

Cooperative Multi-Application Sketching

Ryan Dixon and Timothy Sherwood
University of California, Santa Barbara
Santa Barbara, CA 93106-9560
{rsd,sherwood}@cs.ucsb.edu

ABSTRACT

Existing sketch-based applications are typically developed and tailored to a specific domain and packaged as atomic applications that run independently of one another. We propose a new approach to sketch application development whereby free-form input is treated as a first-class citizen, data is freely shared among concurrently running sketch applications, and the act of sketching is the main mode in which new applications are invoked.

ACM Classification: H5.2 [Information interfaces and presentation]: User Interfaces. - Graphical user interfaces.

General terms: Design, Human Factors, Experimentation, Languages

Keywords: Human computer interaction, whiteboard computing, sketch recognition

Introduction

For many engineers, whiteboards provide a means of describing and communicating the relationships between complex interacting parts. Although digital interaction with whiteboard sized surfaces is not a new concept [2, 5], most existing whiteboard systems provide only rudimentary support for sketch interaction, permitting the recording and replaying of input and little else. While more sophisticated systems exist for tablet devices, even these are limited primarily to sketch-based front-ends to other monolithic back-end applications. Users invoke a specialized application in a new window (such as a circuit recognizer) and then proceed to draw and simulate domain specific symbols (such as gates and wires) which can then be simulated in place. The basic idea behind our presented work is to create a system that captures and computationally augments the multitasking that naturally takes place across the space of a whiteboard. Under our system a user is free to draw and simulate a state machine, solve an equation, and test a circuit all at the same time.

In this presentation we will demonstrate three solutions to problems previously unaddressed by the sketch recognition community relating to co-executing and cooperating applications. The key to this approach is a digital whiteboard architecture that shares the state of the board with all applications,

the use of strokes and notations on strokes as the primitives for both rendering and communication, and the ability of applications to be invoked implicitly through the creation of drawings of interest to them.

Problem 1: Concurrent Applications

In a whiteboard setting it is common to have many ideas expressed across different portions of the board at any given time, multiple users may even share different regions of the board. If we expect a digital whiteboard to behave in a manner that closely resembles a traditional whiteboard while adding the ability to augment drawn strokes with the results of computation, it is imperative that multiple applications be able to run side-by-side. We have developed a system that does just that, allowing an arbitrary drawing order with no bounds on the particular spatial regions that an application must occupy.

To provide this ability the system must treat free-form input as a first-class citizen, and strokes must be openly available to all applications. In our framework, free-form input is the elementary building block on top of which all computation is performed, the only requisite for applications designed for this system is compliance with a minimal interface. Each application is alerted whenever a stroke—or set of strokes—is added, removed, or edited. This avoids the issue of incorrect stroke “ownership”. When a line is drawn it is difficult to tell which application should be interested in that stroke (is it the digit 1 or an edge in a graph?) and incorrect ownership decisions are incredibly difficult to discover and undo. Instead, it is the responsibility of each individual application to determine what that sequence of events means with respect to its own computation. Figure 1 provides an example of three applications running concurrently on our system.

Problem 2: Inter-Application Communication

One limiting factor in this area of research is the complexity involved in parsing and understanding human sketches. Although research has made progress towards categorizing, describing, and recognizing free-form input [1, 3], accuracy rates dwindle when compared to more constrained input scenarios, such as handwriting recognition. Unfortunately, most recognition systems are designed independently and operate within their own unique environment; the highly specialized nature of these hand-crafted recognizers inhibits their use in other applications. In our prototype system we felt it was necessary to allow the integration of both general purpose and domain specific recognition/computation to be shared.

In our framework, every application running on the system

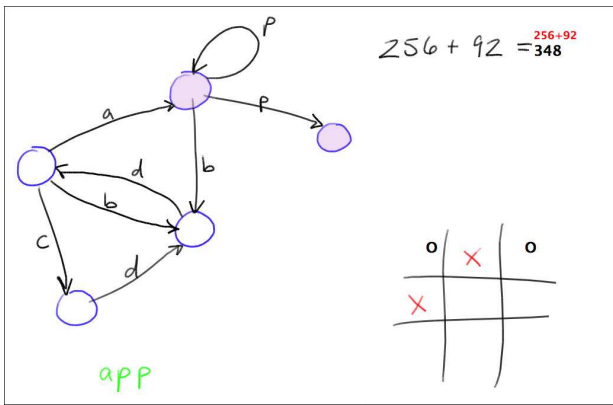


Figure 1: A screen capture of three concurrent whiteboard applications: an equation solver, Tic-Tac-Toe, and a finite state machine.

has the ability to view and tag any stroke with metadata. This metadata can then be viewed and used by other applications. In this manner code reuse is encouraged and existing solutions can be leveraged by new applications. In Figure 1, for example, a text recognizer samples all user input and assigns character values to individual strokes which an equations evaluator then reads and reacts to. The finite state machine application also uses this text metadata to understand labels written along the graph's edges.

Problem 3: Application Instantiation

One of the most important directions of this work is in providing the ability for the whiteboard to naturally observe drawn strokes and react to them. An important ramification of this direction is that the act of application invocation need can be implicit in a specific ordering of strokes. For example, in a Tic-Tac-Toe application, the game itself does not exist until a full board has been drawn; once the final line is in place (regardless of the order in which it was drawn) a new Tic-Tac-Toe game is instantiated and loaded into memory. Similarly, once part of a complete board is erased, the game is removed. The user can easily run multiple instances of the Tic-Tac-Toe application by drawing another game anywhere on the whiteboard surface, even interleaving the two games arbitrarily.

Related Works

Over the years there has been a significant amount of research dedicated to interpreting and understanding free-form input with respect to a single domain, such as handwriting. While our work certainly benefits from this past research, our current system draws most of its inspiration from research in multi-domain sketch recognition and a number of recently developed tablet-based scientific applications [1, 4, 6]. Our efforts are unique in that we are targeting large surfaces as our primary platform. While we utilize many of the stroke segmentation and identification techniques described in the literature, we are attempting to create a space where multiple applications can coexist and be combined to solve complex problems as a typical user would expect to solve problems in a whiteboard setting.

We are not the first to propose a whiteboard interface. In the early 1990s, Xerox created a commercial product called the “LiveBoard” in an attempt to provide a digital whiteboard suitable for workgroup meetings [2]. While we do share similar goals with the LiveBoard, our vision of a whiteboard computing environment is dramatically different. While the LiveBoard focused mainly on the ability to record and present information, our focus is on computation.

Design

Our current work is focused on developing a whiteboard environment where users can freely sketch and view computations on-the-fly. Our design approach can, in many ways, be viewed as a generalization of the spreadsheet; portions of the board may interact and affect other regions of computation. As soon as a stroke is drawn, changes are immediately presented to the user. Figure 1 provides a sample of the applications we have built thus far. These three applications exemplify the intended interaction with the whiteboard. For example, as input is written for the the finite state machine (the bottom text in Figure 1), the active states are automatically updated and highlighted. Likewise, when edges and nodes are added or removed from the state machine, changes are reflected immediately in the new graph.

The most recently developed application within our framework is the equation solver. While former work, such as [4], has demonstrated success in parsing written equations, the equation solver was built in an attempt to highlight the extensibility of our framework. We believe that providing a set of core utilities, such as an equation solver, will dramatically increase programmer productivity. With the equation solver in place, future sketch applications can simply incorporate the solver's results and leave the details of understanding equations to the solver.

Conclusions

This work presents an ongoing effort in the creation of multitasking whiteboard computing environments. Designing an efficient, extensible, and easy-to-use system in the presence of potentially ambiguous data is a difficult task; however, we believe our framework is a step in the right direction. By exporting a minimal interface and promoting inter-application data sharing, this framework lays the groundwork for a number of sophisticated whiteboard applications.

REFERENCES

1. Christine Alvarado and Randall Davis. Dynamically Constructed Bayes Nets for Multi-Domain Sketch Understanding. In *Proceedings of IJCAI-05*, pages 1407–1412, San Francisco, California, August 1 2005.
2. Scott Elrod et al. Liveboard: A Large Interactive Display Supporting Group Meetings, Presentations and Remote Collaboration. In *CHI'92*.
3. Tracy Hammond and Randall Davis. LADDER: A Language to Describe Drawing, Display, and Editing in Sketch Recognition. In *IJCAI*, pages 461–467, Acapulco, Mexico, 2003.
4. Joseph J. Laviola and Robert C. Zeleznik. MathPad2: A System for the Creation and Exploration of Mathematical Sketches. *ACM Trans. Graph.*, 23(3):432–440, August 2004.
5. Luidia. eBeam - Interactive Whiteboard Technology, 2008.
6. Dana Tenneson and Sascha Becker. ChemPad: Generating 3D Molecules from 2D Sketches. In *SIGGRAPH '05: Posters*, 2005.