The Automatic Recognition of Gestures

Dean Harris Rubine

December, 1991

CMU–CS–91–202

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Computer Science at Carnegie Mellon University.

Thesis Committee:

Roger B. Dannenberg, Advisor
Dario Giuse
Brad Myers
William A. S. Buxton, University of Toronto

Abstract

Gesture-based interfaces, in which the user specifies commands by simple freehand drawings, offer an alternative to traditional keyboard, menu, and direct manipulation interfaces. The ability to specify objects, an operation, and additional parameters with a single intuitive gesture makes gesture-based systems appealing to both novice and experienced users.

Unfortunately, the difficulty in building gesture-based systems has prevented such systems from being adequately explored. This dissertation presents work that attempts to alleviate two of the major difficulties: the construction of gesture classifiers and the integration of gestures into direct-manipulation interfaces. Three example gesture-based applications were built to demonstrate this work.

Gesture-based systems require classifiers to distinguish between the possible gestures a user may enter. In the past, classifiers have often been hand-coded for each new application, making them difficult to build, change, and maintain. This dissertation applies elementary statistical pattern recognition techniques to produce gesture classifiers that are trained by example, greatly simplifying their creation and maintenance. Both single-path gestures (drawn with a mouse or stylus) and multiple-path gestures (consisting of the simultaneous paths of multiple fingers) may be classified. On a 1 MIPS workstation, a 30-class single-path recognizer takes 175 milliseconds to train (once the examples have been entered), and classification takes 9 milliseconds, typically achieving 97% accuracy. A method for classifying a gesture as soon as it is unambiguous is also presented.

This dissertation also describes GRANDMA, a toolkit for building gesture-based applications based on Smalltalk's Model/View/Controller paradigm. Using GRANDMA, one associates sets of gesture classes with individual views or entire view classes. A gesture class can be specified at runtime by entering a few examples of the class, typically 15. The semantics of a gesture class can be specified at runtime via a simple programming interface. Besides allowing for easy experimentation with gesture-based interfaces, GRANDMA sports a novel input architecture, capable of supporting multiple input devices and multi-threaded dialogues. The notion of virtual tools and semantic feedback are shown to arise naturally from GRANDMA's approach.

Acknowledgments

First and foremost, I wish to express my enormous gratitude to my advisor, Roger Dannenberg. Roger was always there when I needed him, never failing to come up with a fresh idea. In retrospect, I should have availed myself more than I did. In his own work, Roger always addresses fundamental problems, and his solutions are always simple and elegant. I try to follow Roger's example in my own work, usually falling far short. Roger, thank you for your insight and your example. Sorry for taking so long.

I was incredibly lucky that Brad Myers showed up at CMU while I was working on this research. His seminar on user interface software gave me the knowledge and breadth I needed to approach the problem of software architectures for gesture-based systems. Furthermore, his extensive comments on drafts of this document improved it immensely. Much of the merit in this work is due to him. Thank you, Brad. I am also grateful to Bill Buxton and Dario Giuse, both of whom provided valuable criticism and excellent suggestions during the course of this work.

It was Paul McAvinney's influence that led me to my thesis topic; had I never met him, mine would have been a dissertation on compiler technology. Paul is an inexhaustible source of ideas, and this thesis is really the *second* idea of Paul's that I've spent multiple years pursuing. Exploring Paul's ideas could easily be the life's work of hundreds of researchers. Thanks, Paul, you madman you.

My wife Ruth Sample deserves much of the credit for the existence of this dissertation. She supported me immeasurably, fed me and clothed me, made me laugh, motivated me to finish, and lovingly tolerated me the whole time. Honey, I love you. Thanks for everything.

I could not have done it with the love and support of my parents, Shirley and Stanley, my brother Scott, my uncle Donald, and my grandma Bertha. For years they encouraged me to be a doctor, and they were not the least bit dismayed when they found out the kind of doctor I wanted to be. They hardly even balked when "just another year" turned out to be six. Thanks, folks, you're the best. I love you all very much.

My friends Dale Amon, Josh Bloch, Blaine Burks, Paul Crumley, Ken Goldberg, Klaus Gross, Gary Keim, Charlie Krueger, Kenny Nail, Eric Nyberg, Barak Pearlmutter, Todd Rockoff, Tom Neuendorffer, Marie-Helene Serra, Ellen Siegal, Kathy Swedlow, Paul Vranesevic, Peter Velikonja, and Brad White all helped me in innumerable ways, from technical assistance to making life worth living. Peter and Klaus deserve special thanks for all the time and aid they've given me over the years. Also, Mark Maimone and John Howard provided valuable criticism which helped me prepare for my oral examination. I am grateful to you all.

I wish to also thank my dog Dismal, who was present at my feet during much of the design, implementation, and writing efforts, and who concurs on all opinions. Dismal, however, strongly objects to this dissertation's focus on *human* gesture.

I also wish to acknowledge the excellent environment that CMU Computer Science provides; none of this work would have been possible without their support. In particular, I'd like to thank Nico Habermann and the faculty for supporting my work for so long, and my dear friends Sharon Burks, Sylvia Berry, Edith Colmer, and Cathy Copetas.

DOCKET ALARM

Explore Litigation Insights

Docket Alarm provides insights to develop a more informed litigation strategy and the peace of mind of knowing you're on top of things.

Real-Time Litigation Alerts

Keep your litigation team up-to-date with **real-time alerts** and advanced team management tools built for the enterprise, all while greatly reducing PACER spend.

Our comprehensive service means we can handle Federal, State, and Administrative courts across the country.

Advanced Docket Research

With over 230 million records, Docket Alarm's cloud-native docket research platform finds what other services can't. Coverage includes Federal, State, plus PTAB, TTAB, ITC and NLRB decisions, all in one place.

Identify arguments that have been successful in the past with full text, pinpoint searching. Link to case law cited within any court document via Fastcase.

Analytics At Your Fingertips

Learn what happened the last time a particular judge, opposing counsel or company faced cases similar to yours.

Advanced out-of-the-box PTAB and TTAB analytics are always at your fingertips.

API

Docket Alarm offers a powerful API (application programming interface) to developers that want to integrate case filings into their apps.

LAW FIRMS
Build custom dashboards for your attorneys and clients with live data direct from the court.

Automate many repetitive legal tasks like conflict checks, document management, and marketing.

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

E-DISCOVERY AND LEGAL VENDORS
Sync your system to PACER to automate legal marketing.