Enabling Collaboration in the Sketching Domain

by

Jesse Michael Smithnosky

Submitted to the Department of Electrical Engineering and Computer Science on May 19, 2005, in partial fulfillment of the requirements for the degree of Master of Engineering in Electrical Engineering and Computer Science

Abstract

Sketching, although deceptively simple and seemingly primitive, is a powerful paradigm for designing and understanding many types of engineering systems. Many problem domains, such as designing electrical circuits, developing flow charts, and modeling simple mechanical devices, rely heavily on the ability to produce sketches efficiently in order to bring out the most salient features.

Engineers working in these domains usually rely on pen and paper to generate their design sketches. They do this because more advanced technologies (such as notebook computers) are often unavailable, hard to learn, or cumbersome. It is important for engineers to collaborate with their colleagues while working on their sketches. Unfortunately, collaboration on sketches that exist only as pen and paper often proves to be tedious, requiring a minimum of a fax machine and scanner. Engineers could benefit from a more efficient means of collaboration when dealing with pen and paper sketches. The technology exists to improve the current situation and make pen and paper sketches a more effective medium for collaborative design.

This thesis presents an implementation of a system that achieves three goals. First, the system allows two users to collaborate on the production of a sketch in much the same way they would collaborate when composing a document (with one user composing a sketch, then accepting or rejecting the changes of his collaborator). Second, it allows users to watch a collaborator’s additions play in real time, like watching a movie. And finally, it links the sketch recognition and simulation software developed by the Design Rationale Group at MIT with a simple pen and paper interface, allowing engineers to run simulations of their design sketches. These goals are achieved by using a commercial pen produced by the Anoto Group that is capable of storing the strokes it draws. In essence, the user creates both a hard and soft copy of the sketch simultaneously, and can share the soft copy with any collaborator. Using this model of production, sketches can be collaboratively generated, edited, and reviewed quickly and easily, all using only a pen, paper, and a standard printer.

Thesis Supervisor: Randall Davis
Title: Professor of Computer Science
Acknowledgements

First and foremost, I owe a debt of gratitude to my advisor, Randall Davis. Without his guidance, creative thought, and confidence in me, this project would surely have failed. I also must thank Professor Patrick Henry Winston for introducing me to the field of Artificial Intelligence. Along these lines I must thank the other members of the Design Rational Group whose knowledge, expertise, and support were invaluable throughout the course of this work. Also, without the help of Ken Overgard and Scott Blackwell, both engineers at HP, this project could not have succeeded.

I would be greatly remiss if I did not acknowledge the help and guidance of several exceptional teachers at Ringgold High School. Without them, it is unlikely that I would even have applied to MIT, let alone been accepted and matriculated. Patricia Zober, Barry Harris, and Linda Fetchen all helped to shape me academically and to prepare me for the rigors of MIT, and I will never forget it.

I also must thank the countless friends and other people who have made my MIT experience so much better. Whether they helped me learn the material for a specific class, suggested research topics, or simply were there for me during a rough patch of life, without them the work that culminated in this project would have been extremely difficult, if not impossible, to complete.

Finally, I need to whole-heartedly thank my parents. Without their unwavering financial and emotional support, none of this would have been possible. They have always provided me with the best opportunities I could ever hope for, and allowed me to always shoot for my dreams. Their hard work and dedication have enabled me to make a wonderful life for myself, and for that I am truly grateful.
Contents

1 Introduction 9

1.1 Overview ................................................................. 9
1.2 Implementation .................................................... 11
1.3 Preview ................................................................. 12

2 Background 13

2.1 The Design Rationale Group and Assist ....................... 13
2.2 Collaboration in the Design Process ............................. 14
2.3 Pen Computing ........................................................ 16

3 System Components 18

3.1 Anoto Group io2 Digital Pen ....................................... 18
3.2 Digital Paper .......................................................... 19
3.3 Pen Download Software ............................................. 20
3.4 Service Application .................................................. 20
3.5 Anoto Group Software Development Kit...................... 21
3.6 Sketch Builder ......................................................... 21
3.7 Assist Stroke Recognizer and Working Model........ 21

4 Sketch Builder 23

4.1 Direct Link to Working Model 2D......................... 23
4.2 Accepting and Rejecting Changes....................... 28
4.3 Temporal Selection....................................... 31
4.4 Viewing and Analyzing Sketches.......................... 33

5 A Working Example 36

5.1 The First User.............................................. 36
5.2 The Second User.......................................... 37
5.3 The Final Product........................................... 37

6 Discussion 39

6.1 Cross-medium Analysis..................................... 39

7 Contributions 41

8 References 47
List of Figures

Figure 4.1.1  An example of a user’s sketch as seen in Sketch Builder, displayed automatically after docking the io2 Digital Pen. 24

Figure 4.1.2  An example of a user’s sketch as seen in Assist, displayed automatically after docking the io2 Digital Pen. In this example Assist has successfully recognized all the components in the sketch. 25

Figure 4.1.3  An example of a user’s sketch as seen in Working Model 2D, displayed automatically after docking the io2 Digital Pen. This figure shows the simulation in its initial conditions. 26

Figure 4.1.4  A still image of a simulation of a user’s sketch in Working Model 2D, after the simulation has begun. 27

Figure 4.1.5  A still image of a simulation of a user’s sketch in Working Model 2D, after the simulation has completed. 27

Figure 4.2.1  A sketch that has been annotated by a second user. The original author’s strokes appear in black, while the collaborator’s strokes appear in green. 29
Figure 4.2.2 A sketch with selected strokes displayed in red. Users can select a stroke by either clicking it individually or dragging the mouse with the left button depressed to create a rectangle that encompasses the entire stroke. 30

Figure 4.2.3 A final version of the sketch, with additions and deletion selected. 31

Figure 4.3.1 A sketch on which temporal selection has been performed using the sliders at the bottom of Sketch Builder. Temporal selection also be performed while the user watches a movie of the sketch being recreated. 32

Figure 4.4.1 A zoomed-in view of the arrow shown in previous figures. 34

Figure 4.4.2 A zoomed-in view of the tip of the arrow shown in Figure 4.4.1. In this view, the user can see the individual data points that comprise the sketch. The tip of the arrow is very noisy, contributing to the difficulties inherent in sketch recognition. 35
Preface

Sketching is a powerful problem-solving paradigm in virtually all fields and aspects of engineering. Mechanical engineers draw levers and gears, electrical engineers draw digital circuits, and project managers draw flow charts, just to name a few examples. New technologies, such as notebook computers and other touch screen displays, are being developed and may someday replace pen and paper sketches. However, these technologies are far from pervasive and still have a good way to go before they could possibly supplant pen and paper. Even when such technologies do reach maturity, it is probable that many engineers will still prefer pen and paper, as it is the medium with which they are most comfortable. Thus, it is safe to say that pen and paper sketches won’t be disappearing from design meetings any time soon.

While work continues on enabling informal sketching in electronic media (such as the sketch recognition algorithms being developed by the Design Rationale Group at MIT), I believe that it is important to develop technologies and a model of interaction that will increase the effectiveness of sketching using pen and paper. The system presented in this thesis allows engineers to collaborate on designs produced using only pen and paper, as well as to link their sketches to powerful recognition and simulation software that is already in place. By achieving these two things, the work presented here takes a step toward increasing the usefulness of hand-drawn sketches in the design process, allowing engineers to focus more on the specific problems of their domain and less on the tools with which they hope to solve them.
Chapter 1

Introduction

1.1 Overview

I believe that improving the capacity for collaboration in the domain of pen and paper sketches is an important task that will aid designers and engineers in many fields. Without a means for efficient collaboration, engineers cannot take full advantage of the pen and paper sketches they produce. A sketch in a notebook or on the back of an envelope can be neither run as a simulation nor edited easily by a colleague at another location. This thesis aims to ameliorate this difficulty. The following sequence of events describes one way in which the necessary collaboration and simulation could take place:

1. Andy creates a sketch of a system in his notebook. He keeps his paper copy, and also has an electronic copy in a form that is easy to view and edit on a stroke by stroke basis using a simple piece of software.

2. Andy processes his sketch with recognition software and runs a simulation of his system, using the same piece of software.
3. Once he is satisfied with the simulation of his system, Andy emails the electronic copy of his sketch to Betty, who uses the same piece of software to view and print the sketch.

4. Betty prints out a copy of the sketch and annotates it. She keeps her paper copy, and also has electronic versions that correspond to the original sketch and her annotated version. She processes the annotated version with a sketch recognizer and runs it in a simulation.

5. When she is satisfied with the simulated behavior of her system, Betty emails the electronic copy of the annotated sketch back to Andy, who opens it using the same piece of software.

6. Andy decides which of Betty's annotations to keep and which to reject, in much the same way a user of Microsoft Word decides to keep or reject changes another user has made to a document. He also adds new strokes where appropriate.

7. Andy runs the sketch recognizer and simulates his system again, hopefully with improved results.

8. The process repeats, if necessary, with Andy emailing the latest version of the sketch back to Betty again for further review.

This sequence is effective and efficient when compared to most alternative approaches (such as using a fax machine and scanner, in conjunction with other software to enable the recognition and simulation). A computing environment that allows this sequence to take place could enhance the effectiveness of pen and paper sketches in many engineering
domains, as it allows engineers to better capitalize on the thought and effort they have put into their pen and paper sketches.

1.2 Implementation

This thesis presents an implementation that allows the sequence described in Section 1.1 to take place, by combining a commercial product from the Anoto Group (the io2 Digital Pen) with a Java graphical user interface I have created called Sketch Builder and existing software developed by the Design Rationale Group at MIT. A user of the system can draw with the io2 Digital Pen on paper with a special background (the pen and paper are discussed in detail in Chapter 3). When the user connects the pen to a PC, the stroke information stored on the pen is automatically downloaded and stored as a text file. Using Sketch Builder, the user can view the sketch and edit it, removing any strokes that do not belong in the final version. Sketch Builder also allows him to process his sketch with a recognizer or run it in a simulation by providing a link to Assist, an existing system developed by the Design Rationale Group. Once he is satisfied with the behavior of his design, he can email the file to another user, who can use the same software to print it out on another sheet of the special paper. This second user can then annotate the sketch using a different io2 Digital Pen. Once she connects the pen to her PC, a file containing her strokes is generated. She can then use Sketch Builder to combine her file with the original, and run recognition as well as a simulation. Finally, she can email the finished product back to the original user. At this point, the first user can use Sketch Builder to accept or reject her annotations on an individual basis, or add new strokes. Finally he can again run recognition and another simulation. If he is not yet satisfied with the design, the process can be repeated as necessary.
1.3 Preview

The rest of this thesis provides background information and specific details of the implemented system.

Chapter 2 provides background information that is helpful in understanding the task at hand, including a description of the research in the Design Rationale Group at MIT in the realm of sketch understanding, as well as a discussion of the concept of pen computing, a fledgling field with the potential to change the way people view and use computers. There is also an overview of the importance of collaboration in the design process, as well as information regarding computer supported collaboration design.

Chapter 3 discusses the components used to create the system and the functionality that each component contributes.

Chapter 4 details Sketch Builder, the graphical user interface that represents the bulk of this thesis.

Chapter 5 contains a detailed description of how the pieces mentioned in Chapter 4 fit together to produce the desired system. This chapter walks through a scenario of two engineers using the system to collaborate on the design of a simple mechanical device.

Chapter 6 presents a discussion of the successes of the project. A proposal for future research is presented in the hope that the work presented here will serve as a motivation for future developments.

Finally, Chapter 7 presents the main contributions of this thesis.
Chapter 2

Background

This chapter presents an overview of research relevant to the work presented here. It details some of the work being done in the Design Rationale Group at MIT, the importance of collaboration in the design process, and the field of pen computing.

2.1 The Design Rational Group and Assist

One of the high level goals of the Design Rational Group at MIT is improving the way in which humans interact with computers. The basic style of human computer interaction (the use of a mouse and keyboard) has remained practically unchanged for decades. This model of interaction is overly restrictive when using computer systems to do design work, and needs to be improved. The Design Rational Group is working to provide at least one such improved model.

When engineers start the design process, they naturally tend to produce many sketches. Unfortunately, it is common for these sketches to remain mostly inactive throughout the early stages of the process, until they are eventually entered into some software program (such as CAD) when they are almost complete. The translation from pen and paper sketches to the digital world in the early stages of design work presents a
challenging barrier. The Design Rationale Group has built a sketch recognition system, Assist, which addresses this problem by bridging the gap between the hard and soft copies of a sketch. Assist presents the user with a sketching interface that feels much like using a pen and paper (drawing with a stylus on a Tablet PC), and recognizes the user's strokes in real time (essentially assisting the user with his design). For example, Assist is capable of ascertaining the difference between a wheel and a pin joint in a mechanical engineering diagram. The system presented in this thesis links to Assist, allowing users to make sketches using pen and paper, and then to load the sketches into Assist for further editing and stroke recognition.

2.2 Collaboration in the Design Process

The main goal of this thesis is to present a model of interaction that improves collaboration during the design process, using pen and paper as the primary development tools. It is important first to understand how collaboration during the design process occurs in practice, and why it is important that it be improved.

Perry and Sanderson (1998) have presented extensive findings on how teams of engineers tend to collaborate in practice. They stress that the two main aspects of collaboration important in the design process are the way in which team members communicate and the artifacts that they produce. Oftentimes, these two aspects are very closely coupled. For example, they found that one of the most common paradigms for a design meeting involved the team members huddling around a large sketch, discussing aspects of the design and using it as an intermediary form of communication. They also discuss the need for "boundary objects," artifacts that serve to link together two distributed teams of
designers (such models or project work plans). The system presented in this thesis effectively turns a pen and paper sketch into a boundary object by allowing two users (or teams) in different locations to exchange it and the ideas it conveys in an iterative fashion. Perry and Sanderson also point out that it is important to pass around artifacts in a lossless manner. Unfortunately, this is not always the case, as faxed sketches often become unreadable after several roundtrips. The system presented in this thesis achieves this goal by using digital documents that are not prone to degradation.

It is also important to note that sketching affords excellent collaboration in idea generation meetings (Van der Lugt, 2005). This phenomenon is also discussed by Fish and Scrivener (1990):

"...sketches have the important function of assisting the mind to translate descriptive propositional information into depiction. This depictive information may then be scanned by attentional processes to extract new and perhaps original descriptive information, which in turn can lead to new depiction."

One of the most valuable characteristics of sketches is their ability to help engineers generate new ideas during the design process. The system presented in this thesis allows for older versions of a sketch to be easily stored (in both hard and soft copy) and thus takes advantage of this property of the sketching domain.

One final note of interest on the topic of collaboration in the design process is the distinction made by Thomas Kvan (2000) between collaborative and cooperative design. He argues that for a design process to be truly collaborative, the actions and decisions of the team
involved must be tightly coupled. If they are not, then the process is simply cooperative in nature, not collaborative. It is possible that the system presented in this thesis enables only cooperative design, not collaborative design, as defined by Kvan, since the model of interaction is linear (only a single user works on a design at a time). However, I believe it would be possible to extend the system to enable the type of collaborative design suggested by Kvan.

2.3 Pen Computing

The field of pen computing has been on the radar screen of people in both industry and academia for some time now. Just recently, LeapFrog Inc. released their “FLY pentop computer” which is designed to be an educational device for middle school aged children. The product works using some of the same components in the system presented in this thesis (namely the digital pen and paper), but adds to that a base of software that allows it to have useful and interesting interactions with its user. For example, users can draw a calculator, push their pen to the buttons, and hear the pen’s speech synthesizer report the answers, or write a word in English and have it translated into Spanish.

While the FLY pentop computer is an educational toy aimed at middle school children, other products like it could soon be developed for specific problem domains (such as writing music on staff lines or performing mathematical proofs). Users of such systems would benefit from a method of easily transferring documents back and forth to one another and collaborating on those documents in an efficient manner. The system presented here provides a method for doing both. It is my hope that research will continue to advance in the
field of pen computing, and that the work presented in this thesis will enhance the capabilities of the technologies produced by that research.
Chapter 3

System Components

This chapter presents an overview of each component in the system except the Sketch Builder interface, which is detailed in Chapter 4.

3.1 Anoto Group io2 Digital Pen

The most important component of the system is the Anoto io2 Digital Pen. This pen, which is .85 inches in diameter and weighs 1.3 ounces, automatically stores information about the strokes that are made with it. The pen contains a small camera near its tip that takes pictures at a rate of about 77 images per second. Each picture provides a sample point, with x coordinate, y coordinate, timestamp, and pressure reading. The pen does some processing to drop sample points that are collinear with their neighbors in order to conserve memory. The pen can be docked in a cradle that connects to a PC using a standard USB cable, and once docked in the cradle, the information stored in the pen is downloaded to the PC, and processing can begin.
3.2 Digital Paper

The pen relies on special “digital paper” that can be printed using a standard printer. Each sheet of this paper is covered with a special background comprising a large array of very small dots. The dots are originally arranged in a rectangular grid, each one placed .3 millimeters from its four neighbors. However, each dot is offset a small amount in one of the four cardinal directions, and thus contains two bits of information. The field of vision of the camera in the pen covers a circle with a radius of roughly one centimeter, allowing it to cover a total area of roughly 300 square millimeters. Each dot covers .09 square millimeters (as they are on average .3 millimeters apart), so the average image contains around 3000 dots. Each dot carries two bits of information, and thus the total information stored in the image is around 6000 bits, or nearly a kilobyte. This is enough information for the pen to uniquely identify its position on the page.

Using Anoto’s Forms Design Tool (basically a plug-in for Adobe 6.0 Professional), users can design their own digital forms, convert them to PostScript format, and print them for use. After designing the form, the user must generate a Paper Definition, or pad, file that uniquely identifies the form. Any number of forms can be designed, and it is possible to perform different processing on each by registering each to a different application (the Pen Download Software, explained in the next section, handles this registration and the corresponding lookup).

The paper also contains two areas that have logical meaning to the pen: the “New Logical Page Box” in the upper left hand corner and the “Send Box” in the lower right hand corner of the page. When starting a sketch, the user must mark the “New Logical Page Box”, in order to tell the pen that he drawing on a new sheet of digital paper. All the forms used by
our system have an identical dot pattern, so if this procedure is not followed the pen has no way to distinguish when two sketches should be treated as distinct objects. When a user has completed a sketch, he must make a mark in the “Send Box” in order to let the pen know he is finished. At this point, the pen stores in memory all the strokes that have been added since the last time the user marked the “New Logical Page Box” as a new sketch. The user can then either dock his pen to allow his sketch to be processed, or repeat the process and generate another sketch. When he does eventually dock the pen, each sketch will be processed as a unique request.

3.3 Pen Download Software

When the pen is docked in the cradle, the Paper Lookup Service, or PLS, recognizes it and polls for all the sketches that have not yet been downloaded. The PLS then ships off the information to the appropriate service, based on which form the information was drawn on. The PLS is told which services handle information from which forms by registering both the pad file for a given form and the associated service application with the PLS. This is done using a simple Visual Basic script, which can be found in the Appendix A.

3.4 Service Application

The service application is a C++ application that meets Anoto’s IServiceApplication interface by providing the NotifyClass method. When the service application’s NotifyClass method is invoked, it creates a Pen object that contains all the information from the sketch to be processed. The service application extracts this information and writes it in a specified
format to a text file that is stored on the local machine and can later be accessed by Sketch Builder. The specification of this file format is shown in Appendix B.

3.5 **Anoto Group Software Development Kit**

The service application uses the class library provided in the Anoto Group Software Development Kit. This library provides methods for extracting all the information from the Pen object received by the service application. For example, methods for obtaining an Iterator of Stroke objects from a Pen are provided, as well as methods for obtaining an Iterator of Samples from a Stroke object. Also, the x-y coordinates, as well as the timestamp and pressure information about each sample in the sketch can be easily extracted using this library.

3.6 **SketchBuilder**

SketchBuilder is the Java graphical user interface through which users can manipulate their hand-drawn sketches. It represents the bulk of the original work in this thesis, and is explained in detail in Chapter 4.

3.7 **Assist Stroke Recognizer and Working Model Software**

The Assist Stroke Recognizer was developed by the Design Rationale Group, and is capable of recognizing strokes in sketches involving various problem domains. For example, the user can draw a rough circle and, if it is close enough to an actual circle, Assist will recognize it as such and do the touch-up work of making the stroke into a perfect circle of
similar size. Other things that Assist can recognize include forces, joints, springs, dampers, and anchors. The original goal of Assist was to assist the user as she was authoring a sketch. As strokes are added, Assist attempts to classify them based on temporal and spatial information. If at any point the user is unhappy with Assist’s classification of a set of strokes, she can simply try drawing the object again. Once a user is satisfied with her sketch, she can click the “Run” button to open it in Working Model 2D, commercial software capable of running animated simulations of sketches. For example, if Assist recognized a circle and a downward pointing arrow in the drawing, upon opening Working Model the circle would fall according to the laws of physics, with gravity acting downward.
Chapter 4

Sketch Builder

The Sketch Builder graphical user interface provides users of the digital pen and paper with a computing environment in which they can perform efficient collaboration, as well as a link to preexisting sketch recognition and simulation software.

4.1 Direct Link to Working Model 2D

One of the most useful aspects of Sketch Builder is the direct link it provides to the Working Model 2D software. After a user creates a sketch using his digital pen, he docks the pen in its USB cradle. Upon recognizing the pen, the Pen Download software contacts the service application, which instantiates an instance of Sketch Builder, shown in Figure 4.1.1.
Figure 4.1.1: An example of a user’s sketch as seen in Sketch Builder, displayed automatically after docking the io2 Digital Pen.

This instance of Sketch Builder reads the raw pen data and translates it into a form that Assist can recognize (the drs file format specified in Appendix C) and passes the sketch to Assist, as shown in Figure 4.1.2.
Figure 4.1.2 An example of a user’s sketch as seen in Assist, displayed automatically after docking the io2 Digital Pen. In this example Assist has successfully recognized all the components in the sketch.

Once Assist has completed recognition, Sketch Builder opens the final product in Working Model 2D, as shown in Figures 4.1.3, 4.1.4, and 4.1.5. The entire process (docking the pen to running the simulation in Working Model 2D) takes between five and ten seconds.
Figure 4.1.3  An example of a user’s sketch as seen in Working Model 2D, displayed automatically after docking the io2 Digital Pen. This figure shows the simulation in its initial conditions.
Figure 4.1.4  An example of a simulation of a user’s sketch in Working Model 2D, after the simulation has begun.

Figure 4.1.5  A view of a completed simulation of a user’s sketch in Working Model 2D.
All of this occurs with no action on the user’s part after docking the pen in its cradle. Sketch Builder allows a digital pen user to bring his sketches to life instantly in an animated simulation. The link to Assist and Working Model 2D can also be utilized manually, by selecting the “View in Assist” option of the “View” menu. From Assist, the user can click the “Run” button to see his sketch simulated in Working Model 2D.

4.2 Accepting and Rejecting Changes

In order to enable collaboration, Sketch Builder provides the user with the ability to edit his sketch after a second user has made annotations. These annotations are created in the following way. The second user opens the original sketch using Sketch Builder and selects the “Save as PDF” option from the File Menu. He prints the resulting PDF on top of a sheet of digital paper and thereby creates a hard copy of the original sketch. He can then make annotations on this copy and send them back to the original user. The original user can then choose which changes he wants to keep and which he wants to reject, in the same fashion that users collaborating on a document use Microsoft Word. In the figure below, the user has opened the sketch from the previous section after his collaborator has made some annotations. The collaborator’s strokes appear in green, while the original author’s strokes appear in black.
Figure 4.2.1 A sketch that has been annotated by a second user. The original author’s strokes appear in black, while the collaborator’s strokes appear in green.

The left panel shows all the strokes in the sketch, while the right panel represents the final version. In essence, the user builds the sketch in the right panel from the strokes available in the left panel. The user can select a stroke either by clicking on it individually, or by holding the left mouse button and dragging to create a rectangle that encompasses the stroke. All selected strokes in a given panel are red, as shown below.
Figure 4.2.2  A sketch with selected strokes displayed in red. Users can select a stroke by either clicking it individually or dragging the mouse with the left button depressed to create a rectangle that encompasses the entire stroke.

The user can change which strokes appear in the final version by using the “Add” and “Remove” buttons. He can also access these actions by right clicking on the sketch to access a context menu. Finally, if the user selects “Drawing Mode” from the combo box on the right, he can draw entirely new strokes in the final version. This allows the user to make small additional changes suggested by the collaborator’s annotations. The final sketch could look like the one shown in Figure 4.2.3.
4.3 Temporal Selection

In general, selection is performed spatially. Users choose which strokes to add or remove based on their position within the sketch. However, Sketch Builder also provides the user with an alternative approach. The sliders at the bottom of the Sketch Builder GUI allow users to perform temporal selection. The top slider is the “Start” slider, the middle slider is the “End” slider, and the bottom slider is the “Speed” slider. The Start and End sliders represent a timeline that covers the life of the sketch. All the strokes that occurred between the values of the sliders are automatically selected in the left sketch. On startup, these sliders’ values are minimized, so that no strokes are selected. However, moving the End slider to the right causes strokes to be selected one by one, starting with the first stroke the original author produced. Moving the Start slider to the right unselects these strokes one at a time, in the
same order. This functionality allows users to select regions of a sketch in a temporal fashion, as if to say “I’m interested in what my collaborator was doing now” instead of “I’m interested in what my collaborator was doing here.”

Figure 4.3.1 A sketch on which temporal selection has been performed using the sliders at the bottom of Sketch Builder. Temporal selection can also be performed while the user watches a movie of the sketch being recreated.

Sketch Builder provides users with another method for performing temporal selection. By clicking the “View Movie” button on the left, users are able to watch their sketch be recreated in the right panel, just as it was drawn on the digital paper. If the sketch in the left panel has two parts (both green and black strokes) only the green strokes are played. This functionality allows the original author to “watch” as his collaborator annotates the sketch. If,
however, the sketch only has one part (only black strokes) then the entire sketch is played forward. This functionality allows sketches with only one author to be viewed in their entirety. As the sketch plays, the Start and End sliders’ values increase until the user clicks the “Set Start” or “Set End” buttons. Once either button is clicked, the corresponding slider stops moving. All strokes that occurred between the Start and End slider values are selected in the left panel. Finally, the user can change the speed at which the sketch plays by adjusting the Speed slider. This allows the user to effectively fast forward past pieces that do not interest him, while watching more intriguing portions of the sketch in slow motion.

4.4 Viewing and Analyzing Sketches

Sketch Builder aims to make viewing and analyzing sketches as easy as possible. Users can pan within a sketch in three ways: using the scroll bar, using the arrow keys, or right clicking and dragging the sketch. Sketch Builder also supports zooming, which is performed by pressing the ‘z’ and ‘x’ keys (zoom in and zoom out respectively). The combination of these two capabilities makes Sketch Builder a powerful tool for viewing the low-level details that are otherwise hidden within the sketch. Sketch Builder brings out these details further by allowing the user to view his sketch with the actual data points marked with red crosses. Thus, a user can pan to any section of a sketch, zoom in as close as he likes, and see the actual data points collected by the pen that represent his sketch. This is shown in Figures 4.4.1 and 4.4.2.
Figure 4.4.1  A zoomed-in view of the arrow shown in previous figures.
Figure 4.4.2 A zoomed-in view of the tip of the arrow shown in Figure 4.4.1. In this view, the user can see the individual data points that comprise the sketch. The tip of the arrow is very noisy, contributing to the difficulties inherent in sketch recognition.

As Figure 4.4.2 illustrates, the data points at the tip of the arrow are very noisy, a detail that is invisible to the human eye except at high magnification. Noise like this highlights why the problem of sketch understanding and recognition is so difficult.
Chapter 5

A Working Example

The best way to understand how any system works is to walk through a specific example of it in operation. This chapter will take the reader through a single iteration of the collaborative process using the implemented system in which two engineers are attempting to design a simple mechanical device. In the example, Andy will serve as the first designer, and Betty will be his collaborator.

5.1 The First User

Andy is attempting to design a mechanical system as part of a new product. He prints a sheet of digital and begins to draw using his digital pen. Once he is comfortable with his sketch, he docks the pen in its cradle and the data of his sketch is automatically downloaded to his machine, where it is saved in a directory of his choosing. Andy then opens Sketch Builder in order to manipulate his sketch. He views his sketch in Assist using the “View in Recognizer” command in the View menu. Assist appears to have recognized all of his components, so he clicks the run button to bring his system to life in Working Model. Content with his first attempt, Andy emails the text file to Betty.
5.2 The Second User

Betty receives Andy’s email and saves the attached file. She then runs Sketch Builder and uses it to open Andy’s file, just as he did in the previous step. She chooses the “Save as PDF” option from the File menu and creates a PDF version of Andy’s sketch. Next, Betty loads a single sheet of digital paper into her printer and prints the PDF on top of it. The piece of digital paper that comes out of the printer now contains the same strokes as Andy’s original hard copy, laid on top of an identical pattern of dots. Betty undocks her own io2 Digital Pen and begins to make annotations to the drawing. Once she is satisfied with her changes, she redocks the pen and a text file containing only her strokes is automatically saved on her machine. In Sketch Builder she selects “Add Strokes from File” from the File menu, and her strokes appear on the left pane of the GUI in green (Andy’s original strokes are shown in black). She opens the new sketch in Assist to confirm that the recognizer handles her new strokes appropriately, and then runs a simulation using Working Model to see that it works. Finally, Betty saves her sketch as a text file, which she emails back to Andy.

5.3 The Final Product

Andy receives Betty’s email and opens her file in Sketch Builder, just as she did. Now Andy sees two sketches on his screen. The one on the left contains his original strokes in black and Betty’s additions in green, while the one on the right contains only his original strokes in black. First, Andy clicks the “View as Movie” button on the GUI and Betty’s green strokes are drawn in the right pane in same speed and order as she added them. As the strokes are added, the slider knobs progress from left to right on the screen, the marking the relative point in time to which the sketch has progressed. As Andy watches the strokes appear on the
screen, he can mark a section of the annotations by clicking the “Start” and “Stop” buttons on the GUI. Clicking the “Start” button causes the top knob to stop sliding, while clicking the “Stop” button causes the bottom knob to stop sliding. All the strokes that are drawn between the clicks are selected in the left sketch, and thus are drawn in red. This model of selection allows Andy to pick a certain window of time during which Betty was making helpful additions, thus allowing temporal selection. If he desires, he can also move the knobs on the sliders manually, selecting a temporal region of the sketch without watching the movie.

Andy can also select and add strokes spatially. He performs selection either by clicking on a specific stroke (to select or deselect it), or by left-clicking the mouse and dragging it to create a rectangle and thus select all the strokes lying completely inside it. Andy adds strokes by using the buttons on the left side of the GUI or the popup menu that becomes visible when he right clicks the mouse over the left panel. Similarly, he can remove strokes from the finished version in the right panel by using the buttons under it or the options of the same name in a popup menu that appears when he right clicks on the sketch in the right panel. He can always revert the drawing on the right side of the GUI to his original efforts by clicking the “Reset” button. Once he has incorporated all the strokes he wants into his final sketch, Andy opens it in Assist, to make sure that recognition goes as he expects, and then runs a simulation using Working Model to make sure his device behaves as he expects. He then saves the final version. At this point, Andy is free to email the sketch back to Betty for further collaboration, and the process can repeat. Andy and Betty have collaborated on the design using pen and paper as their primary development environment, while not losing the ability to run sketch recognition and simulations on their designs.
Chapter 6

Discussion

It is my hope that Sketch Builder will be used to motivate and enable future research. One question that it has already helped answer is that of cross-medium analysis of sketches.

6.1 Cross-medium Analysis

It is known that the medium in which a sketch is drawn affects how a user creates that sketch. For example, drawing on a tablet PC requires a different set of motor skills than drawing on a large plasma display (say one the size of a drafting table). In the first case, users take advantage primarily of the dexterity in their fingers and wrist to compose the sketch. In the second case, however, the fingers and wrist cannot provide a wide enough range of motion, and other joints, including the elbow and shoulder, are often used to manipulate the drawing implement.

How do these differences in motion manifest themselves in the final sketch? The answer to this question could prove to be important in the realm of sketch recognition. Sketch Builder’s ability to zoom in on a particular portion of a sketch and view the individual data points allows users to compare sketches drawn on different mediums (assuming the data from the second medium can be translated into the drs format) and look for any differences that
may stand out. Sketch Builder was used in this fashion to determine if any obvious differences existed between data captured from the digital pen and data captured by drawing on a tablet PC. It was feared that if such differences existed Assist would have trouble recognizing the strokes produced by the digital pen. Three users drew multiple examples of eight components that Assist can recognize (anchors, circles, squares, springs, pulleys, pendulums, arrows, and motors) using both a tablet PC and the digital pen. The tablet data was converted to the drs format, and a side by side comparison was performed using Sketch Builder. No obvious differences existed in any of the components, and the percentage of successful recognition by Assist was approximately equal. These preliminary results indicate that sketching on a tablet PC is similar to sketching with a pen and paper. However, more analysis could reveal subtle differences that could prove important in the sketch recognition domain.
Chapter 7

Contributions

In this thesis I have made two key contributions:

- I have linked the systems developed by the Design Rationale Group at MIT (namely the sketch recognition program Assist and, through it, the commercial simulation software Working Model) to the io2 Digital Pen produced by the Anoto Group. This linkage frees those systems of the requirement that they be used on a tablet PC and allows them to be useful in the domain of pen and paper sketches.

- Implemented a functioning system that allows for effective and efficient collaboration between designers using only pen and paper to produce sketches. The system allows for both spatial and temporal selection, provides users with powerful tools for low-level sketch analysis, and lets users watch sketches being redrawn just as they were originally created.

It is my hope that Sketch Builder will provide engineers with a means for efficient collaboration on sketches developed using pen and paper. I believe some functionality provided by Sketch Builder, such as the ability to view the creation of a sketch as a movie
and the ability to view the individual data points within a sketch, will provide valuable tools in future research. It is important that we realize the power pen and paper sketches, and that we build tools that attempt to harness that power as new technology, such as the digital pen and paper from Anoto Group, becomes available. Sketch Builder is a step in this direction.
Appendix A

Visual Basic Registration Script

This Visual Basic script is used to register a form and corresponding service application with the Pen Download Software. Line 6 specifies the pad file that corresponds to the form (which is named automatically by the Forms Design Tool based on the license number provided by Anoto) while line 8 specifies the service that is to be called.

```vbscript
1    Dim registrationmngr, fso, path
2    ' Create needed objects:
3    Set registrationmngr = CreateObject("Anoto.SDK.RegistrationManager2")
4    Set fso = CreateObject("scripting.filesystemobject")
5    ' Register paper:
6    registrationmngr.RegisterPaper "C:\Documents and Settings\Jesse\Desktop\forms\DateAndInitialsTop\35.0.1.85.pad"
7    ' Register vbExample application:
8    registrationmngr.RegisterApplication "cppExample.NotifyClass"
```
Appendix B

File Format Specification of Raw Pen Data

This is an example of a file of data extracted from the pen. It is generated by code in the NotifyClass file of the C++ application that comes with the digital pen. The NotifyClass file has been changed slightly from its original state. To change the format of the file output by the pen, simply change the code in NotifyClass.

The first five lines of the output file serve as a header, giving the ID of the pen, the number of pages in the sketch, the form’s address in the Anoto Pattern Space, the page bounds, and the number of strokes in the sketch. The pen ID, number of pages in the sketch, and number of strokes in the sketch are all self-explanatory. The form’s address in the Anoto Pattern Space is determined by the license file provided by Anoto. The address is split into four sections, which correspond to the segment, shelf, book, and page numbers of the form. We have licensed only a single page of pattern space, and thus all of our forms have the exact same address. The page bounds specify the upper left and lower right hand corners of the dot pattern on the printed form.

Starting on line six, data about each stroke in the sketch is given. The first three lines again serve as a header, stating the number of samples, the color, and the start time of the stroke (given in seconds since January 1st, 1970). Each subsequent line contains four numbers that give information about a specific sample within the current stroke. The first two numbers correspond to the x and y coordinates of the sample. The coordinate system’s origin is in the upper left hand corner of the page, and increases as you move down and to the right. The unit of measure is an “Anoto Coordinate Unit”, which corresponds to .3 millimeters (about 86
units per inch). The third number is the relative time stamp. This number indicates the number of milliseconds that passed between the previous sample and the current one. Note that this number is zero for the first sample in a stroke. The final number is the pen's pressure sensor reading, which ranges from 0 to 127. I have been told that this pressure sensor reading is not very accurate, and my experiments seem to confirm this assessment. Thus, use it with caution. The coordinates and timestamps, however, are extremely accurate.

Pen id: AR5-AAB-ESS-JH
Number of pages: 1
Page address: 35.0.1.85
Page bounds: 0 0 930 719
Number of strokes: 2
Number of samples: 8
Color: 41 0 139
Start Time: 1114699528.419
227.375 657.000 0 0
227.250 656.875 13 86
227.125 656.750 40 114
228.875 673.625 133 122
228.375 709.750 54 122
228.875 716.375 13 122
230.750 727.000 133 122
230.500 725.625 27 104
Number of samples: 14
Color: 41 0 139
Start Time: 1114699529.301
272.250 673.125 0 42
272.125 672.125 13 114
272.500 671.125 13 114
273.750 669.500 14 116
281.625 664.125 26 118
289.250 663.000 14 118
298.500 666.125 13 120
308.250 673.750 13 120
317.625 684.500 14 120
325.125 696.750 13 122
330.875 708.250 13 122
334.500 716.625 14 120
337.125 720.625 13 120
338.500 720.375 13 86

45
Appendix C

DRS File Format Specification

This is an example of a file that represents a sketch in the drs format. The first three lines contain header information. Each subsequent line represents a single stroke in the sketch. Each line comprises triples that represent x, y, and time data for each sample in the stroke. This sketch has four strokes.

version=v2.0
timeConversion=1.0
name=filename
{ (193 164 1115150542888) (214 200 1115150542898) (224 212 1115150542908) (229 218 1115150542918) (233 223 1115150542928) (245 234 1115150542938) (247 237 1115150542948) (250 239 1115150542958) (251 244 1115150542968) (252 243 1115150542978) (251 244 1115150542988) (248 242 1115150543008) (245 242 1115150543018) } 
{ (172 259 1115150543299) (210 228 1115150543309) (230 212 111515054319) (250 198 1115150543299) (267 186 1115150543339) (279 177 1115150543349) (284 175 1115150543359) (289 173 1115150543369) } 
References


