EUROGRAPHICS Workshop on Sketch-Based Interfaces and Modeling (2007)
M. van de Panne, E. Saund (Editors)

Designing a Sketch Recognition Front-End: User Perception of Interface Elements

Paul Wais, Aaron Wolin, and Christine Alvarado

Department of Computer Science, Harvey Mudd College, Claremont, CA
{pwais,awolin,alvarado}@cs.hmc.edu

Abstract

Programs that can recognize students' hand-drawn diagrams have the potential to revolutionize education by breaking down the barriers between diagram creation and simulation. Much recent work focuses on building robust recognition engines, but understanding how to support this new interaction paradigm from a user's perspective is an equally important and less well understood problem. We present a user study that investigates four critical sketch recognition user interface issues: how users integrate the process of triggering recognition into their work, when users prefer to indicate which portions of the diagram should be recognized, how users prefer to receive recognition feedback, and how users perceive recognition errors. We find that user preferences emphasize the importance of system reliability, the minimization of distractions, and the maximization of predictability.

Categories and Subject Descriptors (according to ACM CCS): H.5.2 [Information Interfaces and Presentation]: User Interfaces: *Evaluation/methodology, Interaction styles, Prototyping, User-centered design*

Categories and Subject Descriptors (according to ACM CCS): I.5.4 [Computing Methodologies]: Pattern Recognition: *Applications*

1. Introduction

Many engineering classes rely on simulation technologies to help students understand the systems they design. Unfortunately, mouse and keyboard interfaces to these programs are cumbersome. Students in these courses draw countless diagrams on paper (or on a Tablet PC) because sketch-based diagram creation is quicker and more natural. In fact, many instructors require students to draw diagrams on paper before entering designs into simulation software so that they focus on their design, not on the software interface.

Systems that can recognize and simulate students' hand-drawn sketches have the potential to lower the cognitive barrier between students and simulation software. However, these systems face their own interface challenges.

One challenge is how to allow users to trigger recognition and how to display recognition feedback. Feedback can be distracting in the early stages of design [HLLM02], but it can also aid recognition as it can help users adapt their drawing styles to match the system's expectations. Researchers have evaluated the usability of various recognition triggers and feedback mechanisms in isolation (e.g., [AD01, NLHL03, LaV06]), but have not compared different techniques directly.

A second challenge is how to allow users to indicate which pieces of the diagram the system should attempt to recognize. Students' homework often consists of a mix of text, equations, and diagrams. Despite advances in parsing heterogeneous notes [WSR06], a recognition system must receive only a single type of input to be practical. Most recognition systems allow the user to draw only one type of input (e.g., electrical circuits [GKS05]), while a few systems allow the user to manually select pieces of their drawing to be recognized after they have finished drawing [LaV06]. Again, little is known about which interface users prefer.

A third challenge is to minimize the impact of recognition errors on usability. Some systems reduce errors by placing constraints on users' drawing style. Others focus on intuitive error correction mechanisms as a way of reducing the impact of recognition errors [MHA00]. We believe that understand-

ing users' tolerance for different types of recognition errors can help guide sketch recognition research by allowing researchers to focus on eliminating errors that have the biggest impact on usability. Little work has been done in this area.

To address the above challenges, we present the first direct comparison of critical free-sketch recognition user interface (UI) elements. Unlike prior research, which usually involves a qualitative analysis of a complete solution, we compare interface elements directly using a novel application of a Wizard of Oz evaluation methodology. Specifically, we evaluate UI mechanisms for triggering recognition, providing feedback, and separating recognizable from unrecognized data, and we examine users' perception of different types of recognition errors. Our results indicate that:

- Users prefer to trigger recognition after they are done drawing, even when the system produces errors.
- Users prefer to segregate pieces of the drawing (e.g., diagram vs. annotation) at creation time rather than recognition time.
- Users want recognition feedback to transform and clutter their sketch as little as possible (even when they have completely finished sketching).
- Users prefer errors that are predictable.

One important user interface question that we do not address is how gesture-based or menu-based interfaces compare to free-sketch recognition interfaces. For example, users might prefer a reliable gesture-based system to an error-prone free-sketch recognition system. Although this question is important, we focus only on free-sketch recognition interfaces for two reasons. First, the students we talked to expressed a strong desire for a system that could simply transform diagrams they already produce for coursework into recognized circuit schematics; they did not want to have to learn a whole new language for interacting with the simulation software. Second, we cannot compare free-sketch recognition systems to gesture or menu-based systems until we better understand how to design an effective user interface for these systems.

2. Related Work

Previous user studies of sketch-based user interfaces focus mainly on the development or evaluation of complete systems. Researchers rely heavily on interviews and ethnographic studies to identify and understand user preferences of sketch-based computer tools. For example, Landay and Myers designed SILK [LM95], a sketch-based system for user interface design, based on the results of a survey of professional user interface developers. Newman *et al.* worked closely with designers throughout the design of DENIM [NLHL03], a sketch-based system for web page design. Their interaction with users revealed that web page development requires very little sketch recognition: DENIM uses gesture recognition techniques to recognize pages (rectangles) and links (arrows) but leaves the rest of the user's sketch unrecognized. Educational software, on the other hand, requires a deeper understanding of the user's sketch. Nevertheless, we rely on the model these studies provide for how to evaluate sketch-based user interfaces.

Evaluation of many recognition-intensive systems tends to focus on recognition error rates, and provides only general insight into user interface issues (e.g., "users found recognition errors frustrating") [AD01, GKS05, HD02]. A few researchers, however, have examined system usability in more detail. LaViola's evaluation of MathPad^2, a sketch-based system that recognizes freely-drawn equations and physical diagrams, reveals specific user preferences: users like MathPad^2's scribble erase gesture and find recognition errors frustrating but tolerable for this task [LaV06]. Other studies of recognition-based user interfaces are currently underway [KCR*07, Ten05]. None of these evaluations directly compare interface elements because of the difficulty in modifying complete recognition systems.

Other researchers have studied the usability of pen-based interaction techniques that are complementary to sketch recognition. The CrossY interface [AG04] explores several novel interface elements that combine gestures and traditional graphical user interface components. Long *et al.* [ACLLRM00] provide a model for measuring the visual similarity of gestures in an effort to inform effective gesture design. Finally Lank and Saund present a model of users' pen-based selection gestures to inform the design of a faster, more accurate selection mechanism [LS05]. The results of these previous studies are complementary to our results in the creation of a complete sketch recognition system.

Finally, though little is known about how free sketch recognition errors affect usability, researchers have studied user perception of handwriting and speech recognition errors. Rhyne and Wolf present early work in this area [RW93]. More recently, Frankish *et al.* find that the relationship between error rates and user acceptance is dependent on the perceived cost to benefit ratio of a specific task [FHM95]. We expect that the same trend holds for sketch recognition systems. Wu *et al.* find that handwriting task completion time is most sensitive to error rates above 6% [WZH03]. Munteanu *et al.* find a linear relationship between speech recognition accuracy and the quality of user experience with webcasts [MBP*06]. Although we do not examine error rates specifically, our analysis helps inform user perception of sketch recognition errors.

3. User Interface Elements

This section describes the interface elements and error types we compared. We chose a subset of elements used in existing sketch recognition applications and suggested by six pre-study participants that provides a restricted yet representative range of options.

We compared three different methods for triggering recognition: button, gesture and pause.

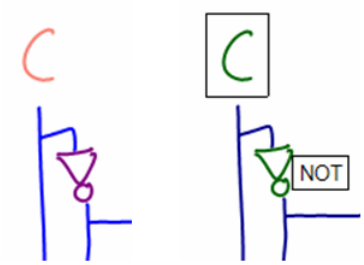


Figure 1: *Text and color feedback comparison.*

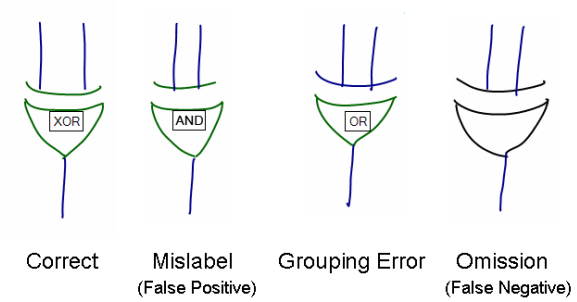


Figure 2: *Illustration of different error types.*

- **Button Trigger.** The user triggers recognition by tapping on a large interface button. This option is reminiscent of traditional recogniton systems.
- **Gesture Trigger.** The user triggers recognition using a "check-tap" gesture (a check mark followed by a dot). We selected this element because many systems posit the efficiency and naturalness of gestures over GUI widgets. Furthermore, this particular gesture seemed to be reliably recognized in informal tests, is not easily confused with with diagram-relevant symbols, and is similar to the "circle-tap" gesture used in MathPad^2 [LaV06].
- **Automatic Trigger.** The system triggers recognition automatically after a brief pause (4 seconds) in sketching. We chose a 4 second pause to accomodate different student work paces.

The second issue we explored was how to allow the user to indicate which strokes the system should recognize. It is possible to provide separate sketching panels for notes and diagrams, but when users want to make annotations directly on the diagram, this setup is infeasible. We examined two techniques for allowing users to segragate recognizable strokes from unrecognized strokes:

- **Pre-separation: "Color" Tool.** This version of the interface requires users to separate domain strokes from annotation strokes as they draw by toggling between stylus modes using a button on the GUI. The system uses ink color to indicate the current mode: in sketch mode, the stylus draws black ink; in annotation mode, the stylus draws gray ink (hence we call this option the "color tool").
- **Post-separation: Lasso Tool.** The user draws sketches and annotations freely, but then must lasso-select strokes to be recognized (similar to the interaction mechanism in MathPad^2 [LaV06]).

The third issue we explored is how to display recognition feedback. We compared two different methods: color feedback and text labels (Figure 1). We rejected the idea of replacing the user's strokes with symbols based on lack of common interest in this idea during pre-studies and the results of previous work [HLLM02].

- **Color Feedback.** The system displays each recognized symbol in a unique color. In addition, users can see a text label by hovering the stylus over the strokes in the symbol.
- **Text-Label Feedback.** The system draws text labels next to recognized symbols.

Finally, we investigated how three types of common recognition errors impact the user experience: false positives, false negatives and stroke grouping errors (Figure 2). False positives occur when the system incorrectly identifies a symbol (e.g., labels an AND gate as an OR gate); false negatives (or *omissions*) occur when the system fails to identify a symbol; and grouping errors occur when the system incorrectly adds or removes adjacent strokes to or from a symbol.

4. Experimental Design

For this study, we limited our scope to digital circuit design in order to keep tasks consistent across all users so as to better understand how interface decisions (as opposed to domain variability) affect usability. Although focusing on a single domain limits the generality of our results somewhat, the style of the tasks we explored is representative of structured educational design tasks in many fields, such as the sciences and engineering.

4.1. Tasks and Participants

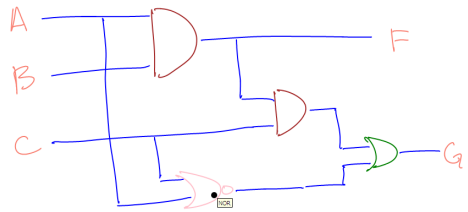
Users in our study designed circuits based on truth table function representations—a common task in an introductory digital design class. Figure 3(a) gives an example of one of these tasks. Users were allowed to use any method to create the circuit, and they were not required to simplify their circuit (although they could if they wanted to). In some cases (described in Section 4.3) we also asked uses to annotate the shortest and longest path through the circuit they designed.

We used a pre-study to design a set of uniformly difficult tasks. We asked participants to rate the difficulty of several tasks and selected for our study only those tasks that users rated as similarly difficult. Participants reported that these tasks were similar in difficulty to typical homework problems in an introductory digital design course.

Please draw a circuit diagram for the following truth table:

Input			Output	
A	B	C	F	G
0	0	0	0	1
0	0	1	0	0
0	1	0	0	1
0	1	1	0	0
1	0	0	0	0
1	0	1	0	0
1	1	0	1	0
1	1	1	1	1

(a) Truth table task



(b) A possible solution (with color feedback). The black dot denotes the user's stylus.

Figure 3: *A truth table task from our user study*

In total, nine people (six male and three female) participated in our formal tests (not including the pre-studies). All participants were Harvey Mudd College students who had taken an introductory digital circuit design class. All participants had used digital circuit simulation software (Xilinx) in their coursework. Six of these students had previous experience (more than an hour) with a Tablet PC, and five had taken notes on a Tablet PC during their digital design course (using Windows Journal or One Note).

4.2. Basic Interface Design

We first designed our prototype interface using iterative design techniques. Figure 4 shows a basic overview of our final interface, although we modified pieces of this interface to explore different interface elements. In this version, the user sketches (unrecognized) notes in the bottom panel, sketches the (recognized) circuit in the top panel and presses the "Recognize" button to trigger recognition.

Users could correct recognition errors only by erasing and redrawing their strokes. We did not aim to explore error correction mechanisms, and this method was adequate for users to complete their tasks. However, error correction mechanisms deserve attention from future studies as they are an important element of sketch recognition interfaces.

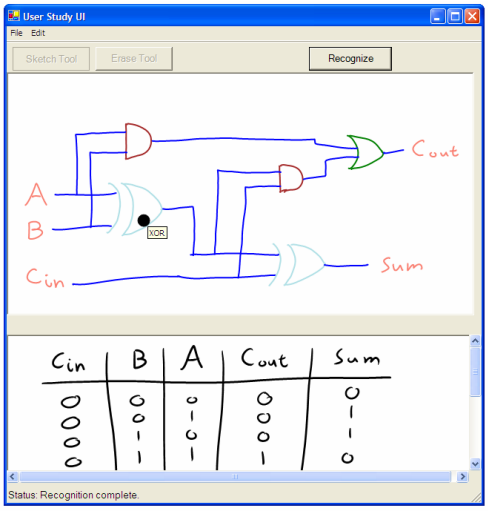


Figure 4: *Complete sketch recognition interface.*

4.3. User Studies

We conducted two separate studies to investigate the four interface elements described above. In the first study we investigated recognition triggers and diagram separation tools. In the second study we examined recognition feedback and error types. We divided our investigation into two studies in order to minimize the mental burden and scheduling commitment on users.

During both user studies we invited all participants to contribute informal feedback and asked them to fill out questionnaires inspired by the previous work of Chin et al. [CDN88] and Landay [Lan96]. Our questionnaires aim to measure user responses to individual interface elements based on relevant characteristics we inferred from our pre-study interviews and related work in sketch recognition user interfaces.

4.3.1. User Study #1: Recognition Triggers and Diagram Separation

Our first study examined triggers and diagram separation preferences across five participants. Each participant completed one truth table task with each recognition trigger using the multi-paneled interface depicted in Figure 4; the order of triggers tested was balanced across participants. Next, each participant completed one truth table task with each diagram separation tool in a single-paneled version of our interface. We prompted participants to label the shortest and longest path in their circuits and to have the system recognize only their diagrams (i.e., not the annotations).

After each task, the experimenter prompted users for qualitative feedback and gave questionnaires asking users to rank the reliability, efficiency, convenience and overall quality of the interface. After completing all tasks, users ranked

their preferred feedback and diagram separation mechanisms. Users completed study sessions in 30-45 minutes.

This study used two methods to "recognize" the user's diagrams and gestures. First, our system recognized users' check-tap gestures using the built-in Microsoft Tablet SDK gesture recognizer. Some users' gestures were recognized reliably, but others were not (we discuss the implications below). Second, the system simulated recognition of the users' diagrams by coloring most of the strokes blue, indicating "correct recognition," but displaying approximately 10% of the strokes in red, indicating a "recognition error." No recognition actually occurred, but most users expressed that they genuinely presumed the recognition results and simulated errors were real.

4.3.2. User Study #2: Feedback Mechanisms and Error Types

Our second study examined recognition feedback and recognition errors across six participants (including two participants from our first study). Each participant again completed one truth table task per feedback method and per error type, and we balanced the order of feedback mechanisms and error types, respectively, across users. (Feedback tasks always preceded error tasks.) When completing the error tasks, users chose the feedback mechanism they preferred. In all tasks, users triggered recognition with a button.

Again after each task, the experimenter prompted users for qualitative feedback and gave them questionnaires asking them to assess the reliability, efficiency, convenience and overall quality of the interface. After completing all tasks, users ranked which feedback mechanism they preferred and which errors were most confusing or difficult. Users completed study sessions in 45-75 minutes.

This study simulated diagram recognition through a novel application of a Wizard of Oz technique. Users worked with a realistic Tablet PC application while a human "Wizard" actively labeled user-drawn symbols. Wizard of Oz studies have proven effective for developing speech recognition interfaces [DJA93, KSC*00], but this is the first application of Wizard of Oz studies to the design of sketch recognition systems of which we are aware. Davis has developed a Wizard of Oz system to support sketch recognition user interface development [DSSL07], but this system was not ready in time for our study.

Our experimental Wizard of Oz system consists of two Tablet PCs directly networked over an Ethernet connection. As the user sketches on one Tablet, the Wizard actively receives copies of user strokes and labels relevant symbols on a second tablet. Once the user triggers recognition, the Wizard sends a labeled result to the user Tablet. For this stage of our study, our human subjects committee required us to inform participants that a human was recognizing their strokes.

The Wizard simulated perfect recognition while participants tested feedback mechanisms. To test error types, we simulated a 15% (approximate) error rate. We chose this rate because it is a realistic target for sketch recognition systems in the near future. We simulated each error type as follows:

- **False Positives.** The user end of our Wizard of Oz system filtered recognition results from the Wizard and randomly applied incorrect labels to approximately 15% of the recognition result.
- **False Negatives.** The system again filtered the Wizard's simulated results, randomly deleting the labels from approximately 15% of symbols.
- **Grouping Errors.** The Wizard incorrectly grouped the strokes in approximately 15% of symbols drawn, usually about 3 to 4 grouping errors per sketch.

5. Results and Discussion

In this section we present select quantitative and qualitative results of our study. Space constraints prohibit us from including complete user response data. In part due to our small sample size, even when users agreed, many of our survey responses do not show statistically significant differences. In many of these cases, however, qualitative feedback supports patterns in quantitative results. When users had conflicting opinions, we summarize these different beliefs so as to inform interface designers about the range of user preferences.

5.1. Student Workflow

Students unanimously reported that they prefer to design their circuits on a Tablet PC or whiteboard rather than enter them directly into a simulation tool. Furthermore, during our study, all of our participants used almost identical workflows. When designing a circuit, each student would write notes or equations, sketch a circuit, trigger recognition, and then correct recognition errors. Even when the experimenter reminded users that they could trigger recognition intermittently during the design process, participants continued to trigger recognition only after sketching a complete or significant portion of a circuit. User comments reveal two reasons for this workflow: users prefer to focus on the design and sketching portions of their tasks without interruption, and they prefer to correct recognition errors in a batch instead of individually.

5.2. Recognition Triggers

User reactions suggest that high reliability is the most important characteristic of a recognition trigger. Most users were able to sucessfully trigger recognition using check-tap only once for every 5 failures. Though users admit the gesture trigger offers ' a desirable convenience, the majority of users (n=3) ranked the button trigger higher than the other two triggers, and users unanimously rated it as highly reliable (Figure 5).

The button trigger's high reliability minimizes users'

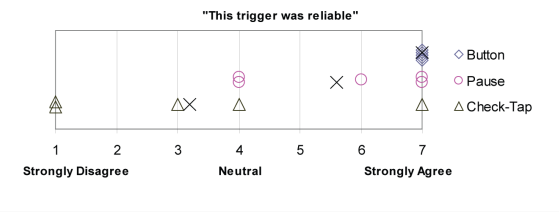


Figure 5: *Individual user responses to recognition trigger reliability. Each symbol represents a single user's response. "X"s show mean values.*

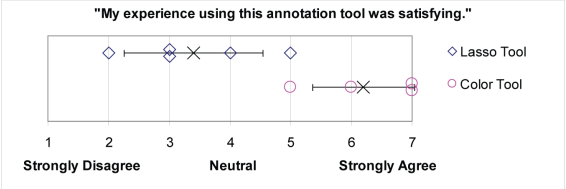


Figure 6: *Individual user responses to sketch separation method satisfaction. Error bars are shown for one standard deviation.*

mental effort and allows them to devise efficient workflows. One user commented that the button trigger helped her make her "approach more systematic" and "figure[] out [the] most efficient way to" use the interface. A second user commented that "[he could] always get faster with a tool that works every time."

The relatively high error rate of the check-tap gesture colored many users' reactions to this trigger. The one user that ranked the gesture trigger as the most desirable was able to trigger recognition successfully on his first six attempts. Even still, this user admitted to seeing only a "marginal difference" between the gesture and button triggers.

User reactions to the pause trigger suggest that students may find the pause trigger acceptable only if it matches the speed of their thought processes. Two users commented that a four second pause was too long, while one user found the pause trigger quite "distracting" to her mental flow and suggested a longer pause. The one user who ranked the pause trigger as his favorite trigger commented that it allowed him to "check as [he] move[d] through creating the diagram."

5.3. Separation/Annotation Tools

Users unanimously preferred the pre-separation (color) tool to the post-separation (lasso) tool, and all users found the pre-separation tool more satisfying to use than the post-separation tool (Figure 6). Many modal interfaces traditionally suffer from the problem that users often forget to toggle to the appropriate mode before performing a task. However, in our study, only one of five users forgot to toggle the color tool into annotation mode before writing notes; furthermore, this user corrected her mistake immediately.

Two participants commented that the lasso tool constrained the way they could create their diagrams. One user described that the color tool "required less planning about where and when to write so as not to screw up [the] lasso[] [gesture]." Another user commented that she "like[d] the [color tool's] ability to draw annotations wherever [she wanted]."

One issue that may have influenced user responses to the separation mechanism is that the lasso tool required two check-tap gestures while the color tool required only one. Nevertheless, users responded more negatively to the burden of circling the symbols and the physical segregating of notes and schematic strokes than to executing the gesture. One user even modified her questionnaire to express her preferences for annotation tools that hypothetically used the button trigger rather than check-tap; this user still preferred the color tool.

5.4. Feedback Mechanisms

Users unanimously agreed or strongly agreed that color feedback "helped them work efficiently" and "produced understandable results"; most users preferred color feedback to text-label feedback. Users found text feedback distracting (Figure 7) and that it added an unnecessary mental burden to the design process. One user commented that "The text gets very cluttered [and] covers up the [sketch's] points of interest quickly," thus interrupting his ability to reason about the diagram. Another user commented that text labels "can get a little distracting with four or five [instances] of the same gate" and finds the color feedback "more elegant" and "less redundant." Although our specific text placement may have influenced user perception of the text labeling feedback mechanism (i.e., sometimes the text obscured part of the sketch), automatic text placement is a difficult problem, and any automatic text placement method will likely obscure some portion of the user's sketch.

Despite the distracting nature of text feedback, color feedback alone is likely not informative enough. During the color feedback tests, each of the six participants immediately hovered her or his stylus over individual gates to verify the labels. In addition, one user commented that she "like[d] both" methods of feedback and found that "a button to go back and forth [between feedback mechanisms] would be nice."

5.5. Error Types

Users had diverse reactions to recognition errors, and our quantitative results show no strong trends. This lack of quantitative trends may be due in part to an unintended effect

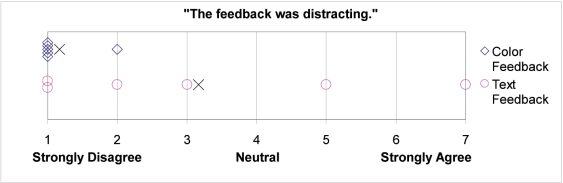


Figure 7: *Individual user responses to feedback distraction.*

of the way we generated errors. To keep error rates consistent, the computer system automatically generated omissions and false positives. However, generating grouping errors was too difficult to automate (it would require a deeper understanding of the sketch in order to split and merge adjacent groups), so the human Wizard generated grouping errors. Consequently, false positives and omissions were often more surprising to users (e.g. a wire classified as an AND gate) than grouping errors. This effect limits the direct comparison we can make between user responses to error type, but allows us to better understand the importance of how users perceive errors based on how well they understand them.

Our qualitative results suggest that user acceptance of the system is related not only to absolute error rates but also to how well they understand and trust the system's recognition. Generally, our users were initially happy to correct any type of recognition error, but quickly became frustrated when they could not understand why errors occurred. Two users specifically explained that their acceptance of the system would depend not on the error rate as much as on the predictability of errors. One of these users remarked that he would not be willing to accept a system with the error rate exhibited unless errors were more predictable and helped him adapt his drawing style to avoid them. This user also found false negatives the most confusing because these errors failed to inform him about "what to move away from" (i.e., how to change his sketching style to reduce errors). On the other hand, another user preferred false negatives because it made him trust the system more when it did recognize his strokes. Though related work suggests that user acceptance may improve with lower absolute error rates, our results illustrate the importance of user perception of error type to acceptance of the system. Future work may seek to further compare the impacts of error type and error rate on user experience.

We also found that users created dramatically different mental models of errors. Two users considered themselves responsible for grouping errors. Two users blamed the system for mislabel and omission errors. One of these users remarked that mislabel and omission errors seemed "out of context," or unpredictable. Two users specifically remarked that they did not try to form a mental model of errors.

Qualitative user responses related to frustration with errors exhibited a general accordance. Four out of six users responded that omission errors were the easiest to perceive; users likely find that the lack of a label is easier to detect than an incorrect label. Furthermore, four out of six users remarked that the grouping error was the most confusing type of error despite the fact that these errors were human-generated.

6. Conclusion

The results of our study inform the design of educational sketch recognition systems. Here we summarize the major implications for both user interface and recognition engine developers.

Recognition triggers should be user-triggered, efficient, and reliable. Our study illuminates that, with respect to our tasks, users prefer reliability as much as they do convenience. Gestures and system activated recognition may be an option, but should not be the only option.

The interface should provide users with a way to separate recognized from unrecognized strokes as they draw. Users desire efficiency and are willing to put in small amounts of effort while they work to avoid larger amounts of effort later in the process. Our study participants preferred our pre-separation tool for its relatively low mental and physical overhead.

Recognition feedback should provide minimal clutter and transform users' strokes as little as possible. Stroke color effectively provides contrast between recognition labels with minimal stroke transformation. More dramatic transformation should occur only upon user request.

Users are willing to correct errors after they are done drawing. Our users treated correction as a game or a necessary evil. Users do error correction all in one batch and tend not want to disturb their design process with on-the-fly correction (this result is consistent with [HLLM02]).

Errors must be predictable and/or understandable. Automatic recognition engines can potentially make nonsensical errors. Recognition engines that incorporate adjustable confidence levels may help user interface designers strike an optimal balance between false positives and false negatives. A recognition engine that could describe precisely why a given interpretation was missed could also help users modify their drawing styles to raise recognition rates.

This study also suggests other important questions to explore. Future studies should examine reaction to error rate, type and correction mechanism in more detail. In addition, although some of our users expressed a moderate willingness to modify their drawing style to reduce recognition error, more work is needed to determine the optimal balance between a gesture-based and free-sketch interface.

Perhaps the most important outcome of our user study was

the excitement users showed for a sketch-based user interface to replace the cumbersome interface they currently use. The major advantage they perceived in a sketch-based interface is the lower cognitive load in entering their circuits, and they expressed a willingness to cope with and correct the recognition errors inherent with such a system.

7. Acknowledgments

We would like to thank our study participants, Kris Karr (our Wizard), and Susan Matonosi and Deb Mashek who provided helpful feedback on our study design. This work is supported in part by an NSF CAREER award (IIS-0546809).

References

[ACLLRM00] A. CHRIS LONG J., LANDAY J. A., ROWE L. A., MICHIELS J.: Visual similarity of pen gestures. In *Proc. of CHI* (2000), ACM Press, pp. 360–367.

[AD01] ALVARADO C., DAVIS R.: Preserving the freedom of paper in a computer-based sketch tool. In *Human Computer Interaction International Proceedings* (2001), pp. 687–691.

[AG04] APITZ G., GUIMBRETIÈRE F.: CrossY: a crossing-based drawing application. In *Proc. of UIST* (2004), ACM Press, pp. 3–12.

[CDN88] CHIN J. P., DIEHL V. A., NORMAN K. L.: Development of an instrument measuring user satisfaction of the human-computer interface. In *Proc. of CHI* (1988), ACM Press, pp. 213–218.

[DJA93] DAHLBÄCK N., JÖNSSON A., AHRENBERG L.: Wizard of oz studies: why and how. In *Proc. of IUI* (1993), ACM Press, pp. 193–200.

[DSSL07] DAVIS R. C., SAPONAS T. S., SHILMAN M., LANDAY J. A.: SketchWizard: Wizard of Oz prototyping of pen-based user interfaces. In *Proc. of UIST* (2007).

[FHM95] FRANKISH C., HULL R., MORGAN P.: Recognition accuracy and user acceptance of pen interfaces. In *Proc. of CHI* (1995), ACM Press/Addison-Wesley Publishing Co., pp. 503–510.

[GKS05] GENNARI L., KARA L. B., STAHOVICH T. F.: Combining geometry and domain knowledge to interpret hand-drawn diagrams. *Computers and Graphics 29*, 4 (2005), 547–562.

[HD02] HAMMOND T., DAVIS R.: Tahuti: A geometrical sketch recognition system for uml class diagrams. In *AAAI Spring Symposium on Sketch Understanding* (2002), AAAI Press, pp. 59–68.

[HLLM02] HONG J., LANDAY J., LONG A. C., MANKOFF J.: Sketch recognizers from the end-user's, the designer's, and the programmer's perspective. In *Sketch Understanding, Papers from the 2002 AAAI Spring Symposium* (2002), pp. 73–77.

[KCR*07] KOILE K., CHEVALIER K., RBEIZ M., ROGAL A., SINGER D., SORENSEN J., SMITH A., TAY K., WU. K.: Supporting feedback and assessment of digital ink answers to in-class exercises. In *Conference on Innovative Applications of AI (In submission)* (2007).

[KSC*00] KLEMMER S. R., SINHA A. K., CHEN J., LANDAY J. A., ABOOBAKER N., WANG A.: Suede: a wizard of oz prototyping tool for speech user interfaces. In *Proc. of UIST* (2000), ACM Press, pp. 1–10.

[Lan96] LANDAY J. A.: *Interactive Sketching for the Early Stages of User Interface Design*. PhD thesis, Carnegie Mellon University, 1996.

[LaV06] LAVIOLA J. J.: An initial evaluation of a pen-based tool for creating dynamic mathematical illustrations. In *Proc. of Eurographics Sketch-Based Interfaces and Modeling* (2006), pp. 157–164.

[LM95] LANDAY J. A., MYERS B. A.: Interactive sketching for the early stages of user interface design. In *Proc of CHI* (1995), pp. 43–50.

[LS05] LANK E., SAUND E.: Sloppy selection: Providing an accurate interpretation of imprecise stylus selection gestures. *Computers and Graphics 29*, 4 (2005), 490–500.

[MBP*06] MUNTEANU C., BAECKER R., PENN G., TOMS E., JAMES D.: The effect of speech recognition accuracy rates on the usefulness and usability of webcast archives. In *Proc. of CHI* (2006), ACM Press, pp. 493–502.

[MHA00] MANKOFF J., HUDSON S. E., ABOWD G. D.: Providing integrated toolkit-level support for ambiguity in recognition-based interfaces. In *Proc. of CHI* (2000), pp. 368–375.

[NLHL03] NEWMAN M. W., LIN J., HONG J. I., LANDAY J. A.: DENIM: An informal web site design tool inspired by observations of practice. *Human-Computer Interaction 18*, 3 (2003), 259–324.

[RW93] RHYNE J. A., WOLF C.: Recognition based user interfaces. *Advances in Human-Computer Interaction* (1993), 191–250.

[Ten05] TENNESON D.: *Technical report on the design and algorithms of ChemPad*. Tech. rep., Brown University, 2005.

[WSR06] WANG X., SHILMAN M., RAGHUPATHY S.: Parsing ink annotations on heterogeneous documents. In *Eurographics Workshop on Sketch-Based Interfaces and Modeling (SBIM)* (2006), pp. 43–50.

[WZH03] WU C., ZHANG K., HU Y.: Human performance modeling in temporary segmentation chinese character handwriting recognizers. *Int. J. Hum.-Comput. Stud. 58*, 4 (2003), 483–508.