Human Computer Interaction
International Proceedings, pp. 687–691,
2001.

Preserving the Freedom of Paper in a Computer-Based Sketch Tool

Christine J. Alvarado and Randall Davis

MIT Artificial Intelligence Laboratory
{calvarad, davis}@ai.mit.edu

Abstract. The tradeoff between the ease of sketching a mechanical system on paper and the power of representing it on a computer is too great, as evidenced by the fact that mechanical designers typically begin by sketching their designs on paper, then transfer them to the computer only after the initial design process is complete. To bring early design to the computer, we must develop a tool that allows the user to sketch freely, yet still interprets and understands the user's drawing as a mechanical system. We present a tool that allows users to sketch simple 2-D mechanisms and report on early reactions to it.

1 Introduction

User interfaces to current computer aided design (CAD) and simulation tools offer little assistance at the conceptual stage of design. As a result, designers typically begin by sketching with paper and pencil and transfer their designs to the computer only when nearly complete.

Our goal is to eliminate this two-stage design process by creating a computer-based sketching tool that is both powerful and natural. The tool should allow the user to sketch freely, without modifying her drawing style, yet still be able to interpret the user's drawing and allow the user to interact with it, in a manner illustrated below.

Free sketching is not the only route to ease of interaction with design tools. As one example, sketch interpretation can be simplified by constraining the user's drawing style, for example by requiring that the user draw each object with a single stroke [9]. [8] explores another approach, creating shapes by carving larger shapes with a carving tool rather than sketching them directly.

We chose instead to allow users to sketch as they would on paper, attempting to alter their style as little as possible. There is an interesting, open question of whether free sketching is indeed the best overall interaction metaphor, but detailed comparisons of sketching vs. menus (or other approaches) cannot be addressed until an adequately powerful sketch system has been built. This work is a first step in building such a system.

Our tool, called ASSIST (A Shrewd Sketch Interpretation and Simulation Tool), enables sketching simple two-dimensional mechanical systems in a natural fashion, i.e., without explicitly informing the system what is being drawn. ASSIST interprets the sketch as the user draws and can simulate the design at any time during the design process. This paper describes the tool, discusses a user study that evaluated its utility in its current form, and suggests directions for future work and improvements.

We begin with a description of a typical interaction with ASSIST, then introduce our model for sketch interpretation and interaction, including a brief discussion of our recognition algorithm. This is followed by a discussion of the user test we performed, the feedback we gathered, and the implications for future design. We conclude with a discussion of related and future work.

2 Interacting with assist

Users interact with the system through a digitizing LCD tablet that allows them to draw directly onto the computer screen and see their strokes appear under their pen as they draw them.

We describe a session in which a user draws the car on a hill seen in Figure 1. The user begins by drawing one side of the polygon that constitutes the car body on an otherwise blank canvas. As she is drawing, she sees her stroke appear under her pen in grey. As she lifts her pen, the system replaces her stroke with a black line to indicate its recognition. At this point the system believes that the user's stroke could possibly be a rod[1] or a line that will become part of some other part (in this case a mechanical body). Next she draws the rest of the body with one stroke. Again, she sees her strokes in gray as she draws; when she lifts her pen the system replaces all of her strokes with a blue polygon that closely matches the strokes she has drawn (Figure 1, center). Note that the system has "cleaned up" her raw strokes to indicate its recog-

[1] A rod is a thin rigid body.

nition. We discuss the advantages and disadvantages of this behavior in Section 4.

The user continues to draw the other pieces of the drawing. By mistake, she draws one of the wheels too large. To delete the wheel, she circles it; the wheel turns red to indicate it has been selected. She then draws a line through it to delete it.

After she finishes the drawing, she draws the hill for the car to sit on, and anchors it to the background by drawing an "X"[2]. At this point the car is floating above the hill. To move it, she circles it to select it, and then drags it down to rest on the hill by putting her pen on one of the selected pieces and dragging it downward until it rests on the hill. As she drags her pen, the whole car moves with it.

If at any point in the drawing process the system misinterprets one of the user's strokes, she can tap the "Try Again" button at the bottom of the screen, and ASSIST will present a list of alternative interpretations for her stroke.

Finally, when she finishes her drawing (or at any time during the design process), she can tap the "Run" button to see a simulation of the system: another window appears (Figure 1 right), showing the car running down the hill. The original drawing remains unchanged, allowing for a convenient sketch-modify-simulate loop.

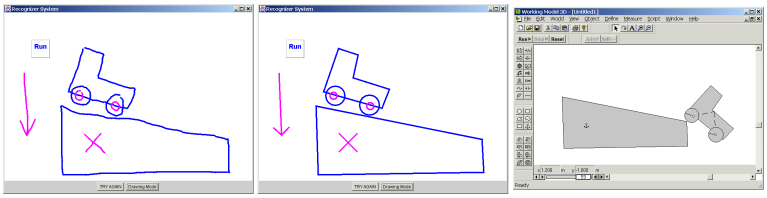

Fig. 1. A car on a hill: As drawn by the user (left), as interpreted and displayed by ASSIST (center) and as simulated in Working Model (right).

3 Approach

3.1 Recognition

Although our recognition algorithm is not the focus of this paper, understanding it will help explain some of our interface design decisions. For a more complete description of the algorithm, see [1].

[2] Anything not anchored can fall.

The first step to recognizing a user's sketch is to identify the patterns in the sketch that represent mechanical parts. To identify these patterns without forcing the user to significantly change her drawing style, ASSIST relies on the fact that mechanical engineering has a fairly concrete visual vocabulary for representing components [2].

Simply recognizing these patterns is not enough, however, because there is a great deal of ambiguity in any mechanical sketch. For example, consider wheels of the car in Figure 1. Are the small circles in the middle of the large circles pin joints or circular bodies themselves? The recognition process must also choose the correct representation for a set of strokes, given multiple patterns to choose from.

After each stroke, ASSIST uses a three-stage procedure to choose the most likely interpretation for that stroke. First, it matches the user's strokes (including the most recently drawn) to a series of templates, producing all possible interpretations of the strokes. Next, the system ranks the interpretations using a series of heuristics (described in [1]) about drawing style and mechanical engineering. As one example, consider one of the the small circles that represent the pin joints in Figure 1. The system was able to recognize it as a pin joint instead of a circular body because in two-dimensional systems bodies do not typically overlap without some means of interconnection (i.e. the pin joint). Finally, the system chooses the best consistent overall set of interpretations from among those produced in step 1, and displays it to the user.

3.2 Level of aggressiveness

One design issue we faced was how "aggressive" the system should be in interpreting the sketch. Interpreting after every stroke keeps the user informed of the system's current understanding, allowing her to intervene promptly to correct any misinterpretations[3].

We could alternatively have chosen to delay interpretation, interpreting only after the user had finished drawing, or at least paused substantially. This approach has the advantage that the user would not have to deal with the system's interpretations after every stroke. But errors are more difficult to correct: The user may have to go back and fix interpretation errors after finishing part of the drawing, as an early mistake can lead subsequent interpretations down the wrong path.

[3] Lack of intervention is, conversely, taken as tacit approval: the system becomes more sure of an interpretation the longer it remains unchallenged by the user.

We decided to build the system with a constant interaction between the system and the user, to keep the user's expectations from becoming too far removed from the system's interpretations. This interaction also offers guidance that keeps the system from getting lost in what would otherwise be a very large space of possible interpretations, as the drawing becomes more complex[4].

4 Evaluation and results

To judge how natural the system is we ran a test in which subjects were asked to sketch three simple mechanical systems, both on paper and using ASSIST via an active matrix LCD tablet. We observed their behavior and asked them to explain their experience with the system, describing in what ways it felt natural and in what ways it felt awkward. Since this is an early-stage design, we were looking for qualitative feedback not quantitative results. We use the feedback to guide the system into its next implementation.

4.1 Method

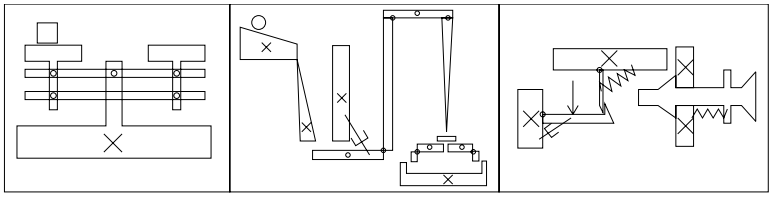

Fig. 2. Three test examples: a scale (left), a Rube-Goldberg machine (center) and a circuit breaker (right).

Eleven subjects from the MIT AI Lab took part in the study. All of the subjects had a background in computer science; two were mechanical engineers.

Sessions with the subjects lasted between 30 minutes and an hour and were videotaped.

[4] Results from our study (Section 4), however, indicate that it might be more natural for the system to display its interpretation less often, even at the cost of a few more mistakes.

We first showed them the diagrams in Figure 2 and told them they would be asked to drawn them twice, once on paper and once using our system. They were told not to worry about replicating the details exactly. After they drew the figures on paper, we gave them a 5 minute introduction to ASSIST that included drawing simple parts, such as a pair of bodies connected by a spring. We also told them how to select, move and delete objects. We then asked them to use ASSIST to draw the same systems they had drawn on paper.

After they were done sketching we asked them a series of questions (Table 1). Although we were observing their behavior as they sketched, we also wanted them to articulate what they felt was particularly intrusive or particularly helpful in working in ASSIST. The first few questions measure the perceived number of mistakes the system made and how tolerant users were of these mistakes; the last two are open ended.

1. How often did you feel the system got the correct interpretation for your strokes?
2. When the system misinterpreted your stroke, how reasonable was its misinterpretation?
3. How clear was the system's interpretation for your strokes?
4. How easy was it to modify the drawing?
5. Compare using this system to drawing on paper.
6. Compare using this system to using a menu-based interface.

Table 1. Questions we asked the subjects.

4.2 Results

Our first observation was that the learning curve for our system is low. All subjects were able to successfully draw all of the systems in Figure 2. While they had to modify slightly the way they drew certain objects, such as polygons and springs, for the most part they were either able to use their natural drawing style or quickly learn the small changes required to work successfully with ASSIST[5].

Because of the slight modifications to their drawing styles, it took subjects slightly longer to draw system with ASSIST than it did on paper. The three examples took the subjects about 10 minutes total to sketch on paper (approximately 2 minutes for the scale, 4 minutes

[5] Since this study, another member of our group has implemented more robust low-level recognizers to be more accurate with less-constrained drawing styles. (See [10].)

for the Rube-Goldberg machine, and 4 minutes for the circuit breaker). The scale took subjects about 5 minutes to draw using ASSIST, while subjects spent about 10 minutes on the Rube-Goldberg machine and about 10 minutes on the circuit breaker.

In response to question 1, almost all subjects reported the that the system got the correct interpretation of their strokes between 70 and 80 percent of the time. One subject reported that the system got the correct interpretation for his strokes only 50 percent of the time, while another said he felt the system interpreted his strokes correctly about 95 percent of the time. Most recognition errors were polygon recognition errors.

When the system did make a mistake, subjects overwhelmingly preferred to delete the mistaken interpretation and redraw it, rather than using the "Try Again" button. A few subjects tried using "Try Again", but did not find the correct interpretation of their piece and ended up deleting the piece they had just drawn anyway. The bias against the "Try Again" button is likely due to the fact that when the system misinterprets a user's stroke, it is often a result of never identifying the correct interpretation for that stroke, in which case the correct interpretation does not appear in the "Try Again" list. It may also be the case that our users are too used to paper, where re-drawing is the common reaction to a misunderstanding.

All subjects reported that it was clear when the system had made a mistake that they needed to correct, but also reported that the feedback was at times distracting. The primary source of distraction was the way the system replaced the user's strokes with the icon indicating its interpretation. Even though the icon is fitted to match the user's stroke as closely as possible, the match is never exact, so the stroke appears to "jump" a little when it is replaced. This behavior bothered some people because they felt that they no longer had total control of the strokes they were putting on the page and it was therefore difficult to have any sort of precision in their drawing. In our current implementation we have removed the replacement of the user's strokes and instead use a color change to indicate interpretation. Further investigation is necessary to determine whether the color change gives adequate feedback.

People were quite pleased with the power that the system brought to the drawing process in the realm of mechanical engineering. Subjects found the ability to simulate the drawings to be quite appealing. It is especially interesting to note that subjects were intrigued when the behavior of their system did *not* match the intended behavior. For example, unless drawn with great attention to symmetry, the scale will

not balance when empty (by default all bodies have the same mass so the scale will tilt toward the side with the longer lever arm). Subjects often tweaked their drawings until they got the desired behavior.

People also liked the idea that the system could be a more complete editing tool. For example, although paper does not give users the ability to cut and paste or rotate their images, most people said that having these features in the program would not make the interface feel any less natural, but would simply add power they have come to expect from a computer program. Also, people liked the idea that ASSIST was able to clean up their drawings, even though they would have preferred for this cleanup to take place after they were done drawing. Several subjects expressed interest in a more controlled way to clean up their drawings. They wanted their strokes to be left rough at first, with the option to go back and touch up the drawing later.

5 Related work

Other sketch-based design tools include the Electronic Cocktail Napkin [3, 4] and SILK (Sketching Interfaces Like Krazy) [5]. Our work explores many similar issues as theirs, but in the domain of mechanical engineering where the task of recognition involves more ambiguity resolution.

Our work deals with many of the issues presented in [6]. Their work acknowledges the idea that in any naturally based interface will have inherent ambiguity in the input. She presents a method for representing these ambiguities and then applying a series of "mediators" to the possible recognitions. Our approach is similar; the second step of our recognition algorithm is similar to the "mediation" stage in her architecture.

6 Future work and conclusions

We are exploring our free sketch interface further by incorporating the feedback received from users into the current version of our interface. We have been working to improve low-level recognition algorithms because most of the interpretation mistakes made by the system were low-level mistakes. We have changed our feedback on the interpretation of the design to simply change the color of the strokes, rather than replace the strokes themselves.

We would also like to produce more intuitive simulations for the mechanical systems sketched by users. We noted that users tweaked their

drawings until the simulator gave them the desired behavior. In many cases they were confused as the why the simulator did not produce the correct behavior, not understanding that a literal interpretation of a hand-drawn sketch rarely produces the desired qualitative simulation of the system. Other work in our group has developed methods that allow users to describe the intended behavior of sketched devices through spoken explanations and sketched gestures [7]. Our goal is to have ASSIST use the behavioral descriptions to adjust parameters such that the simulation gives the desired qualitative behavior.

We have suggested that engineers currently do not use computers in early design because they do not allow for quick and natural sketching of ideas. To be useful in early design, computers must allow the designer to sketch as on paper, and provide benefits, such as the ability to simulate the system, that are not available with paper. We believe the work described here takes several small but important steps in that direction.

References

1. Christine Alvarado and Randall Davis. Resolving ambiguities to create a natural sketch based interface. In *Proceeding of IJCAI-2001*, August 2001.
2. Ivan I. Artobolevsky. *Mechanisms in Modern Engineering Design*, volume 1. MIR Publishers, Moscow, 1976.
3. Ellen Yi-Luen Do and Mark D. Gross. Drawing as a means to design reasoning. *AI and Design*, 1996.
4. Mark D. Gross. The electronic cocktail napkin - a computational environment for working with design diagrams. *Design Studies*, 17:53–69, 1996.
5. James A. Landay and Brad A. Myers. Sketching interfaces: Toward more human interface design. *IEEE Computer*, 34(3):56–64, March 2001.
6. Jennifer Mankoff, Scott E Hudson, and Grefory D. Abowd. Providing intergrated toolkit-level support for ambiguity in recogntiion-based interfaces. In *Proceedings of the CHI 2000 conference on Human factors in computing systems*, pages 368–375, 2000.
7. Michael Oltmans. Understanding natually conveyed explanations of device behavior. Master's thesis, Massachusetts Institute of Technology, 2000.
8. Roope Raisamo. An alternative way of drawing. In *Proceedings of CHI 1999*, pages 175–182, 1999.
9. Dean Rubine. Specifying gestures by example. *Computer Graphics*, pages 329–337, July 1991.
10. Metin Sezgin. Early processing in sketch understanding. Unpublished Master's Thesis, Massachusetts Institute of Technology.