A Tablet-Compatible Web-Interface for Mathematical Collaboration

Marco Pollanen[1], Jeff Hooper[2], Bruce Cater[3], and Sohee Kang[4]

[1] Trent University, Canada
marcopollanen@trentu.ca
http://euclid.trentu.ca/math/marco/
[2] Acadia University, Canada
jeff.hooper@acadiau.ca
http://math.acadiau.ca/hooper/
[3] Trent University, Canada
bcater@trentu.ca
http://www.trentu.ca/economics/staff_cater.php
[4] University of Toronto Scarborough, Canada
soheekang@utsc.utoronto.ca
http://www.utsc.utoronto.ca/cms/sohee-kang

Abstract. Mathematical novices – including students in introductory mathematics and statistics service courses – increasingly need to engage in online mathematical collaboration. Using currently-available interfaces for their mobile and touch-enabled devices, however, this group faces difficulties, for those interfaces are text-based and not directly suitable for mathematical communication and collaboration.

To address the deficiency of digital input methods and interfaces for mathematics, we introduce a cross-platform synchronous communication interface for mathematical collaboration. The interface is designed to be intuitive for multiple user groups ranging from novices to experts. We demonstrate that it is possible to create a Web-based communication interface that simultaneously incorporates TeX-, palette- and pen-based input methods, and that is compatible with both touch-enabled tablet and traditional keyboard-mouse user interface principles. The design principles we introduce may be valuable for the design of other mathematical user interfaces on touch-enabled devices, such as with Computer Algebra System interaction.

Keywords: Mathematical Collaboration, Mathematical User Interfaces, Formula Input

1 Introduction

Communication technologies are now used to great effect in post-secondary education, increasing, for example, outside-the-classroom student-teacher contact. This type of interaction correlates positively with key educational indicators, including academic performance, student retention and student satisfaction [8].

H. Hong and C. Yap (Eds.): ICMS 2014, LNCS 8592, pp. 614–620, 2014.

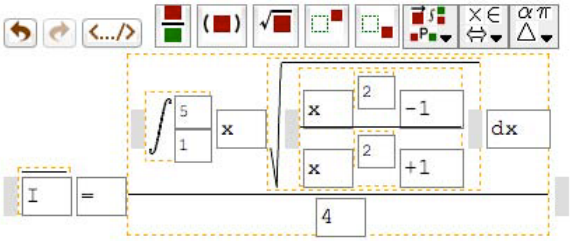

Fig. 1. BrEdiMa: A Structure-based Editor for Mathematical Input [9]

The use of technology for the communication of mathematical ideas can be a particularly effective pedagogical tool [4]. Its potential, however, has yet to be fully realized. This may be due to two factors: most internet communication technologies are text-based and both the text and non-text based mathematics input methods that have been developed have been designed mostly for experts, rather than for the increasingly novice student user-base.

2 Barriers to Communication

Communicating mathematics online using text-based technologies is problematic for two reasons. There are hundreds of commonly used mathematical symbols, many of which have no commonly accepted textual equivalent and must therefore be described. The inherently two-dimensional structure of mathematical notation requires spatial relationships between symbols; such relationships are difficult to communicate in inline text. Consequently, the standards that exist for the text-based entry of mathematics suffer from having a steep learning curve and very low human readability. For instance,

```
\lim_{x\to\infty}\frac{\sqrt{8+x} - 3x^{1/3}}{x^2 -3x +2}
```

and

```
\int_0^2 r \sqrt{ 5 - \sqrt{ 4-r^2}} dr}
```

are LaTeX representations for the two first-year university calculus expressions

$$\lim_{x\to\infty} \frac{\sqrt{8+x} - 3x^{1/3}}{x^2 - 3x + 2} \quad \text{and} \quad \int_0^2 r\sqrt{5 - \sqrt{4 - r^2}}dr \ .$$

Neither LaTeX string is intuitive for novices, and small errors in either can seriously affect the mathematical meaning.

The main alternative to inline text-based input is structure-based direct-manipulation editors, such as those found in Microsoft Word or in BrEdiMa (Figure 1).

In such an editor, the user inserts individual symbols and mathematical structures from palettes of symbols. As with inline text editors, however, structure-based direct-manipulation editors suffer from severe usability problems [12]. In

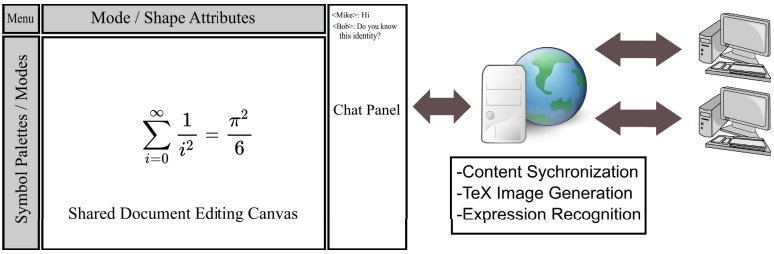

Fig. 2. Schematic of iCE UI Layout and Communication

[14] it is argued that structure-based editors usually force a user to write a formula in a different manner than they would on paper. For example, consider the expression $\sqrt{x}/y$. The default behavior of a structure-based editor forces the user to input the fraction first, followed by the square root symbol, and then the x and y. Intuitively, however, somebody writing this expression with pen and paper would probably write the square root of x first followed by the fraction bar and then the y. So in essence, the user of a structure based editor has to use an unintuitive order to input the expression. This requires the user to have the ability to mentally parse the desired mathematical expression and reorder for input, which can be difficult for students and other novice users. The user must adapt to the technology as it is non-intuitive and has not been designed with casual users in mind.

While digital pen-based input methods would allow users to write mathematics as they would on paper, their use introduces new problems. For instance, robust handwriting recognition algorithms for mathematics are still in their infancy, and pen-input typically requires special hardware that most students do not possess.

In this paper we outline the development of a real-time multi-modal web-based open-source mathematics collaboration interface that works on all commonly available computing devices, ranging from computers to tablets to smartphones, and is intuitive for first-time users who are mathematical novices.

3 Interface Choice

The goal of iCE (interface for Collaborative Equations) is to be a hybrid environment that allows users familiar with any mathematical editor input model (e.g. palette/structure based, TEX, or pen-based) to communicate and collaborate mathematically as quickly and effortlessly as possible. It is structured as a shared SVG document simultaneously being edited by multiple users in a collaborative whiteboard model with a chat-pane on the side to emulate verbal conversation. This model best replicates the usual in-person mathematical collaboration model where a mathematical conversation is usually assisted by facilitating technology, such as a piece of paper or chalkboard [3].

iCE allows for unconstrained input, permitting users to enter mathematical symbols in any order. This approach has been shown to allow faster input, has a minimal learning curve ([3] and [15]) and allows users to enter mathematics as they usually do on a chalkboard and thus minimizes destructive interference and cognitive load [11].

4 Browser-Based Interfaces

With the advent of Web 2.0 and cloud computing, the web browser is increasingly becoming the standard interface to access full-featured applications. In particular, Google Docs and Office 365 have become mainstream browser-based environments for collaborative documents. Thus, in developing a cross-platform mathematics collaboration environment, it is natural for it to be browser-based as well.

The goal of iCE is to create a communication interface that can incorporate collaboration into a variety of web applications through any major browser (Internet Explorer, Chrome, Firefox, Safari) across multiple platforms (PC, Mac, Linux, iPad, iPhone, and Android Smartphones) without installing any software or plugins.

Web-based applications can be difficult to develop due to the application being embedded in an existing browser interface; the browser framework leads to many UI restrictions, limited protocol support, and sandbox restrictions. A cross-browser application must also deal with many API inconsistencies. When both personal computers and mobile devices are to be supported, the application's design must take into account a number of major differences between client instances: keyboard/mouse versus touch UI, screen size, and input accuracy. In addition, mobile devices typically have limited computational power and so CPU intensive operations must be delegated to web services.

Browser-based applications for communication of or editing of mathematical content face additional difficulties, as in-browser layout and display of mathematical expressions is problematic, and even more so if the content is made to be interactive: copied, scaled, and manipulated. While Mathematical Markup Language (MathML) was intended to be the standard for mathematics on the internet, it is thus far not fully supported [1]. For these reasons, iCE has been designed around a front-end of Javascript/SVG (based on SVG-Edit [16]) with calls to several web services to keep it lightweight.

5 User-Interface Interactions

iCE implements an unconstrained direct editor equation model based on a diagram editor. Users are free to place elements anywhere on a whiteboard canvas. However, in this case the elements are not restricted to geometric objects, but include resizable mathematical symbols. This approach is consistent with both mouse and touch-based UI interactions and elements may be manipulated as follows:

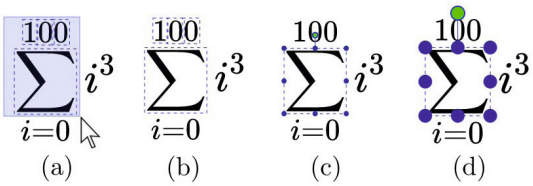

Fig. 3. Object manipulation in iCE: (a) selecting a group of symbols; (b) the symbol group after selection; (c) resizing widgets on a PC; (d) resizing widgets on an iPad.

Moving a Symbol: Individual symbols on the canvas may easily be dragged to any location.

Group Selection: By pressing down (via mouse or finger) on an empty spot of the canvas and dragging out a rectangular outline, all symbols falling inside the rectangle are selected. They will then be treated as a group until unselected by clicking or tapping on a blank section of the canvas. The group of symbols in a selection may, collectively, be dragged to a new location, have their attributes changed, or be deleted.

Resizing: By clicking or tapping on a symbol, resizing widgets appear and that individual symbol may be resized by dragging these widgets. Due to the limited precision of the finger on a touch-based device, the size of these widgets will vary depending on the device. See Figure 3.

In any equation input system the keyboard plays an important role. Since hardware keyboards have fixed keys this makes the symbol mapping complicated. On the other hand, touch-based devices rely on virtual keyboards. While custom virtual keyboards can include mathematics symbols, virtual keyboards on touch devices generally occupy a large percentage of the screen and have problems with users accurately selecting the correct symbol.

Keyboard input on iCE takes one of three forms: text-mode, symbol-mode, and TEX mode. Consistent with a diagram editor, *text-mode* may be selected and a corresponding text-box may be placed anywhere on the screen. Within each text-box, the text may use different fonts with attributes appropriate for labeling a diagram, such as size, style, color, etc. Synchronization across participants occurs only upon completion of the entire text.

On the other hand, the symbol mode incorporates a persistent on-screen cursor that can input text or symbols using shortcuts from the keyboard while it is in almost any mode. The cursor may be placed anywhere on the canvas by cursor keys or by clicking (tapping) on an empty space. Unlike text-mode, symbol-mode is synchronized across participants in real time.

To accommodate the large number of possible math symbols, many symbols are mapped to the same key. By pressing a key in rapid succession a number of different symbols are cycled through (for example, aside from its usual use, the keyboard key 'A' cycles through 'A', alpha (α), for all ($\forall$), logical 'and' ($\wedge$),

aleph (ℵ), and the angle symbol (∠). When the mapping is not obvious, symbols may also be inserted by selecting them from a palette.

The final keyboard mode is a variation of the text-box mode. Users enter TEX code into a text-box and on completion the TEX code is compiled and turned into symbolic content through a web service call. This symbolic content is then redistributed to all users.

6 Web Services

iCE is a collaborative interface and thus makes use of several network components. In its implementation, it relies on a server built using node.js [10] for communication synchronization. To keep iCE lightweight, additional web services can be utilized for CPU-intensive tasks. Currently, TEX code is converted to SVG content through the use of MathJax [6] as a web service. In addition, selections of symbols may also be converted to TEX through a web service call to the spatial recognition algorithm Xpress [14].

7 Further Enhancements

Other web services we plan to implement to make the interface truly multimodal:

Handwriting Recognition: iCE allows for pen input (if available on the platform). However, this is maintained as a digital ink layer and is not processed further. There are a number of handwriting recognition approaches (e.g., [17]) for mathematics that could be explored in the future. Much like the current call to Xpress for TEX rendering, this intensive task is best left on the server and not implemented in client-side Javascript.

Voice Recognition: The HTML5 API [5] allows for Javascript to access a browser's audio stream which may allow iCE input to be paired with mathematical voice recognition [2].

CAS Assistance: It may be possible to make communication faster and more effortless by allowing calls to a server-side Computer Algebra System (CAS) to assist with calculations. This would involve an additional parsing component, since this requires using content markup as opposed to the use of presentation-oriented markup like TEX.

8 Conclusion / Discussion

Communication technology in mathematics lags far behind its use in other academic disciplines. In this paper we introduce a multi-modal mathematics collaboration interface that is designed to be fast and intuitive for both novice and expert users. We hope that this new interface approach will lead to improvements in the design of future interfaces for mathematical input that will have a positive impact on mathematical education and collaboration.

References

1. Cervone, D.: MathJax: a platform for mathematics on the Web. Notices of the AMS 59(2), 312–316 (2012)
2. Fateman, R.: How can we speak math? (2013), http://www.eecs.berkeley.edu/~fateman/papers/speakmath.pdf (Unpublished manuscript)
3. Gozli, D.G., Pollanen, M., Reynolds, M.: The characteristics of writing environments for mathematics: Behavioral consequences and implications for software design and usability. In: Carette, J., Dixon, L., Coen, C.S., Watt, S.M. (eds.) Calculemus/MKM 2009. LNCS (LNAI), vol. 5625, pp. 310–324. Springer, Heidelberg (2009)
4. Hooper, J., Pollanen, M., Teismann, H.: Effective online office hours in the mathematical sciences. Journal of Online Learning and Teaching 2(3) (2006)
5. HTML 5.1 API, http://www.w3.org/TR/html51/
6. MathJax homepage, http://www.mathjax.org
7. Miner, R.: The importance of MathML to mathematics communication. Notices of the AMS 52(5), 532–538 (2005)
8. Nadler, M.K., Nadler, L.B.: Out-of-class communication between faculty and students: A faculty perspective. Communication Studies 51(2), 176–188 (2000)
9. Nakano, Y., Murao, H.: BrEdiMa: yet another Web-browser tool for editing mathematical expressions. In: Proceedings of Math UI 2006 (2006)
10. Node.js homepage: http://nodejs.org/
11. Oviatt, S.L., Arthur, A.M., Brock, Y., Cohen, J.: Expressive pen-based interfaces for math education. In: CSCL 2007, pp. 573–582 (2007)
12. Padovani, L., Solmi, R.: An investigation on the dynamics of direct-manipulation editors for mathematics. In: Asperti, A., Bancerek, G., Trybulec, A. (eds.) MKM 2004. LNCS, vol. 3119, pp. 302–316. Springer, Heidelberg (2004)
13. Pollanen, M.: Interactive Web-based mathematics communication. Journal of Online Mathematics and its Applications 6(4) (2006)
14. Pollanen, M., Wisniewski, T., Yu, X.: Xpress: a novice interface for the real-time communication of mathematical expressions. In: Proceedings of MathUI 2007 (2007) (online)
15. Pollanen, M., Reynolds, M.: A model for effective real-time entry of mathematical expressions. Research, Reflections and Innovations in Integrating ICT in Education, Formatex, pp. 320–324 (2009)
16. SVG-Edit homepage: https://code.google.com/p/svg-edit/
17. Tapia, E., Rojas, R.: Recognition of on-line handwritten mathematical formulas in the E-Chalk System. In: Proceedings of the Seventh International Conference on Document Analysis and Recognition, pp. 980–984 (2003)