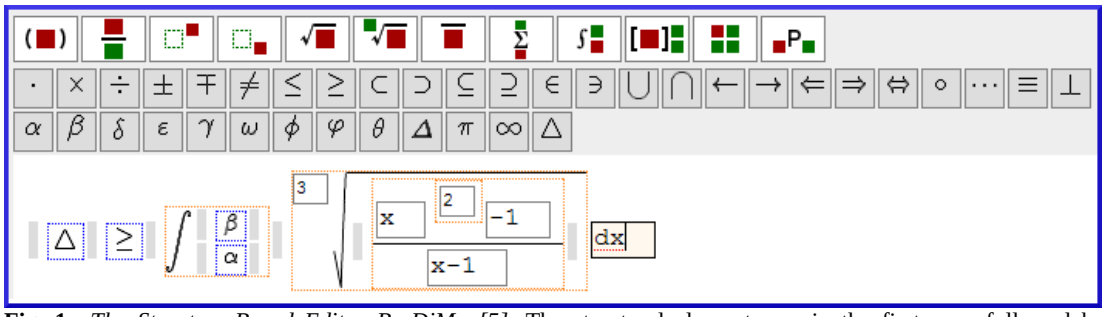

Fig. 1 *The Structure-Based Editor BreDiMa [5]*. The structural elements are in the first row, followed by symbol elements in the next two rows. Structurally subdivisions of the expression are represented by outlined boxes.

Another problem with structure-based editors is that the nested nature of structures makes it difficult to edit the overall structure of an expression (e.g., to simplify an expression once it has been written). Consequently, users commonly restart an expression from scratch, rather than try to edit it [4]. It is also difficult to navigate the sub-expressions in an intuitive way, and generally these editors violate many WYSIWYG design principles [6].

2.3 Drawing Mathematical Expressions

A common software interface that allows users to communicate ideas visually is a whiteboard. Whiteboards can be found in many learning management systems as well as some instant messaging applications. Conceptually, a whiteboard interface is meant to emulate a physical whiteboard, and therefore typically contains a multi-user shared canvas that allows users to "draw'' as they would using a paint program. Usually, drawing tools for objects such as arrows, boxes, and circles, are provided to facilitate image creation.

The concept of a whiteboard has been specialized for mathematics in the communication package enVision [7]. In addition to the typical whiteboard drawing tools like boxes and circles, enVision also contains similar resizable drawing tools for mathematical symbols such as integrals, radicals, and brackets (see Fig. 2). Arguably, as most users have had experience with a paint application, they should be familiar with the interface concept and so new users should find it intuitive. Users who have a digital pen, or equivalent, input device could also use enVision to handwrite mathematics on the whiteboard surface.

Since enVision is modelled after a paint program, the drawing is treated like an image. Although it has some basic editing features, such as undo, most deleting is done using a graphical pixel-based eraser tool. This makes modifying expressions inefficient, because the expression is a single image and not a collection of objects. Deleting part of an expression, therefore, often requires many strokes with the eraser tool, similar to how one erases an image on a chalkboard, instead of indicating what symbols/structures to delete. Another drawback of the enVision interface is that, as the images are only visualizations of mathematical ideas, they cannot be exported into other formats, such as the Microsoft Equation Editor format. However, there are experimental pen-based interfaces that recognize handwritten mathematics, for example FFES [8], so improvements may be possible in this regard in the future.

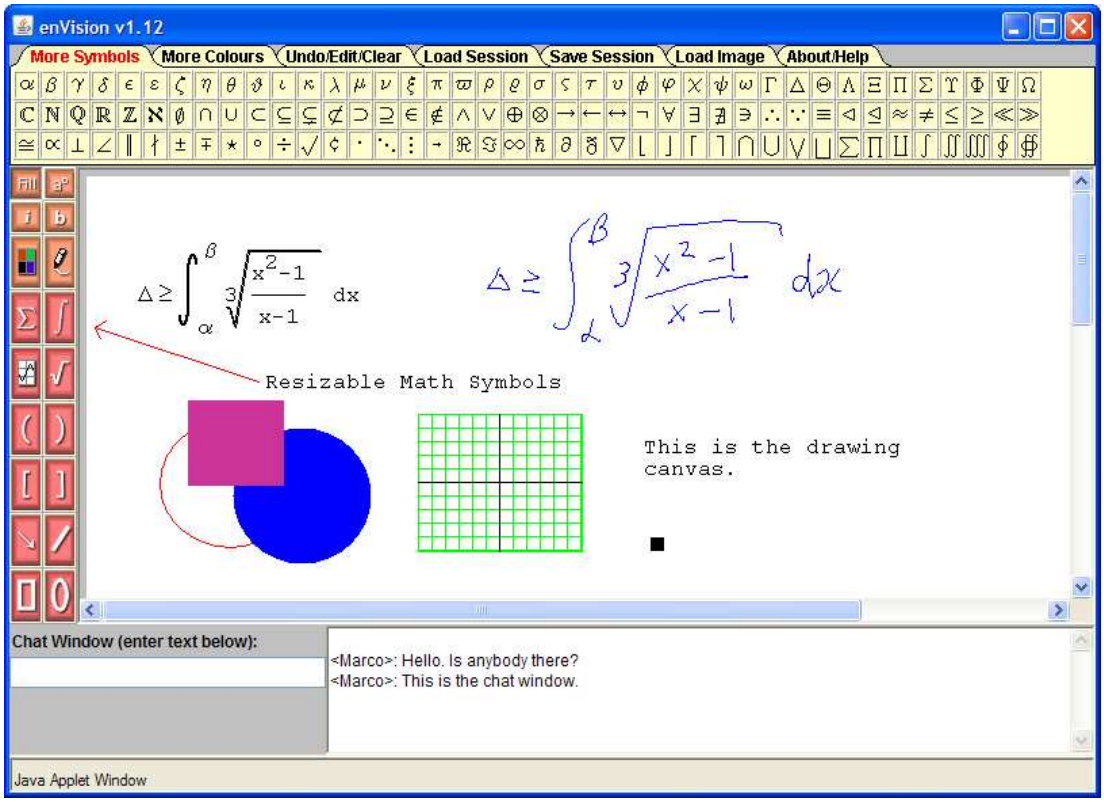

Fig. 2 *The whiteboard interface enVision*. There is a chat window as well as a shared mathematics whiteboard. In addition to drawing tools found in a typically whiteboard, it contains sizeable drag-and-drop mathematics symbols.

2.4 Diagrammatic Input of Mathematical Expressions

Another type of user interface that allows direct access to a two-dimensional canvas for writing mathematics can be found in XPRESS [9]. However, unlike enVision, which is modelled after a paint application, XPRESS is modelled after a diagram editor (see Fig. 3).

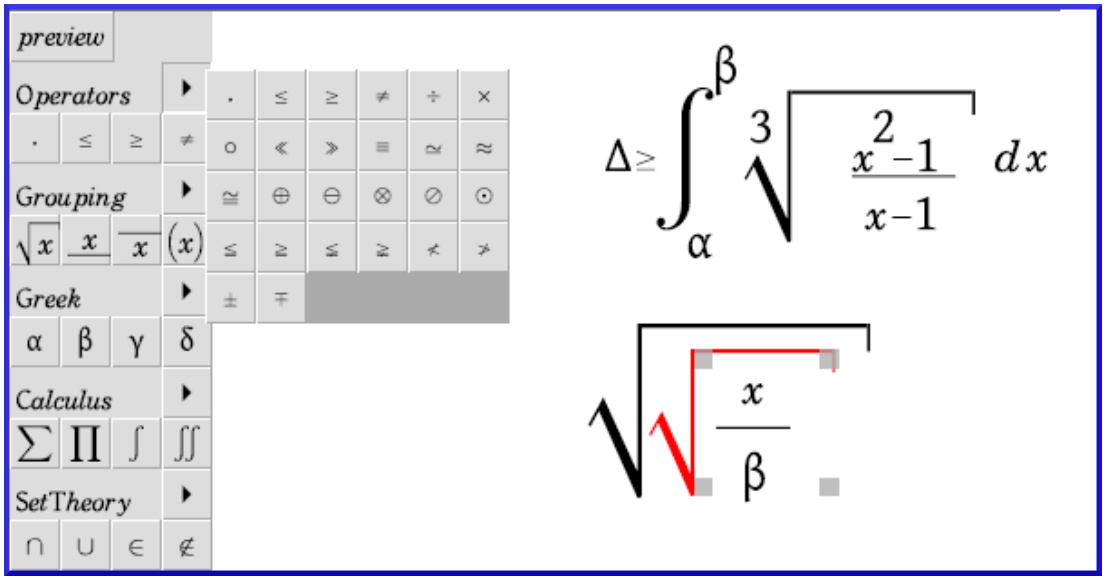

Fig. 3 *The Diagrammatic Editor XPRESS*. This is a palette-based expression input system that allows users to draw mathematical expressions in a similar way to a diagram editor. Note: in this figure the Operators symbol palette is expanded and one of the square root objects is selected showing its resizing widgets.

XPRESS allows users to select symbols from a palette and place them freely anywhere on the canvas. In contrast to a paint application, which is usually for creating pixel-based drawings, each object in XPRESS has a set of attributes, such as location and size, and can be independently moved and resized. Consequently, it is possible to select, move, delete, and resize the symbols (or groups of symbols) at a later time. Once an expression is in its final form, XPRESS is able to apply a spatial analysis algorithm to recognize the expression, and so that it could then be exported in a format usable by other applications.

3. Discussion

Each of the graphical models, and some implementations of the text-based model, solves the symbol problem by having users graphically select symbols from palettes. So the fundamental difference between the four approaches is how spatial structures are created and modified. In the structure-based approach the user is constrained as to where and in what order they can place symbols, whereas both the drawing approach and the diagrammatic approach allow users to place symbols freely anywhere and in any order on a virtual canvas. All three graphic editors differ in how symbols are manipulated. A structure-based approach affords only minimal editing. The drawing approach best resembles pen-and-paper in that the entire canvas is treated like a single drawn image, and so symbols cannot be easily manipulated, and deleting is accomplished by a applying a digital eraser to the overall image. Only the diagrammatic approach, in which each mathematical symbol is an individual object on the canvas, allows for the easy resizing, deleting, and moving of symbols and structures. This is an important consideration in a teaching environment where a problem is worked on step-by-step, or where users are expected to make many mistakes.

In the study [4] comparing how novice users write expressions with a diagrammatic editor versus a structure-based editor, it was shown the novice users wrote expressions in a similar order than they would on paper with a diagrammatic approach, while a structure-based approach forced users to change their writing style. This suggests the diagrammatic approach is more intuitive. Furthermore, the subjects wrote a set of expressions marginally faster (5%) using a diagrammatic editor than with a structure-based editor. However, a large percentage of the time (24%) with the diagrammatic editor was spent on aesthetic changes to the expression that did not alter the expression meaning. It is possible that users might spend less time making these cosmetic changes in a real-time communication setting where time pressure is a factor. For example, in text-messaging spelling and punctuation are often sacrificed for speed.

4. Conclusion

In this paper we considered some of the difficulties communicating mathematics online in real-time. We also examined solutions to this problem currently implemented in user-interfaces available for mathematical input: text-based input, structure-based input, drawing environments, and diagrammatic environments.

Overall, given that the diagrammatic model appears to be the most intuitive for novice users, potentially has a speed advantage, and is the only graphical approach that allows for the easy editing and manipulation of content, suggests that it is the best underlying model for future mathematical communication interfaces.

Acknowledgements This research was supported by a grant from the Social Sciences and Humanities Research Council of Canada (SSHRC).

References

[1] S. Jones and C. Johnson-Yale, First Monday 10, online (2005)
[2] M.K. Nadler and L.B. Nadler, Communication Studies 51, 176-188 (2004)
[3] D.E. Knuth. The TeXbook, (Addison-Wesley, Reading, MA, 1984)
[4] D. Gozli, M. Pollanen, and M. Reynolds, preprint (2009)
[5] Y. Nakano and H. Murao, Proceedings of MathUI 2006, Workingham, UK, 10 August 2006, online.
[6] L. Padovini and R. Solmi, Lecture Notes in Computer Science 3119, pp. 302-316 (2004)
[7] M. Pollanen, Journal of Online Mathematics and its Applications 4, online (2006)
[8] R. Zanibbi, D. Blostein, and J.R. Cordy, Proceedings of the 6th International Conference on Document Analysis and Recognition, Seattle, USA, 10-13 September 2001, pp. 768-773.
[9] M. Pollanen, Proceedings of MathUI 2007, Linz, Austria, 27 June 2007, online.