Multi-Touch Systems that I Have Known and Loved

Bill Buxton
Microsoft Research
Original: Jan. 12, 2007
Version: June 12, 2014

Keywords / Search Terms

Multi-touch, multitouch, input, interaction, touch screen, touch tablet, multi-finger input, multi-hand input, bi-manual input, two-handed input, multi-person input, interactive surfaces, soft machine, hand gesture, gesture recognition .

An earlier version of this page is also available in Belorussian, thanks to the translation by Martha Ruszkowski.

Preamble

Since the announcements of the *iPhone* and Microsoft*'s Surface* (both in 2007), an especially large number of people have asked me about multi-touch. The reason is largely because they know that I have been involved in the topic for a number of years. The problem is, I can't take the time to give a detailed reply to each question. So I have done the next best thing (I hope). That is, start compiling my would-be answer in this document. The assumption is that ultimately it is less work to give one reasonable answer than many unsatisfactory ones.

Multi-touch technologies have a long history. To put it in perspective, my group at the University of Toronto was working on multi-touch in 1984 (Lee, Buxton & Smith, 1985), the same year that the first Macintosh computer was released, and we were not the first. Furthermore, during the development of the iPhone, Apple was very much aware of the history of multi-touch, dating at least back to 1982, and the use of the pinch gesture, dating back to 1983. This is clearly demonstrated by the bibliography of the PhD thesis of Wayne Westerman, co-founder of FingerWorks, a company that Apple acquired early in 2005, and now an Apple employee

- Westerman, Wayne (1999). *Hand Tracking,Finger Identification, and Chordic Manipulation on a Multi-Touch Surface.* U of Delaware PhD Dissertation: http://www.ee.udel.edu/~westerma/main.pdf

In making this statement about their awareness of past work, I am not criticizing Westerman, the iPhone, or Apple. It is simply good practice and good scholarship to know the literature and do one's homework when embarking on a new product. What I *am* pointing out, however, is that "new" technologies - like multi-touch - do not grow out of a vacuum. While marketing tends to like the "great invention" story, real innovation rarely works that way. In short, the evolution of multi-touch is a text-book example of what I call "the long-nose of innovation."

So, to shed some light on the back story of this particular technology, I offer this brief and incomplete summary of some of the landmark examples that I have been involved with, known about and/or encountered over the years. As I said, it is incomplete and a work in progress (so if you come back a second time, chances are there will be more and better information). I apologize to those that I have missed. I have erred on the side of timeliness *vs* thoroughness. Other work can be found in the references to the papers that I do include.

Note: for those note used to searching the HCI literature, the primary portal where you can search for and download the relevant literature, including a great deal relating to this topic (including the citations in the Westerman thesis), is the ACM Digital Library: http://portal.acm.org/dl.cfm. One other relevant source of interest, should you be interested in an example of the kind of work that has been done studying gestures in interaction, see the thesis by Hummels:

- http://id-dock.com/pages/overig/caro/publ_caro.htm

While not the only source on the topic by any means, it is a good example to help gauge what might be considered new or obvious.

Please do not be shy in terms of sending me photos, updates, etc. I will do my best to integrate them.

For more background on input, see also the incomplete draft manuscript for my book on input tools, theories and techniques:

For more background on input devices, including touch screens and tablets, see my directory at:

- http://www.billbuxton.com/InputSources.html

I hope this helps.

Some Dogma

There is a lot of confusion around touch technologies, and despite a history of over 25 years, until relatively recently (2007), few had heard of multi-touch technology, much less used it. So, given how much impach it is having today, how is it that multi-touch took so long to take hold?

1. It took 30 years between when the mouse was invented by Engelbart and English in 1965 to when it became ubiquitous, on the release of Windows 95. Yes, a mouse was shipped commercially as early as 1968 with a German computer from Telefunken, and more visibly on the Xerox Star and PERQ workstations in 1982. Speaking personally, I used my first mouse in 1972 at the National Research Council of Canada. Yet, none of this made a huge dent in terms of the overal number deployed. It took 30 years to hit the tipping point. By that measure, multi-touch technologies, multi-touch got traction 5 years faster than the mouse!
2. One of my primary axioms is: *Everything is best for something and worst for something else.* The trick is knowing what is what, for what, when, for whom, where, and most importantly, why. Those who try the replace the mouse play a fool's game. The mouse is great for many things. Just not everything.The challenge with new input is to find devices that work together, simultaneously with the mouse (such as in the other hand), or things that are strong where the mouse is weak, thereby complimenting it.
3. A single new technology, no matter how potentially useful, is seldom the cause of a product's overall success. As with the mouse and multi-touch, a whole new ecosystem was required before their full potential could begin to be exploited.
4. Arguably, input techniques and technologies have played second-fiddle relative to displays, in terms of investment and attention. The industry seemed content to try and make a better mouse, or mouse replacement (such as a trackball or joystick), rather than change the overall paradigm of interaction.

Some Framing

I don't have time to write a treatise, tutorial or history. What I can do is warn you about a few traps that seem to cloud a lot of thinking and discussion around this stuff. The approach that I will take is to draw some distinctions that I see as meaningful and relevant. These are largely in the form of contrasts:

- **Touch-tablets *vs* Touch screens:** In some ways these are two extremes of a continuum. If, for example, you have paper graphics on your tablet, is that a display (albeit more-or-less static) or not? What if the "display" on the touch tablet is a tactile display rather than visual? There are similarities, but there are real differences between touch-sensitive display surfaces, *vs* touch pads or tablets. It is a difference of *directness.* If you touch exactly where the thing you are interacting with is, let's call it a touch screen or touch display. If your hand is touching a surface that is not overlaid on the screen, let's call it a touch tablet or touch pad.

- **Discrete *vs* Continuous:** The nature, or "language" of touch input is highly shaped by the type of actions that are used in interacting with the touch technology. The same touch technology on the same device can assume a *very* different character, depending on whether the interface depends on discrete *vs* continuous actions. For example, the most common way of working with touch screens is with direct finger selection of items. For example, one might be asked to "push" the graphical OK button to conclude a transaction on an ATM, or "tap" on the keys of a graphical keyboard in order to enter text (in this latter case, multi-touch supports the ability to hold the SHIFT key down while simultaneously tapping one or more alphabetic keys, in order to get upper case).

  In contrast, one can also design the interaction such that control is asserted by means of *continuous* actions, or gestures, such as the lateral stroke gesture that is commonly used in photo-viewing applications to enable the user to go to the next, or previous, image in a sequence, depending on the direction of the to zoom in or out of an image or map, for example.

The discrete actions are typically accompanied by graphical cues, or feedback (feedforward, actually), that make them self-revealing. Some contiuous actions share this property, such as dragging the handle of a graphical linear potentiometer to change the speaker volume for a video, but many do not - such as the example of flicking through photos or pinching to zoom into a map. In these cases, the user needs to somehow know what can be done, how to do it, and when it can be done. The point is this: with the same multi-touch device on the same hardware, the nature of the experience can vary greatly depending on which kind of interaction is used, where, and how.

- **Location Specificity**: How accuratley the user has to position a touch at a particular location for a particular action has a significant effect on the nature of the interaction. Typing the "e" key on a graphical keyboard requires a rather high level of accuity, but less than selecting the gap between the second and third 'l" in "allleu" in order to correct the spelling to "alleleu". On the other hand, some actions, such as the lateral flick frequently used to go to the next or previous image in a photo viewer is far less demanding on where it occurs. With full-screen viewing, for example, it can be initiated pretty much anywhere on the screen. How demanding touch-screen interaction is in this regard has a significant impact on not only overall user experience, but also its suitability for certain applications. In general, the more precise one must be in terms of where the touch occurs, the more visually demanding the task is. And, the more the interaction demands visual attention, the less acceptable that interface is for cases where the eyes (not to mention the hands) should be deployed elsewhere. For example, the design of most touch screen controlled devices that I have seen should be illegal to use while driving an automobile, and certainly should not be integrated into the console. Because of its importance, let me dive into this a little bit deeper.

  As primarilly deployed today, touch-screens are relatively uniform flat surfaces. There is no tactile feedback like that provided by a piano keyboad, with its the cracks between the keys, or the different levels of the black and white keys, or the different shapes of the knobs of your old-school car radio, which "told" you - through touch - that you were touching the volume control, tuning knob, or preset button. With touch screens, yes, you may know approximately where the graphical QWERTY keyboard is positioned, for example, and you may know about where the comma (",") key is located, but unlike a traditional car radio, you cannot feel your way to its location. The absence of tactile feedback means you need to use your eyes. A key message that I want to convey is that this is true even with touch screens that provide so-called tactile feedback. The reason is that typically only provides feedback as to what action was just done, not what you are about to do. For our purposes, what I am describing as missing is not tactile feed*back*, but what might be better referred to as feed*forward*. (In reality, what I am calling "feedforward" is actually still "feedback", but it is feedback for the task of finding the appropriate control, not activating it. This just points out that we need to have a finer granularity in our task analysis, as well as the types of feedback supported). Finally, the significance and impact of all of this is amplified by the fact that traditional mechanical controls a persistent in their location, and therefore one can not only feel them, one can commit their approximate location to muscle memory, through practice. With touch screen interfaces, multiple controls typically appear at different times at the same location, thereby creating a "moded" situation which most likely reduces the potential for such motor learning in many, if not most, situations.

- **Degrees of Freedom:** The richness of interaction is highly related to the richness/numbers of degrees of freedom (DOF), and in particular, continuous degrees of freedom, supported by the technology. The conventional GUI is largely based on moving around a single 2D cursor, using a mouse, for example. This results in 2DOF. If I am sensing the location of two fingers, I have 4DOF, and so on. When used appropriately, these technologies offer the potential to begin to capture the type of richness of input that

- **Size matters:** Size largely determines what muscle groups are used, how many fingers/hands can be active on the surface, and what types of gestures are suited for the device.

- **Orientation Matters - Horizontal *vs* Vertical:** Large touch surfaces have traditionally had problems because they could only sense one point of contact. So, if you rest your hand on the surface, as well as the finger that you want to point with, you confuse the poor thing. This tends not to occur with vertically mounted surfaces. Hence large electronic whiteboards frequently use single touch sensing technologies without a problem.

- **There is more to touch-sensing than contact and position:** Historically, most touch sensitive devices only report that the surface has been touched, and where. This is true for both single and multi touch devices. However, there are other aspects of touch that have been exploited in some systems, and have the potential to enrich the user experience:

  1. **Degree of touch / pressure sensitivity:** A touch surfaces that that can independently and continuously sense the degree of contact for each toouch point has a far higher potential for rich interaction. Note that I use "degree of contact" rather than pressure since frequently/usually, what passes for pressure is actually a side effect – as you push harder, your finger tip spreads wider over the point of contact, and what is actually sensed is amount/area of contact, not pressure, *per se*. Either is richer than just binary touch/no touch, but there are even subtle differences in the affordances of pressure *vs* degree.

  2. **Angle of approach:** A few systems have demonstrated the ability to sense the angle that the finger relative to the screen surface. See, for example, McAvinney's *Sensor Frame*, below. In effect, this lgives the finger the capability to function more-or-less as a virtual joystick at the point of contact, for example. It also lets the finger specify a vector that can be projected into the virtual 3D space behind the screen from the point of contact - something that could be relevant in games or 3D applications.

  3. **Force vectors:** Unlike a mouse, once in contact with the screen, the user can exploit the friction between the finger and the screen in order to apply various force vectors. For example, without moving the finger, one can apply a force along any vector parallel to the screen surface, including a rotational one. These techniques were described as early as 1978, as shown below, by Herot, C. & Weinzapfel, G. (1978). Manipulating Simulated Objects with Real-World Gestures Using a Force and Position Sensitive Screen, *Computer Graphics*, 18(3), 195-203.].

  Such historical examples are important reminders that it is human capability, not technology, that should be front and centre in our considerations. While making such capabilities accessible at reasonable costs may be a challenge, it is worth remembering further that the same thing was also said about multi-touch. Furthermore, note that multi-touch dates from about the same time as these other touch innovations.

- **Size matters II:** The ability of to sense the size of the area being touched can be as important as the size of the touch surface. See the Synaptics example, below, where the device can sense the difference between the touch of a finger (small) *vs* that of the cheek (large area), so that, for example, you can answer the phone by holding it to the cheek.

- **Single-finger *vs* multi-finger:** Although multi-touch has been known since at least 1982, the vast majority of touch surfaces deployed are single touch. If you can only manipulate one point, regardless of with a mouse, touch screen, joystick, trackball, etc., you are restricted to the gestural vocabulary of a fruit fly. We were given multiple limbs for a reason. It is nice to be able to take advantage of them.

techniques used if it is peculiar to the technology or not. Many, if not most, of the so-called "multi-touch" techniques that I have seen, are actually "multi-point". Think of it this way: you don't think of yourself of using a different technique in operating your laptop just because you are using the track pad on your laptop (a single-touch device) instead of your mouse. Double clicking, dragging, or working pull-down menus, for example, are the same interaction technique, independent of whether a touch pad, trackball, mouse, joystick or touch screen are used.

- **Multi-hand *vs* multi-finger:** For much of this space, the control can not only come from different fingers or different devices, but different hands working on the same or different devices. A lot of this depends on the scale of the input device. Here is my analogy to explain this, again referring back to the traditional GUI. I can point at an icon with my mouse, click down, drag it, then release the button to drop it. Or, I can point with my mouse, and use a foot pedal to do the clicking. It is the same dragging technique, even though it is split over two limbs and two devices. So a lot of the history here comes from a tradition that goes far beyond just multi-touch.

- **Multi-person *vs* multi-touch:** If two points are being sensed, for example, it makes a huge difference if they are two fingers of the same hand from one user *vs* one finger from the right hand of each of two different users. With most multi-touch techniques, you do *not* want two cursors, for example (despite that being one of the first thing people seem to do). But with two people working on the same surface, this may be exactly what you *do* want. And, insofar as multi-touch technologies are concerned, it may be valuable to be able to sense which person that touch comes from, such as can be done by the *Diamond Touch* system from MERL (see below).

- **Points *vs* Gesture:** Much of the early relevant work, such as Krueger (see below) has to do with sensing the pose (and its dynamics) of the hand, for example, as well as position. That means it goes way beyond the task of sensing multiple points.

- **Stylus and/or finger:** Some people speak as if one must make a choice between stylus *vs* finger. It certainly is the case that many stylus systems will not work with a finger, but many touch sensors work with a stylus or finger. It need not be an either or question (although that might be the correct decision – it depends on the context and design). But any user of the Palm Pilot knows that there is the potential to use either. Each has its own strengths and weaknesses. Just keep this in mind: if the finger was the ultimate device, why didn't Picasso and Rembrandt restrict themselves to finger painting? On the other hand, if you want to sense the temperature of water, your finger is a better tool than your pencil.

- **Hands and fingers *vs* Objects:** The stylus is just one object that might be used in multi-point interaction. Some multi-point / multi-touch systems can not only sense various different objects on them, but what object it is, where it is, and what its orientation is. See Andy Wilson's work, below, for example. And, the objects, stylus or otherwise, may or may not be used in conjunction and simultaneously with fingers.

- **Different *vs* The Same:** When is something the same, different or obvious? In one way, the answer depends on if you are a user, programmer, scientist or lawyer. From the perspective of the user interface literature, I can make three points that would be known and assumed by anyone skilled in the art:

    1. *Device-Independent Graphics:* This states that the same technique implemented with an alternative input device is still the same technique. For example, you can work your GUI with a stylus, touch screen, mouse, joystick, touchpad, or trackball, and one would still consider techniques such as double-clicking, dragging, dialogue boxes as being "the same" technique;

    2. *The Interchange of devices is not neutral from the perspective of the user:* While the skill of using a

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.

WHAT WILL YOU BUILD? | sales@docketalarm.com | 1-866-77-FASTCASE

fastcase® Smarter legal research.