

Designing Gestural Interfaces

Dan Saffer

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Sebastopol • Taipei • Tokyo

Designing Gestural Interfaces

by Dan Saffer

Copyright © 2009 O'Reilly Media, Inc. All rights reserved.
Printed in Canada.

Published by O'Reilly Media, Inc., 1005 Gravenstein Highway North, Sebastopol, CA 95472.

O'Reilly books may be purchased for educational, business, or sales promotional use. Online editions are also available for most titles (safari.oreilly.com). For more information, contact our corporate/institutional sales department: (800) 998-9938 or corporate@oreilly.com.

Editor: Mary Treseler

Production Editor: Rachel Monaghan

Copyeditor: Audrey Doyle

Proofreader: Katie Nopper DePasquale

Indexer: Julie Hawks

Cover Designer: Karen Montgomery

Interior Designer: Ron Bilodeau

Illustrator: Jessamyn Read

Printing History:

November 2008: First Edition.

Nutshell Handbook, the Nutshell Handbook logo, and the O'Reilly logo are registered trademarks of O'Reilly Media, Inc. *Designing Gestural Interfaces*, the image of a channel-billed toucan, and related trade dress are trademarks of O'Reilly Media, Inc.

Many of the designations used by manufacturers and sellers to distinguish their products are claimed as trademarks. Where those designations appear in this book, and O'Reilly Media, Inc. was aware of a trademark claim, the designations have been printed in caps or initial caps.

While every precaution has been taken in the preparation of this book, the publisher and author assume no responsibility for errors or omissions, or for damages resulting from the use of the information contained herein.



This book uses Repkover,[™] a durable and flexible lay-flat binding.

ISBN: 978-0-596-51839-4

[F]

Contents

	Preface.....	xiii
Chapter 1	Introducing Interactive Gestures	1
	Tap Is the New Click	3
	Direct Versus Indirect Manipulation	4
	A Brief History of Gestural Interfaces	6
	The Mechanics of Touchscreens and Gestural Controllers	12
	Designing Interactive Gestures: The Basics	16
	The Characteristics of Good Gestural Interfaces	19
	The Attributes of Gestures	22
	Interface Conventions	25
	Determining the Appropriate Gesture	27
	Using People to Determine the Appropriate Gesture	28
	Matching the Gesture to the Behavior	29
	For Further Reading	30
Chapter 2	Designing for the Human Body.....	31
	Basic Kinesiology	31
	Muscles	31
	Joints	32
	Motions	35
	The Ergonomics of Interactive Gestures	37
	Human Factors	37
	The Ergonomics of Motion	38
	Fingers and Hands	39
	Screen Coverage	40
	Designing Touch Targets	41
	Accessibility	44
	For Further Reading	44

Chapter 3	Patterns for Touchscreens and Interactive Surfaces.....	45
	How to Use Patterns	45
	Tap to Open/Activate	46
	What	46
	Use When	46
	Why	46
	How	46
	Examples	47
	Tap to Select	48
	What	48
	Use When	48
	Why	48
	How	48
	Examples	48
	Drag to Move Object	50
	What	50
	Use When	50
	Why	50
	How	50
	Examples	50
	Slide to Scroll	52
	What	52
	Use When	52
	Why	52
	How	52
	Examples	53
	Spin to Scroll	54
	What	54
	Use When	54
	Why	54
	How	54
	Examples	55

Slide and Hold for Continuous Scroll	56
What	56
Use When	56
Why	56
How	56
Examples	57
Flick to Nudge	58
What	58
Use When	58
Why	58
How	58
Examples	59
Fling to Scroll	60
What	60
Use When	60
Why	60
How	60
Examples	61
Tap to Stop	62
What	62
Use When	62
Why	62
How	62
Examples	62
Pinch to Shrink and Spread to Enlarge	64
What	64
Use When	64
Why	64
How	64
Examples	65
Two Fingers to Scroll	66
What	66
Use When	66
Why	66

How	66
Examples	67
Ghost Fingers	68
What	68
Use When	68
Why	68
How	68
Examples	69

Chapter 4 **Patterns for Free-form Interactive Gestures 71**

Proximity Activates/Deactivates	71
What	71
Use When	71
Why	71
How	72
Examples	72
Move Body to Activate	74
What	74
Use When	74
Why	74
How	74
Examples	74
Point to Select/Activate	76
What	76
Use When	76
Why	76
How	76
Examples	76
Wave to Activate	78
What	78
Use When	78
Why	78

How	78
Examples	78
Place Hands Inside to Activate	80
What	80
Use When	80
Why	80
How	80
Examples	80
Rotate to Change State	82
What	82
Use When	82
Why	82
How	82
Examples	82
Step to Activate	84
What	84
Use When	84
Why	84
How	84
Examples	84
Shake to Change	86
What	86
Use When	86
Why	86
How	86
Examples	86
Tilt to Move	88
What	88
Use When	88
Why	88
How	88
Examples	88

Chapter 5	Documenting Interactive Gestures	91
	Why Document Anything?	91
	Existing Movement Notation Systems	92
	Labanotation	92
	Benesh Movement Notation	95
	Eshkol Wachman Movement Notation	96
	Documenting Gestures in Interactive Systems	99
	Scenarios	99
	Use Cases	100
	Task Analysis and Flows	101
	Wireframes	102
	Gestural Modules Document	106
	Storyboarding	107
	Animations	111
	Movies	114
	For Further Reading	116
Chapter 6	Prototyping Interactive Gestures	117
	Faking It: Low-Fidelity Prototypes	118
	Paper Prototypes	118
	Rough Models: Objects and Environments	119
	“The Man Behind the Curtain”	122
	High-Fidelity Prototypes	123
	The Three Types of High-Fidelity Prototypes	124
	Matching Sensors to Actions	135
	Testing Prototypes	137
	Prototyping Resources	137
	Touchscreen Resources	137
	Physical Computing Resources	138
	For Further Reading	139
Chapter 7	Communicating Interactive Gestures	141
	Three Zones of Engagement	143
	Attraction Affordance, or How I Learned to Stop Worrying and Love the Gesture	145

	Methods of Communicating Interactive Gestures	146
	Written Instructions	146
	Illustration	148
	Demonstration	151
	Use of Symbols	152
	For Further Reading	154
Chapter 8	The Future of Interactive Gestures	155
	Future Trends	155
	Widespread Adoption in the Home and Office	155
	Specialized Applications and Products	160
	Convergence with Other Technologies	163
	Gesturing with Smart Objects	165
	Optimized for Touch	169
	Better Tools for Creation	170
	Robots and Cobots	170
	Gestures As a Design Tool	172
	Toward Standards	174
	Patents	174
	The Ethics of Gestures	176
	For Further Reading	178
Appendix	A Palette of Human Gestures and Movements	179
	Gestures for Touchscreens	181
	Gestures for Free-Form Systems	183
	Head	183
	Torso	185
	Arms	199
	Face	205
	Hands	210
	For Further Reading	233
	Index	235

Preface

For the past several years, I've been learning to play the cello. As any adult who has ever tried to learn a musical instrument—especially a fretless stringed instrument—can tell you, it's painfully hard. You feel like you don't really know your body at all: your muscles get sore in odd places, you develop strange calluses on your fingers, and you perform stretches with your hands you never thought possible. It requires almost all of your concentration simply to coax the instrument into making a pleasing sound, which is very different from actually making music.

But, ah, when you do begin to make music, it is a thing of joy. When you are able to get a sweet sound with emotional resonance from an inanimate object using only your body, it is an amazing feeling. Gestures create meaning.

That's what this book is about.

Technological, social, and market forces have converged to create a fertile new ground for designers and engineers to plow. The price of processing speed has dropped and sensors are readily available. Touchscreens on our mobile devices, ATMs, and airline check-in kiosks have taught us to expect to be able to manipulate things on-screen with our hands. Games have shown us we can make gestures in space to control objects on-screen. Public restrooms are, believe it or not, test laboratories for interactive gestures: placing your hands under a faucet to turn it on, waving your hands to get a paper towel, stepping into a room to turn on the lights.

All of these things have ushered in a new era of interaction design, one where gestures on a surface and in the air replace (or at least supplement) keyboards, mice, and styli. This new era, however, means those who design and develop more "traditional" systems need to grow their skills, adding in knowledge about kinesiology, sensors, ergonomics, physical computing, touchscreen technology, and new interface patterns.

That's what this book is about.

I wrote this book because, on my third gestural interface project, I was frustrated that I could find only bits and pieces of information about a subject—this subject—that was obviously important and growing rapidly. Why wasn't there a book about these things? I groused. How come I have no idea how big a touch-screen button needs to be? What is a capacitive touchscreen? What kinds of gestures can I use? And thus the idea for the book you are reading was born.

Aside from the constraints of the human body, many of the other parts of this book were changing even as I wrote about them, and new products that utilized interactive gestures seemingly came out every week, if not several times a week, over the course of the nine months I wrote this book (November 2007–July 2008). The newness of the subject was glaringly apparent, but I have tried, when and where possible, to emphasize what is unlikely to change and techniques that are, if not proven, at least being used by practicing designers and developers in the field today.

WHO SHOULD READ THIS BOOK

This book is mostly for interaction and industrial designers who have found themselves in the same position I found myself in several years ago: being asked to design gestural interfaces (especially touchscreens) and, coming from the world of web and desktop software or physical products, having no central place to go to for information to get started. I hope this book is a starting place and a reference point.

And even though there isn't a line of code in this book, I also wrote it with developers in mind, knowing full well that they are often the people who have to take what designers dream up and make it real. I hope everyone involved in creating new products that make use of sensors and interactive surfaces will find something of use in these pages.

HOW TO USE THIS BOOK

This book is divided into roughly four parts. Although certainly you can read the book straight through, you don't have to. I recommend that everyone start with Chapters 1 and 2 just to provide a background for what comes later.

The next section, comprising Chapters 3 and 4, is meant to be used as reference material, mostly when you're designing or are in the process of designing. The patterns show how other designers have solved interface challenges in the past.

Chapters 5 through 7 are about the process of design, from documentation to prototyping to communicating what the product does to its audience. You also can use these chapters for reference as necessary.

Chapter 8 takes a look at future trends in this subject—a future that gets closer and closer every day. It was, after all, only six years ago that the gestural interfaces in *Minority Report* were science fiction, and now we can see them being deployed everywhere. “The future is here,” as William Gibson famously noted. “It’s just not evenly distributed.”

Designers can flip through the appendix for inspiration, especially when creating free-form gestural interfaces.

SAFARI® BOOKS ONLINE

When you see a Safari® Books Online icon on the cover of your favorite technology book, that means the book is available online through the O’Reilly Network Safari Bookshelf.

Safari offers a solution that’s better than e-books. It’s a virtual library that lets you easily search thousands of top tech books, cut and paste code samples, download chapters, and find quick answers when you need the most accurate, current information. Try it for free at <http://safari.oreilly.com>.

ACKNOWLEDGMENTS

It’s easy to come up with an idea for a book, but it’s much harder to write one. It’s a task that, like most other products, requires lots of people to make it a reality.

This book would not be possible without the patience, support, and 6:00 a.m. writing wake-up calls of my wife, Rachael King. Her encouragement and understanding kept me going on those days when I could eek out only a sentence—which were many.

My daughter Fiona’s insistence that I not work on this book and instead play Wii with her also, I’m sure, kept me sane and contributed vastly to the quality of this tome. (At the very least, I got to practice many interactive gestures!)

My family and friends have my undying gratitude for their love, patience, and support throughout this process, especially my parents.

My former colleagues at Adaptive Path fed me a constant diet of new material, both in links to interesting stuff and in projects to work on that constantly expanded what I knew—or needed to know—to make this book as rich as it is.

This book would be boring indeed without the images contained therein, and I'd like to thank the companies that contributed them and the individuals who graciously made their photographs available via Creative Commons licenses for commercial use.

My colleague and friend, Rachel Graves, created many of the illustrations in the book, and the book is much improved because of it.

My other colleague and friend, Sarah B. Nelson, took the photographs in the appendix and made them look good. My model, Ellen Ho, really brought the gestures to life.

I also benefited greatly from the review of others who read and commented on the manuscript as it was in development. For their advice and expertise, I thank them: Bill DeRouchey, Kevin Arthur, Nathan Moody, Juhan Sonin, Cem Keskin, Chong Lee Khoo, Dan Harrelson, Kate Fitch, and Jennifer Bove.

The contributors to the Interaction Design Association's mailing list and especially to the Interactive Gestures Wiki were an enormous help and source of material and inspiration.

I also want to thank the staff at O'Reilly who took a chance on this book and have supported its development and promotion: Jacque McIlvaine, Marlowe Shaeffer, Katheryn Barrett, Laurel Ackerman, Ron Bilodeau, Rachel Monaghan, Adam Witwer, Audrey Doyle, Jessamyn Read, Tim O'Reilly, and especially my editor, Mary Treseler.

San Francisco, California

August 2008

Introducing Interactive Gestures

“One of the things our grandchildren will find quaintest about us is that we distinguish the digital from the real.”

—William Gibson in a *Rolling Stone* interview, November 7, 2007

A man wearing special gloves stands in front of a large, translucent screen. He waves his hand in front of it, and objects on the screen move. It's as though he's conducting an orchestra or is some sort of high-tech sorcerer's apprentice, making objects fly about with just a sweep of his arm. He makes another gesture, and a video begins to play. With both hands, he stretches the video to a larger size, filling more of the screen. It's like magic.

Another place, another time: a different man stands in front of an audience. He's running his fingers over a table-size touchscreen before him as though he is a keyboard player in a rock band, his fingers rapidly manipulating images on the screen by dragging them around. He's making lines appear on-screen with his fingers and turning them into silky, ink-like paintings. He's playing, really—showing off. He drags his fingers across the surface and leaves a trail of bubbles. It's also like magic.

The first man doesn't really exist, although you'd probably recognize the actor playing him: Tom Cruise. The scene is from the movie *Minority Report* (2002), and it gave the general public its first look at a computer that responds to gestures instead of to speech, a keyboard, or a mouse. It was an impressive feat of visual effects, and it made a huge impression on people everywhere, especially interaction designers, some of whom had been working on or thinking about similar systems for years.

The second man does exist, and his name is Jeff Han. Not only did his jumbo touchscreen devices influence *Minority Report*, but his live demonstrations—first privately and then publicly at the 2006 TED conference*—will likely go down in computer history near the “Mother of All Demos” presentation that Doug Engelbart made in 1968, in which he showed now-familiar idioms such as

* Watch the demo yourself at <http://www.ted.com/index.php/talks/view/id/65>.

email, hypertext, and the mouse. Han's demos sparked thousands of conversations, blog posts, emails, and commentary.



Figure 1-1. Jeff Han demos a multitouch touchscreen at the 2006 TED conference. Since then, Han has created Perceptive Pixel, a company that produces these devices for high-end clients. Courtesy TED Conferences, LLC.

Since then, consumer electronics manufacturers such as Nintendo, Apple, Nokia, Sony Ericsson, LG, and Microsoft have all released products that are controlled using interactive gestures. Within the next several years, it's not an exaggeration to say that hundreds of millions of devices will have gestural interfaces. A **gesture**, for the purposes of this book, is any physical movement that a digital system can sense and respond to without the aid of a traditional pointing device such as a mouse or stylus. A wave, a head nod, a touch, a toe tap, and even a raised eyebrow can be a gesture.

In addition to touchscreen kiosks that populate our airports and execute our banking as ATMs, the most famous of the recent products that use gestures are Nintendo's Wii and Apple's iPhone and iPod Touch. The Wii has a set of wireless controllers that users hold to play its games. Players make movements in space that are then reflected in some way on-screen. The iPhone and iPod Touch are devices that users control via touching the screen, manipulating digital objects with a tap of a fingertip.



Figure 1-2. Rather than focusing on the technical specs of the gaming console like their competitors, Nintendo designers and engineers focused on the controllers and the gaming experience, creating the Wii, a compelling system that uses gestures to control on-screen avatars. Courtesy Nintendo.

TAP IS THE NEW CLICK

We've entered a new era of interaction design. For the past 40 years, we have been using the same human-computer interaction paradigms that were designed by the likes of Doug Engelbart, Alan Kay, Tim Mott, Larry Tesler, and others at Xerox PARC in the 1960s and 1970s. Cut and paste. Save. Windows. The desktop metaphor. And so many others that we now don't even think about when working on our digital devices. These interaction conventions will continue, of course, but they will also be supplemented by many others that take advantage of the whole human body, of sensors, of new input devices, and of increased processing power.

We've entered the era of interactive gestures.

The next several years will be seminal years for interaction designers and engineers who will create the next generation of interaction design inputs, possibly defining them for decades to come. We will design new ways of interacting with our devices, environment, and even each other. We have an opportunity that comes along only once in a generation, and we should seize it. How we can create this new era of interactive gestures is what this book is about.

Currently, most gestural interfaces can be categorized as either **touchscreen** or **free-form**. Touchscreen gestural interfaces—or, as some call them, touch user interfaces (TUIs)—require the user to be touching the device directly. This puts a constraint on the types of gestures that can be used to control it. Free-form gestural interfaces don’t require the user to touch or handle them directly. Sometimes a controller or glove is used as an input device, but even more often (and increasingly so) the body is the only input device for free-form gestural interfaces.

Our relationship to our digital technology is only going to get more complicated as time goes on. Users, especially sophisticated users, are slowly being trained to expect that devices and appliances will have touchscreens and/or will be manipulated by gestures. But it’s not just early adopters: even the general public is being exposed to more and more touchscreens via airport and retail kiosks and voting machines, and these users are discovering how easy and enjoyable they are to use.

DIRECT VERSUS INDIRECT MANIPULATION

The ease of use one experiences with a well-designed touchscreen comes from what University of Maryland professor Ben Shneiderman coined as **direct manipulation** in a seminal 1983 paper.* Direct manipulation is the ability to manipulate digital objects on a screen without the use of command-line commands—for example, dragging a file to a trash can on your desktop instead of typing `del` into a command line. As it was 1983, Shneiderman was mostly talking about mice, joysticks, and other input devices, as well as then-new innovations such as the desktop metaphor.

Touchscreens and gestural interfaces take direct manipulation to another level. Now, users can simply touch the item they want to manipulate right on the screen itself, moving it, making it bigger, scrolling it, and so on. This is the ultimate in direct manipulation: using the body to control the digital (and sometimes even the physical) space around us. Of course, as we’ll discuss in Chapter 4, there are indirect manipulations with gestural interfaces as well. One simple example is The Clapper.

* Shneiderman, Ben. “Direct Manipulation: A Step Beyond Programming Languages.” *IEEE Computer* 16(8): 57–69, August 1983.



Figure 1-3. *The Clapper turns ordinary rooms into interactive environments. Occupants use indirect manipulation in the form of a clap to control analog objects in the room. Courtesy Joseph Enterprises.*

The Clapper was one of the first consumer devices sold with an auditory sensor.* It plugs into an electrical socket, and then other electronics are plugged into it. You clap your hands to turn the electrical flow off (or on), effectively turning off (or on) whatever is plugged into The Clapper. It allows users indirect control over their physical environment via an interactive gesture: a clap.

This use of the whole body, however, can be seen as the more natural state of user interfaces. (Indeed, some call interactive gestures natural user interfaces [NUIs].) One could argue, in fact, that the current “traditional” computing arrangement of keyboard, mouse, and monitor goes against thousands of years of biology. As a 1993 *Wired* article on Stanford professor David Liddle notes:†

“We’re using bodies evolved for hunting, gathering, and gratuitous violence for information-age tasks like word processing and spreadsheet tweaking. And gratuitous violence.

“Humans are born with a tool kit at least 15,000 years old. So, Liddle asks, if the tool kit was designed for foraging and mammoth trapping, why not try to make the tasks we do with our machines today look like the tasks the body was designed for? ‘The most nearly muscular mentality that we use (in computation) is pointing with a mouse,’ Liddle says. ‘We use such a tiny part of our repertoire of sound and motion and vision in any interaction with an electronic system. In retrospect, that seems strange and not very obvious why it should be that way.’

“Human beings possess a wide variety of physical skills—we can catch baseballs, dodge projectiles, climb trees—which all have a sort of ‘underlying computational power’ about them. But we rarely take

* The Clapper also had an iconic commercial with an extremely catchy jingle: “Clap on! Clap off!” See the commercial and watch The Clapper in action at <http://www.youtube.com/watch?v=WsxcdVbE3mI>.

† “Dogs Don’t Do Math,” by Tom Bestor, November 1993. Found online at <http://www.wired.com/wired/archive/1.05/dogs.html>.

advantage of these abilities because they have little evolutionary value now that we're firmly ensconced as the food chain's top seed."

Fifteen years after Liddle noted this, it's finally changing.

A BRIEF HISTORY OF GESTURAL INTERFACES

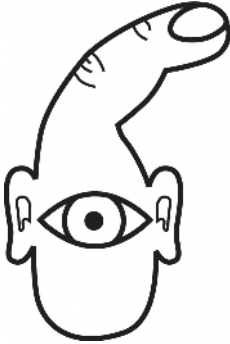


Figure 1-4. *How the computer sees us. With traditional interfaces, humans are reduced to an eye and a finger. Gestural interfaces allow for fuller use of the human body to trigger system responses. Courtesy Dan O'Sullivan and Tom Igoe.*



Figure 1-5. *If you don't need a keyboard, mouse, or screen, you don't need much of an interface either. You activate this faucet by putting your hands beneath it. Of course, this can lead to confusion. If there are no visible controls, how do you know how to even turn the faucet on? Courtesy Sloan Valve Company.*

As The Clapper illustrates, gestural interfaces are really nothing new. In one sense, everything we do with digital devices requires some sort of physical action to create a digital response. You press a key, and a letter or number appears on-screen. You move a mouse, and a pointer scurries across the screen.

What is different, though, between gestural interfaces and traditional interfaces is simply this: gestural interfaces have a much wider range of actions with which to manipulate a system. In addition to being able to type, scroll, point and click, and perform all the other standard interactions available to desktop systems,* gestural interfaces can take advantage of the whole body for triggering system behaviors. The flick of a finger can start a scroll. The twist of a hand can transform an image. The sweep of an arm can clear a screen. A person entering a room can change the temperature.

Removing the constraints of the keyboard-controller-screen setup of most mobile devices and desktop/laptop computers allows devices employing interactive gestures to take many forms. Indeed, the form of a “device” can be a physical object that

* This has some notable exceptions. See later in this chapter.

is usually analog/mechanical. Most touchscreens are like this, appearing as normal screens or even, in the case of the iPhone and iPod Touch, as slabs of black glass. And the “interface”? Sometimes all but invisible. Take, for instance, the motion-activated sinks now found in many public restrooms. The interface for them is typically a small sensor hidden below the faucet that, when detecting movement in the sink (e.g., someone putting her hands into the sink), triggers the system to turn the water on (or off).

Computer scientists and human-computer interaction advocates have been talking about this kind of “embodied interaction” for at least the past two decades. Paul Dourish in his book *Where the Action Is* captured the vision well:

“By embodiment, I don’t mean simply physical reality, but rather, the way that physical and social phenomena unfold in real time and real space as a part of the world in which we are situated, right alongside and around us...Interacting in the world, participating in it and acting through it, in the absorbed and unreflective manner of normal experience.”

As sensors and microprocessors have become faster, smaller, and cheaper, reality has started to catch up with the vision, although we still have quite a way to go.

Of course, it hasn’t happened all at once. Samuel C. Hurst created the first touch device in 1971, dubbed the Elograph.* By 1974, Hurst and his new company, Elographics, had developed five-wire resistive technology, which is still one of the most popular touchscreen technologies used today. In 1977, Elographics, backed by Siemens, created Accutouch, the first true touchscreen device. Accutouch was basically a curved glass sensor that became increasingly refined over the next decade.

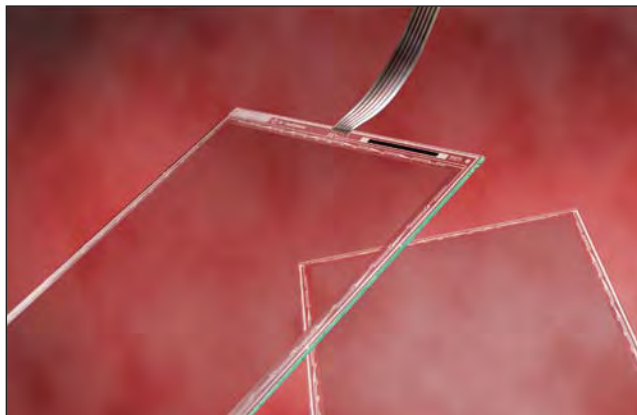


Figure 1-6. The Accutouch, the first real “touchscreen.” Courtesy Elo TouchSystems.

* See <http://www.elotouch.com/AboutElo/History/> for a detailed history of the Elograph.

Myron Krueger created in the late 1970s what could rightly be called the first indirect manipulation interactive gesture system, dubbed VIDEOPLACE. VIDEOPLACE (which could be a wall or a desk) was a system of projectors, video cameras, and other hardware that enabled users to interact using a rich set of gestures without the use of special gloves, mice, or styli.



Figure 1-7. In VIDEOPLACE, users in separate rooms were able to interact with one another and with digital objects. Video cameras recorded users' movements, then analyzed and transferred them to silhouette representations projected on a wall or screen. The sense of presence was such that users actually jumped back when their silhouette touched that of other users. Courtesy Matthias Weiss.

In 1982, Nimish Mehta at the University of Toronto developed what could be the first **multitouch** system, the Flexible Machine Interface, for his master's thesis.* Multitouch systems allow users more than one contact point at a time, so you can use two hands to manipulate objects on-screen or touch two or more places on-screen simultaneously. The Flexible Machine Interface combined finger pressure with simple image processing to create some very basic picture drawing and other graphical manipulation.

Outside academia, the 1980s found touchscreens making their way to the public first (as most new technology does) in commercial and industrial use, particularly in point-of-sale (POS) devices in restaurants, bars, and retail environments. Currently, touchscreen POS devices have penetrated more than 90% of food and beverage establishments in the United States.†

* Mehta, Nimish. "A Flexible Machine Interface." M.A.Sc. thesis, Department of Electrical Engineering, University of Toronto, 1982. Supervised by Professor K.C. Smith.

† According to *The Professional Bar and Beverage Manager's Handbook*, by Amanda Miron and Douglas Robert Brown (Atlantic Publishing Company).



Figure 1-8. A POS touchscreen. According to the National Restaurant Association, touchscreen POS systems pay for themselves in savings to the establishment. Courtesy GVISION USA, Inc.

The Hewlett-Packard 150 was probably the first computer sold for personal use that incorporated touch. Users could touch the screen to position the cursor or select on-screen buttons, but the touch targets (see later in this chapter) were fairly primitive, allowing for only approximate positioning.



Figure 1-9. Released in 1983, the HP 150 didn't have a traditional touchscreen, but a monitor surrounded by a series of vertical and horizontal infrared light beams that crossed just in front of the screen, creating a grid. If a user's finger touched the screen and broke one of the lines, the system would position the cursor at (or more likely near) the desired location, or else activate a soft function key. Courtesy Hewlett-Packard.

At Rank EuroPARC, Pierre Wellner designed the Digital Desk in the early 1990s.* The Digital Desk used video cameras and a projector to project a digital surface onto a physical desk, which users could then manipulate with their hands.† Notably, the Digital Desk was the first to use some of the emerging patterns of interactive gestures such as *Pinch to Shrink* (see Chapter 3).

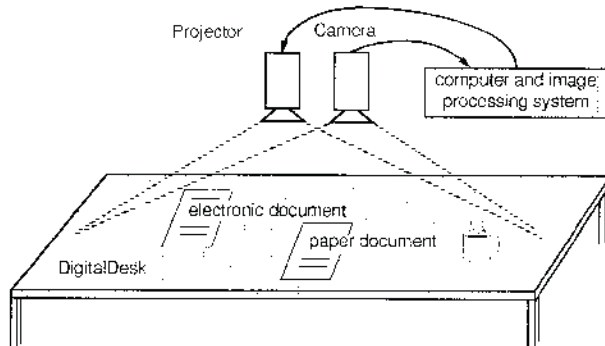


Figure 1-10. A diagram of the Digital Desk system.



Figure 1-11. Simon, released in 1994, was the first mobile touchscreen device. It suffered from some serious design flaws, such as not being able to show more than a few keyboard keys simultaneously, but it was a decade ahead of its time. Courtesy IBM.

More than a decade before Apple released the iPhone (and other handset manufacturers such as LG, Sony Ericsson, and Nokia released similar touchscreen phones as well), IBM and Bell South launched Simon, a touchscreen mobile phone. It was ahead of its time and never caught on, but it demonstrated that a mobile touchscreen could be manufactured and sold.

In the late 1990s and the early 2000s, touchscreens began to make their way into wide public use via retail kiosks, public information displays, airport check-in services, transportation ticketing systems, and new ATMs.

* Wellner, Pierre. "The DigitalDesk Calculator: Tactile Manipulation on a Desktop Display." Proceedings of the Fourth Annual Symposium on User Interface Software and Technology (UIST): 27–33, 1991.

† Watch the video demonstration at <http://video.google.com/videoplay?docid=5772530828816089246>.



Figure 1-12. Antenna Design's award-winning self-service check-in kiosk for JetBlue Airlines. Courtesy JetBlue and Antenna Design.



Figure 1-13. The Essential Reality P5 Glove is likely the first commercial controller for gestural interfaces, for use with the game *Black & White*. Courtesy Lionhead Studios.

Lionhead Studios released what is likely the first home gaming gestural interface system in 2001 with its game, *Black & White*. A player controlled the game via a special glove that, as the player gestured physically, would be mimicked by a digital hand on-screen. In arcades in 2001, Konami's *MoCap Boxing* game had players put on boxing gloves and stand in a special area monitored with infrared motion detectors, then "box" opponents by making movements that actual boxers would make.

The mid-2000s have simply seen the arrival of gestural interfaces for the mass market. In 2006, Nintendo released its Wii gaming system. In 2007, to much acclaim, Apple launched its iPhone and iPod Touch, which were the first touchscreen devices to receive widespread media attention and television advertising demonstrating their touchscreen capabilities. In 2008, handset manufacturers such as LG, Sony Ericsson, and Nokia released their own touchscreen mobile devices. Also in 2008, Microsoft launched MS Surface, a large, table-like touchscreen that is used in commercial spaces for gaming and retail display. And Jeff Han now manufactures his giant touchscreens for government agencies and large media companies such as CNN.

GEEKY STUFF YOU MIGHT CARE ABOUT

What's the Deal with Interactive Gestures and Public Restrooms?

If you have visited a public bathroom in the past several years, you've likely encountered interactive gestures that control everything from washing your hands to flushing the toilet. What gives? Why have trips to the bathroom become visits to interaction design labs?

There are a few possible reasons. The first is that bathrooms are places where bacteria live, and people have grown increasingly hesitant to touch things there. Thus, gestures allow us to interact with the environment without requiring much physical contact.

The second reason is simply one of maintenance and conservation. Bathrooms require a lot of upkeep, and having toilets automatically flush, paper towels dispense only a small amount, and sinks automatically shut off saves both maintenance and resources (read: money).

The third reason is one of human nature. As Arthur C. Clarke noted, "Any sufficiently advanced technology is indistinguishable from magic," and interactive gestures are quite magical right now. And we may need magic in our bathrooms—they can be places of anxiety and discomfort. Anthropologist Bronislaw Malinowski in his "Theory of the Gap" claimed that humans used magic to reduce anxiety, to overcome unpredictability and uncertainty. So, perhaps it makes perfect sense to have magic in our bathrooms.

Whatever the reason, public restrooms are the new labs for interactive gestures.

The future (see Chapter 8) should be interesting.

THE MECHANICS OF TOUCHSCREENS AND GESTURAL CONTROLLERS

Even though forms of gestural devices can vary wildly—from massive touchscreens to invisible overlays onto environments—every device or environment that employs gestures to control it has at least three general parts: a **sensor**, a **comparator**, and an **actuator**. These three parts can be a single physical component, or, more typically, multiple components of any gestural system, such as a motion detector (a sensor), a computer (the comparator), and a motor (the actuator).

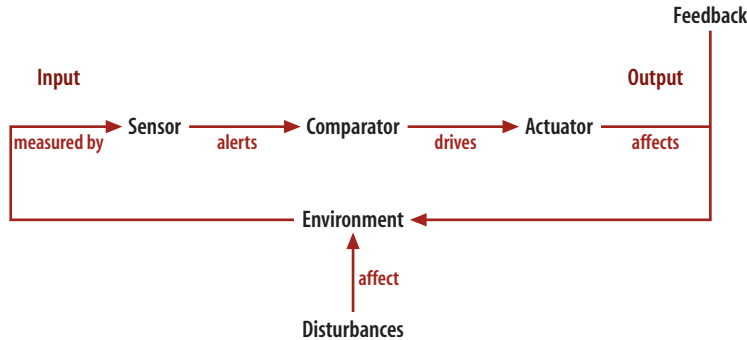


Figure 1-14. *The basic components of any gestural system*

A **sensor** is typically an electrical or electronic component whose job is to detect changes in the environment. These changes can be any number of things, depending on the type of sensor, of which there are many.* The most common types currently used for interactive gestures are:

Pressure

To detect whether something is being pressed or stood on. This is often mechanical in nature.

Light

To detect the presence of light sources (also called a *photodetector*). This is used mostly in environments, especially in lighting systems.

Proximity

To detect the presence of an object in space. This can be done in any number of ways, from infrared sensors to motion and acoustic sensors.

Acoustic

To detect the presence of sound. Typically, this is done with small microphones.

Tilt

To detect angle, slope, and elevation. Tilt sensors generate an artificial horizon and then measure the incline with respect to that horizon.

Motion

To detect movement and speed. Some common sensors use microwave or ultrasonic pulses that measure when a pulse bounces off a moving object (which is how radar guns catch you speeding).

* <http://en.wikipedia.org/wiki/Sensor> has a more complete list, including many sensors that are currently used only for scientific or industrial applications.

Orientation

To detect position and direction. These are often used in navigation systems currently, but position within environments could become increasingly important and would need to be captured by cameras, triangulating proximity sensors, or even GPSes in the case of large-scale use.

It's no exaggeration to state that the type of sensor you employ entirely determines the types of gestural interactions that are possible. If the system can't detect what a user is doing, those gestures might as well not be happening. I can wave at my laptop as much as I want, but if it doesn't have any way to detect my motion, I simply look like an idiot.

It is crucially important to calibrate the sensitivity of the sensor (or the moderation of the comparator). A sensor that is too sensitive will trigger too often and, perhaps, too rapidly for humans to react to. A sensor that is too dull will not respond quickly enough, and the system will seem sluggish or nonresponsive.

The size (for touchscreens) or coverage area of the sensor is also very important, as it determines what kinds of gestures (broad or small, one or two hands, etc.) are appropriate or even possible to have.



The larger the sensor coverage area is, the broader the gesture possible.

Often in more complex systems, multiple sensors will work together to allow for more nuanced movement and complicated gesture combinations. (To have 3D gestures, multiple sensors are a necessity to get the correct depth.) Many Apple products (including Apple laptops) have accelerometers to detect speed and motion built into them, as do Wii controllers. But accelerometers are tuned to themselves, not to the environment, so they alone can't determine the user's position in the room or the direction the user is facing, only whether the device is moving and the direction and speed at which it is moving. For orientation within an environment or sophisticated detection of angles, other sensors need to be deployed. The Wii, for instance, deploys both accelerometers and gyroscopes within its controllers for tilt and motion detection, and an infrared sensor that communicates to the "sensor bar" to indicate orientation* for a much wider range of possible gestures.

* See "At the Heart of the Wii Micron-Size Machines," by Michel Marriott, in *The New York Times*, December 21, 2006; <http://www.nytimes.com/2006/12/21/technology/21howw.html?partner=permalink&exprod=permalink>.

GEEKY STUFF YOU MIGHT CARE ABOUT

Touch Events and Touchscreen Sensors

A **touch event** is the technical term for when a touchscreen system knows that a user has touched the screen. Touch events are a combination of the sensor and the comparator, but the technology the system uses to detect a touch varies.

In most modern touchscreen interfaces, the sensor is a touch-responsive glass panel that usually employs one of three technologies: resistive, surface wave, or capacitive. *Resistive systems* are made up of two layers. When a user touches the top layer, the two layers press together, triggering a touch event. Because of how resistive systems work, they require pressure (and can measure it well), but multitouch does not work very well (if at all). *Surface wave systems* generate ultrasonic waves. When a user touches the screen, a portion of the wave is absorbed, and that registers as a touch event. *Capacitive sensor panels* are coated with a material that stores electrical charge. When a user touches the screen (or, in some cases, even hovers over the screen), a portion of the charge is transferred to the user, decreasing the panel's capacitive layer and thus triggering a touch event.

Another method, particularly for large displays, incorporates *infrared beams* that skim the (flat) surface of a screen in a grid-like matrix. When an object, such as a user's finger, touches the screen, it breaks the beams, and the *X* and *Y* positions of the object can be calculated. Sensitivity is determined by how close the beams are to each other.

Infrared light can also be used in *frustrated total internal reflection (FTIR)* and *diffused illumination (DI)* systems, which use infrared cameras to determine touch events, especially multitouch events.* With an FTIR setup, an infrared camera is placed below a sheet of Plexiglas and an infrared light is shone into the side of the Plexiglas (most often by shining IR LEDs onto the sides of the acrylic). The light reflects around the inside of the Plexiglas unhindered. But when a finger touches the surface, this light is “frustrated,” causing the light to scatter downward where the camera can detect it, resulting in a touch event.

There are two different kinds of DI systems: rear and front, depending on the direction in which the infrared light is being projected. With rear DI, infrared lights shine upward at a clear surface (glass, acrylic, Plexiglas, etc.) from below (i.e., from the rear of the surface). A diffuser (a device that diffuses, spreads out, or scatters light in some manner to create a soft light) is placed on top of or underneath the touch surface. When an object, such as a finger, touches the surface (creating what are known as *blobs*), it reflects more light than the diffuser or objects in the background, and the camera below senses this extra light, creating a touch event. Depending on the diffuser, rear DI can also detect objects hovering over the surface.

Front DI is when the infrared light is projected on top of the surface from above (i.e., from the front). As with rear DI, a diffuser used to soften the light is placed on the top or bottom of the surface. When an object, such as a finger, touches the surface, it makes a shadow in the position of the object that the camera positioned below detects and uses to determine a touch event.

* For a more detailed view of FTIR and DI systems, read “Getting Started with MultiTouch” at <http://nuigroup.com/forums/viewthread/1982/>.

Once a sensor detects its target, it passes the information on to what is known in systems theory as a **comparator**. The comparator compares the current state to the previous state or the goal of the system and then makes a judgment. For many gestural interfaces, the comparator is a microprocessor running software, which decides what to do about the data coming into the system via the sensor. Certainly, there are all-mechanical systems that work with gestures, but they tend toward cruder on/off scenarios, such as lights that come on when someone walks past them. Only with a microprocessor can you design a system with much nuance, one that can make more sophisticated decisions.

Those decisions get passed on to an **actuator** in the form of a command. Actuators can be analog or mechanical, similar to the way the machinery of The Clapper turns lights on; or they can be digital, similar to the way tilting the iPhone changes its screen orientation from portrait to landscape. With mechanical systems, the actuator is frequently a small electric motor that powers a physical object, such as a motor to open an automatic door. As with the comparator's decision making, for digital systems, it is software that drives the actuator. It is software that determines what happens when a user touches the screen or extends an arm.

Of course, software doesn't design and code itself, and sensors and motors and the like aren't attached randomly to systems. They need to be designed.

DESIGNING INTERACTIVE GESTURES: THE BASICS

The design of any product or service should start with the needs of those who will use it, tempered by the constraints of the environment, technology, resources, and organizational goals, such as business objectives. The needs of users can range from simple (I want to turn on a light) to very complex (I want to fall in love). (Most human experience lies between those two poles, I think.) However natural, interesting, amusing, novel, or innovative an interactive gesture is, if the users' needs aren't met, the design is a failure.

The first question that anyone designing a gestural interface should ask is: should this even be a gestural interface? Simply because we can now do interactive gestures doesn't mean they are appropriate for every situation. As Bill Buxton notes,^{*} when it comes to technology, everything is best for something and worse for something else, and interactive gestures are no exception.

There are several reasons to *not* have a gestural interface:

^{*} See Bill Buxton's multitouch overview at <http://www.billbuxton.com/multitouchOverview.html>.

Heavy data input

Although some users adapt to touchscreen keyboards easily, a keyboard is decidedly faster for most people to use when they are entering text or numbers.

Reliance on the visual

Many gestural interfaces use visual feedback alone to indicate that an action has taken place (such as a button being pressed). In addition, most touchscreens and many gestural systems in general rely entirely on visual displays with little to no haptic affordances or feedback. There is often no physical feeling that a button has been pressed, for instance. If your users are visually impaired (as most adults over a certain age are) a gestural interface may not be appropriate.

Reliance on the physical

Likewise, gestural interfaces can be more physically demanding than a keyboard/screen. The broader and more physical the gesture is (such as a kick, for instance), the more likely that some people won't be able to perform the gesture due to age, infirmity, or simply environmental conditions; pressing touchscreen buttons in winter gloves is difficult, for instance. The inverse is also true: the subtler and smaller the movement, the less likely everyone will be able to perform it. The keyboard on the iPhone, for instance, is entirely too small and delicate to be used by anyone whose fingers are large or otherwise not nimble.

Inappropriate for context

The environment can be nonconductive to a gestural interface in any number of situations, either due to privacy reasons or simply to avoid embarrassing the system's users. Designers need to take into account the probable environment of use and determine what, if any, kind of gesture will work in that environment.

There are, of course, many reasons to use a gestural interface. Everything that a noninteractive gesture can be used for—communication, manipulating objects, using a tool, making music, and so on—can also be done using an interactive gesture. Gestural interfaces are particularly good for:

More natural interactions

Human beings are physical creatures; we like to interact directly with objects. We're simply wired this way. Interactive gestures allow users to interact naturally with digital objects in a physical way, like we do with physical objects.

Less cumbersome or visible hardware

With many gestural systems, the usual hardware of a keyboard and a mouse isn't necessary: a touchscreen or other sensors allow users to perform actions without this hardware. This benefit allows for gestural interfaces to be put in places where a traditional computer configuration would be impractical or out of place, such as in retail stores, museums, airports, and other public spaces.



Figure 1-15. New York City in late 2006 installed touchscreens in the back seats of taxicabs. Although clunky, they allow for the display of interactive maps and contextual information that passengers might find useful, such as a Zagat restaurant guide. Courtesy New York City Taxi and Limousine Commission.

More flexibility

As opposed to fixed, physical buttons, a touchscreen, like all digital displays, can change at will, allowing for many different configurations depending on functionality requirements. Thus, a very small screen (such as those on most consumer electronics devices or appliances) can change buttons as needed. This can have usability issues (see later in this chapter), but the ability to have many controls in a small space can be a huge asset for designers. And with nontouchscreen gestures, the sky is the limit, space-wise. One small sensor, which can be nearly invisible, can detect enough input to control the system. No physical controls or even a screen are required.

More nuance

Keyboards, mice, trackballs, styli, and other input devices, although excellent for many situations, are simply not as able to convey as much subtlety as the human body. A raised eyebrow, a wagging finger, or crossed arms can deliver a wealth of meaning in addition to controlling a tool. Gestural systems have not begun to completely tap the wide emotional palette of humans that they can, and likely will, eventually exploit.

More fun

You can design a game in which users press a button and an on-screen avatar swings a tennis racket. But it is simply more entertaining—for both players and observers—to mimic swinging a tennis racket physically and see the action mirrored on-screen. Gestural systems encourage play and exploration of a system by providing a more hands-on (sometimes literally hands-on) experience.

Once the decision has been made to have a gestural interface, the next question to answer is what kind of gestural interface it will be: direct, indirect, or hybrid. As I write this, particularly with devices and appliances, the answer will be fairly easy: a direct-manipulation touchscreen is the most frequently employed gestural interface currently. In the future, as an increasing variety of sensors are built into devices and environments, this may change, but for now touchscreens are the new standard for gestural interfaces.

THE CHARACTERISTICS OF GOOD GESTURAL INTERFACES

Although particular aspects of gestural systems require more and different kinds of consideration, the characteristics of a good gestural interface don't differ much from the characteristics of any other well-designed interactive system.* Designers often use Liz Sanders' phrase "useful, usable, and desirable"[†] to describe well-designed products, or they say that products should be "intuitive" or "innovative." All of that really means gestural interfaces should be:

Discoverable

Being discoverable can be a major issue for gestural interfaces. How can you tell whether a screen is touchable? How can you tell whether an environment is interactive? Before we can interact with a gestural system, we have to know one is there and how to begin to interact with it, which is where **affordances** come into play. An affordance is one or multiple properties of an object that give some indication of how to interact with that object or a feature on that object. A button, because of how it moves, has an affordance of pushing. Appearance and texture are the major sources of what psychologist James Gibson called affordances,[‡] popularized in the design community by Don Norman in his seminal 1988 book *The Psychology of Everyday Things* (later renamed *The Design of Everyday Things*).

* For a longer discussion, see *Designing for Interaction* by Dan Saffer (Peachpit Press): 60–68.

† See "Converging Perspectives: Product Development Research for the 1990s," by Liz Sanders, in *Design Management Journal*, 1992.

‡ Gibson, J.J. "The theory of affordances," in *Perceiving, Acting, and Knowing: Toward an Ecological Psychology*, R. Shaw and J. Bransford (Eds.) (Lawrence Erlbaum): 67–82.



Figure 1-16. Without the tiny diagrams on the dispenser, there would be no affordances to let you know how to get the toilet paper out. Gestural interfaces need to be discoverable so that they can be used. Courtesy Yu Wei Products Company.

Trustworthy

Unless they are desperate, before users will engage with a device, the interface needs to look as though it isn't going to steal their money, misuse their personal data, or break down. Gestural interfaces have to appear competent and safe, and they must respect users' privacy (see "The Ethics of Gestures" in Chapter 8). Users are also now suspicious of gestural interfaces and often an **attraction affordance** needs to be employed (see Chapter 7).

Responsive

We're used to instant reaction to physical manipulation of objects. After all, we're usually touching things that don't have a microprocessor and sensor that need to figure out what's going on. Thus, responsiveness is incredibly important. When engaged with a gestural interface, users want to know that the system has heard and understood any commands given to it. This is where **feedback** comes in. Every action by a human directed toward a gestural interface, no matter how slight, should be accompanied by some acknowledgment of the action whenever possible and as rapidly as possible (100 ms or less is ideal as it will feel instantaneous). This can be tricky, as the responsiveness of the system is tied directly to the responsiveness of the system's sensors, and sensors that are too responsive can be even more irksome than those that are dull. Imagine if The Clapper picked up every slight sound and turned the lights on and off, on and off, over and over again! But not having near-immediate feedback can cause errors, some of them potentially serious. Without any response, users will often repeat an action they

just performed, such as pushing a button again. Obviously, this can cause problems, such as accidentally buying an item twice or, if the button was connected to dangerous machinery, injury or death. If a response to an action is going to take significant time (more than one second), feedback is required that lets the user know the system has heard the request and is doing something about it. Progress bars are an excellent example of responsive feedback: they don't decrease waiting time, but they make it seem as though they do. They're responsive.

Appropriate

Gestural systems need to be appropriate to the culture, situation, and context they are in. Certain gestures are offensive in certain cultures. An “okay” gesture, commonplace in North America and Western Europe, is insulting in Greece, Turkey, the Middle East, and Russia, for instance.* An overly complicated gestural system that involves waving arms and dancing around in a public place is not likely to be an appropriate system unless it is in a nightclub or other performance space.

Meaningful

The coolest interactive gesture in the world is empty unless it has meaning for the person performing it; which is to say, unless the gestural system meets the needs of those who use it, it is not a good system.

Smart

The devices we use have to do for us the things that we as humans have trouble doing—rapid computation, having infallible memories, detecting complicated patterns, and so forth. They need to remember the things we don't remember and do the work we can't easily do alone. They have to be smart.

Clever

Likewise, the best products predict the needs of their users and then fulfill those needs in unexpectedly pleasing ways. Adaptive targets are one way to do this with gestural interfaces. Another way to be clever is through interactive gestures that match well the action the user is trying to perform.

Playful

One area in which interactive gestures excel is being playful. Through play, users will not only start to engage with your interface—by trying it out to see how it works—but they will also explore new features and variations on their gestures. Users need to feel relaxed to engage in play. Errors need to be difficult to make so that there is no need to put warning messages all over the interface. The ability to undo mistakes is also crucial for fostering the environment for play. Play stops if users feel trapped, powerless, or lost.

* See *Field Guide to Gestures*, by Nancy Armstrong and Melissa Wagner (Quirk Books): 45–48.

Pleasurable

“Have nothing in your house,” said William Morris, “that you do not know to be useful, or believe to be beautiful.” Gestural interfaces should be both aesthetically and functionally pleasing. Humans are more forgiving of mistakes in beautiful things.* The parts of the gestural system—the visual interface; the input devices; the visual, aural, and haptic feedback—should be agreeable to the senses. They should be pleasurable to use. This engenders good feelings in their users.

Good

Gestural interfaces should have respect and compassion for those who will use them. It is very easy to remove human dignity with interactive gestures—for instance, by making people perform a gesture that makes them appear foolish in public, or by making it so difficult to perform a gesture that only the young and healthy can ever perform it. Designers and developers need to be responsible for the choices they make in their designs and ask themselves whether it is good for users, good for those indirectly affected, good for the culture, and good for the environment. The choices that are made with gestural interfaces need to be deliberate and forward-thinking. Every time users perform an interactive gesture, in an indirect way they are placing their trust in those who created it to have done their job ethically.

THE ATTRIBUTES OF GESTURES

Although touchscreen gestural interfaces differ slightly from free-form gestural interfaces, most gestures have similar characteristics that can be detected and thus designed for. The more sophisticated the interface (and the more sensors it employs), the more of these attributes can be engaged:

Presence

This is the most basic of all attributes. Something must be present to make a gesture in order to trigger an interaction. For some systems, especially in environments, a human being simply being present is enough to cause a reaction. For the simplest of touchscreens, the presence of a fingertip creates a touch event.

Duration

All gestures take place over time and can be done quickly or slowly. Is the user tapping a button or holding it down for a long period? Flicking the screen or sliding along it? For some interfaces, especially those that are simple, duration is less important. Interfaces using proximity sensors, for instance, care little for duration and only whether a human being is in the area. But for games and other types of interfaces, the ability to determine duration is crucial.

* See Don Norman's book, *Emotional Design* (Basic Books), for a detailed discussion of this topic.

Duration is measured by calculating the time of first impact or sensed movement compared to the end of the gesture.

Position

Where is the gesture being made? From a development standpoint, position is often determined by establishing an *x/y* location on an axis (such as the entire screen) and then calculating any changes. Some gestures also employ the *z*-axis of depth. Note that because of human beings' varying heights, position can be relational (related to the relative size of the person) or exact (adjusted to the parameters of the room). For instance, a designer may want to put some gestures high in an environment so that children cannot engage in them.

Motion

Is the user moving from position to position or striking a pose in one place? Is the motion fast or slow? Up and down, or side to side? For some systems, any motion is enough to trigger a response; position is unnecessary to determine.

Pressure

Is the user pressing hard or gently on a touchscreen or pressure-sensitive device? This too has a wide range of sensitivity. You may want every slight touch to register, or only the firmest, or only an adult weight (or only that of a child or pet). Note that some pressure can be "faked" by duration; the longer the press/movement, the more "pressure" it has. Pressure can also be faked by trying to detect an increasing spread of a finger pad: as we press down, the pad of our finger widens slightly as it presses against a surface.

Size

Width and height can also be combined to measure size. For example, touchscreens can determine whether a user is employing a stylus or a finger based on size (the tip of a stylus will be finer) and adjust themselves accordingly.

Orientation

What direction is the user (or the device) facing while the gesture is being made? For games and environments, this attribute is extremely important. Orientation has to be determined using fixed points (such as the angle of the user to the object itself).

Including objects

Some gestural interfaces allow users to employ physical objects alongside their bodies to enhance or engage the system. Simple systems will treat these other objects as an extension of the human body, but more sophisticated ones will recognize objects and allow users to employ them in context.

For instance, a system could see a piece of paper a user is holding as being simply part of the user's hand, whereas another system, such as the Digital Desk system (see Figure 1-12, earlier in this chapter), might see it as a piece of paper that can have text or images projected onto it.

Number of touch points/combination

More and more gestural interfaces have multitouch capability, allowing users to use more than one finger or hand simultaneously to control them. They may also allow combinations of gestures to occur at the same time. One common example is using two hands to enlarge an image by dragging on two opposite corners, seemingly stretching the image.

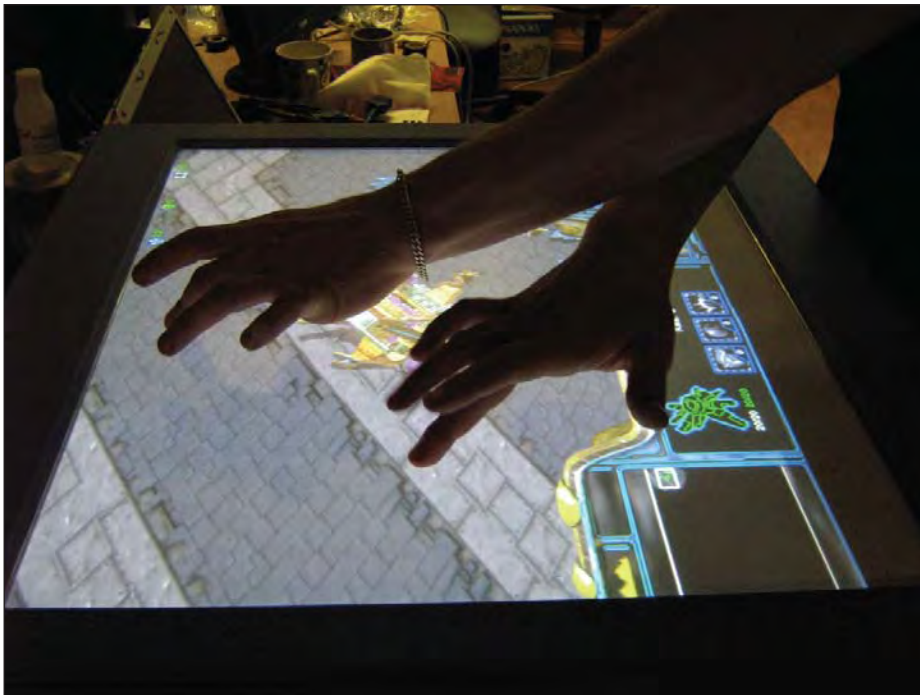


Figure 1-17. Designers experimenting with a multitouch system to play Starcraft with two hands. Courtesy Harry van der Veen and Natural User Interface.

Sequence

Interactive gestures don't necessarily have to be singular. A wave followed by a fist can trigger a different action than both of those gestures done separately. Of course, this means a very sophisticated system that remembers states. This is also more difficult for users (see "States and Modes," later in this chapter).

Number of participants

It can be worthwhile with some devices—such as Microsoft’s Surface, which is meant to be used socially for activities such as gaming or collaborative work—to detect multiple users. Two people operating a system using one hand each is very different from one person operating a system using both hands.

When designing a particular interactive gesture, these attributes, plus the range of physical movement (see Chapter 2), should be considered. Of course, simple gestural interfaces, such as most touchscreens, will use only one or two of these characteristics (presence and duration being the most common), and designers and developers may not need to dwell overly long on the attributes of the gesture but instead on the ergonomics and usability of interactive gestures (see Chapter 2).

INTERFACE CONVENTIONS

Many of the traditional interface conventions work well in gestural interfaces: selecting, drag-and-drop, scrolling, and so on. There are several notable exceptions to this:

Cursors

With gestural interfaces, a cursor is often unnecessary since a user isn’t consistently pointing to something; likewise, a user’s fingers rarely trail over the touchscreen where a cursor would be useful to indicate position. Users don’t often lose track of their fingers! Of course, for gaming, a cursor is often absolutely essential to play, but this is usually on free-form gestural interfaces, not touchscreens.

Hovers and mouse-over events

For the same reason that cursors aren’t often employed, hovers and mouse-over events are also seldom used, except in some free-form games and in certain capacitive systems. Nintendo’s Wii, for instance, often includes a slight haptic buzz as the user roles over selectable items. Some sensitive capacitive touchscreens can detect a hand hovering over the screen, but hovers need to be aware of *screen coverage* (see Chapter 2).

Double-click

Although a double click can be done with a gestural interface, it should be used with caution. A threshold has to be set (e.g., 200 ms) during which two touch events in the same location are counted as a double click. The touchscreen has to be sensitive and responsive enough to register touch-rest-touch. Single taps to click are safer to use (see “Tap to Open/Activate” in Chapter 3).

Right-click

Most gestural interfaces don't have the ability to bring up an alternative menu for objects. The direct-manipulation nature of most gestural interfaces tends to go against this philosophically. This is not to say that digital objects could not display a menu when selected, just that they frequently avoid this traditional paradigm.

Drop-down menus

These generally don't work very well for the same reasons as right-click menus, combined with the limitations of hover.

Cut-and-paste

As of this writing, cut-and-paste is only partially implemented or theorized on most gestural interfaces. It will likely be implemented shortly, but as of summer 2008, it has not been on most common gestural interfaces.

Multiselect

As humans, we're limited by the number of limbs and fingers we have to select multiple items on a screen or a list. There are ways around this, such as a select mode that could be turned on so that everything on-screen, once selected, remains selected; alternately, an area could be "drawn" that selects multiple items.

Selected default buttons

Since pressing a return key (and thus pushing a selected button) isn't typically part of a gestural system, all a selected default button can do is highlight probable behavior. Users will have to make an interactive gesture (e.g., pushing a button) no matter what to trigger an action.

Undo

It's hard to undo a gesture; once a gesture is done, typically the system has executed the command, and there is no obvious way, especially in environments, to undo that action. It is better to design an easy way to cancel or otherwise directly undo an action (e.g., dragging a moved item back) than it is to rely on undo.

Assuredly, there are exceptions to all of these interface constraints, and clever designers and developers will find ways to work around them. For the new conventions that have been established with gestural interfaces, see Chapters 3 and 4.

STATES AND MODES

Most gestural interfaces are stateless or modeless, which is to say that there is only one major function or task path for the system to accomplish at any given time. An airport kiosk assists users in checking in, for instance. The Clapper turns lights on or off. Users don't switch between "modes" (as, say, between writing and editing modes), or if they do, they do so only in complex devices such as mobile phones.

The reason for this is both contextual and related to the nature of interactive gestures. Gestural interfaces are often found in public spaces, where attention is limited and simplicity and straightforwardness are appreciated; this is combined with the fact that—especially with free-form interfaces—there might be no visual indicator (i.e., no screen or display) to convey what mode the user is in. Thus, doing a gesture to change to another mode may accomplish the task, but how does the user know it was accomplished? And how does the user return to the previous mode? By performing the same gesture? Switching between states is a difficult interaction design problem for gestural interfaces.

Thus, it is considerably easier for users (although not for the designers) to have either clear paths through the gestural system or a single set of choices to execute. For example, a retail kiosk might be designed to help with the following tasks: searching for an item, finding an item in the store, and purchasing an item. It is better to have these activities in a clear path (search to find to buy) than to require users to switch to different modes to execute an action, such as buying.

As users become more sophisticated and gestural interfaces more ubiquitous, this may change, but for now, a stateless design is usually the better design.

DETERMINING THE APPROPRIATE GESTURE

Once you've decided that a gestural interface is appropriate for your users and your environment, you need to pair the appropriate gestures to the tasks and goals the users need to accomplish. This requires a combination of three things: the available sensors and related input devices, the steps in the task, and the physiology of the human body. Sensors determine what the system can detect and how. The steps in the task show what actions have to be performed and what decisions have to be made. The human body provides physical constraints for the gestures that can be done (see Chapter 2).

For most touchscreens, this can be a very straightforward equation. There is one sensor/input device (the touchscreen); the tasks (check in, buy an item, find a location, get information) are usually simple. The touchscreen needs to be accessible and used by a wide variety of people of all ages. Thus, simple gestures such as pushing buttons are appropriate.



The complexity of the gesture should match the complexity of the task at hand.

This is to say that simple, basic tasks should have equally simple, basic gestures to trigger or complete them, for instance, taps, swipes, and waves. More complicated tasks may have more complicated gestures.

Take, for example, turning on a light. If you just want to turn on a light in a room, a wave or swipe on a wall should be sufficient for this simple behavior. Dimming the light (a slightly more complex action), however, may require a bit more nuance, such as holding your hand up and slowly lowering it. Dimming all the lights in the house at once (a sophisticated action) may require a combination gesture or a series of gestures, such as clapping your hands three times, then lowering your arm. Because it is likely seldom done and is conceptually complicated, it can have an equally complex associated movement.

This is not to say that all complex behaviors need to or should have accompanying complex gestures—quite the opposite, in fact—only that simple actions should not require complex actions to initiate. The best interactive gestures are those that take the complex and make them simple and elegant.

One way to do this, especially with touchscreen devices, is to make all the features accessible with simple gestures such as taps (via a menu system, say), and then to provide alternative gestures that are more sophisticated (but faster) for more advanced users. In this way, an interactive gesture can act as a shortcut to features in much the same way as a key command works on desktop systems. Of course, communicating this advanced gesture then becomes an issue to address (see Chapter 7).

USING PEOPLE TO DETERMINE THE APPROPRIATE GESTURE

Rather than have the designer determine the gestures of the system, another method for determining the appropriate action for a gesture is to employ the knowledge and intuition of those who will use it. You can ask the users to match a feature to the gesture they would like to use to employ it. Asking several users will hopefully begin to show patterns matching gestures to features. The reverse of this would be to demonstrate a gesture and see what feature users would expect that gesture to trigger.

MATCHING THE GESTURE TO THE BEHAVIOR

Japanese product designer Naoto Fukasawa has observed that the best designs are those that “dissolve in behavior,”^{*} meaning that the products themselves disappear into whatever the user is doing. It’s seemingly effortless (although certainly not for those creating this sort of frictionless system—intuitive, natural designs require significant effort) and a nearly subconscious act to use the product to accomplish what you want to do. This is the promise of interactive gestures in general: that we’ll be able to empower the gestures *that we already do* and give them further influence and meaning.

Adam Greenfield, author of *Everyware*, talked about this type of natural interaction in an interview:[†]

“We see this, for example, in Hong Kong where women leave their RFID-based Octopus cards in their handbags and simply swing their bags across the readers as they move through the turnstiles. There’s a very sophisticated transaction between card and reader there, but it takes 0.2 seconds, and it’s been subsumed entirely into this very casual, natural, even jaunty gesture.

“But that wasn’t designed. It just emerged; people figured out how to do that by themselves, without some designer having to instruct them in the nuances. . . . The more we can accommodate and not impose, the more successful our designs will be.”

The best, most natural designs, then, are those that match the behavior of the system to the gesture humans might already do to enable that behavior. Simple examples include pushing a button to turn something on or off, turning to the left to make your on-screen avatar turn to the left, putting your hands under a sink to turn the water on, and passing through a dark hallway to illuminate it.

The design dissolves into the behavior.

In the next chapter, we’ll look at an important piece of the equation when designing gestural interfaces: the human body.

^{*} See, for instance, *Dwell* magazine’s interview with Fukasawa, “Without a Trace,” by Jane Szita, September 2006, which you can find at <http://www.dwell.com/peopleplaces/profiles/3920931.html>.

[†] *Designing for Interaction* by Dan Saffer, p. 217.

FOR FURTHER READING

Where the Action Is, Paul Dourish (MIT Press)

Everyware, Adam Greenfield (New Riders Publishing)

The Ecological Approach to Visual Perception, James Gibson (Lawrence Erlbaum)

Designing Interactions, Bill Moggridge (MIT Press)

The Design of Everyday Things, Don Norman (Basic Books)

Emotional Design, Don Norman (Basic Books)

Designing for Interaction, Dan Saffer (Peachpit Press)

Designing Interfaces, Jenifer Tidwell (O'Reilly)