A THREE-STATE MODEL OF GRAPHICAL INPUT[*+1]

William A.S. BUXTON

Computer Systems Research Institute, University of Toronto, Toronto, Ontario, Canada M5S 1A4

A model to help characterize graphical input is presented. It is a refinement of a model first introduced by Buxton, Hill and Rowley (1985). The importance of the model is that it can characterize both many of the demands of interactive transactions, and many of the capabilities of input transducers. Hence, it provides a simple and usable means to aid finding a match between the two.

After an introduction, an overview of approaches to categorizing input is presented. The model is then described and discussed in terms of a number of different input technologies and techniques.

1. INTRODUCTION

All input devices are not created equal in their capabilities, nor input techniques (such as pointing, dragging, and rubber-banding) in their demands. The variation in each is both qualitative (types of signals) and quantitative (amount of signal, or bandwidth). Because of this variation, it is important that designers be able to answer questions like, "What particular demands does dragging make of a transducer, compared to selection?" or "How well are these task requirements met by a touch screen or a trackball?" Unfortunately, there is little available in the way of help from either theory or guidelines.

If language is a tool of thought, then a start to addressing this shortcoming is to develop a vocabulary common to both devices and techniques, and which augments our ability to recognize and explore relationships between the two.

In the remainder of this paper, a model which contributes to such a vocabulary is presented. It is a simple state-transition model which elucidates a number of properties of both devices and techniques.

2. TAXONOMIES OF INPUT

Traditionally, input devices have been discussed in terms of their mechanical and electrical properties (Foley & Van Dam, 1982; Sherr, 1988). Discussions centre around "joysticks," "trackballs," and "mice," for example.

---

* The research reported has been sponsored by the Natural Sciences and Engineering Research Council of Canada and Xerox PARC.

+ Citation: Buxton, W. (1990). A three state model of graphical input. In D. Diaper et al. (Eds), *Human-Computer Interaction - INTERACT '90.* Amsterdam: Elsevier Science Publishers B.V. (North-Holland), 449-456.

1 In the published version of this paper my scholarship was lacking in that I missed a very important reference, that of Newman (1968). I have added the reference for now, and will integrate a discussion of that classic paper to my text in order to place the two in proper context.

Several studies have attempted to evaluate the technologies from the perspective of human performance. Many of these are summarized in Greenstein and Arnaut (1988) and Milner (1988). A common problem with such studies, however, is that they are often overly device-specific. While they may say something about a particular device in a particular task, many do not contribute significantly to the development of a general model of human performance. (There are exceptions, of course, such as Card, English and Burr, 1978.)

With the objective of isolating more fundamental issues, some researchers have attempted to categorize input technologies and/or techniques along dimensions more meaningful than simply "joystick" or "trackball." The underlying assumption in such efforts is that better abstractions can lead us from phenomenological descriptions to more general models, and hence better analogies.

An early development in this regard was the concept of *logical devices* (GSPC, 1977, 1979). This occurred in the effort to evolve device-independent graphics. The object was to provide standardized subroutines that sat between physical input devices and applications. By imposing this intermediate layer, applications could be written independent of the idiosyncrasies of specific devices. The result was more standardized, general, maintainable, and portable code.

As seen by the application, logical devices were defined in terms of the type of data that they provided. There was one logical device for what the designers felt was each generic class of input. Consequently, we had the beginnings of a taxonomy based on use. The devices included a *pick* (for selecting objects), a *locator,* and *text.*

Foley, Wallace, and Chan (1984) took the notion of logical devices, and cast them more in the human perspective than that of the application software. They identified six generic transactions (which were more-or-less the counterparts of the GSPC logical devices) that reflected the user's intentions:

- *select* an object
- *position* an object in 1, 2, 3 or more dimensions;
- *orient* an object in 1, 2, 3 or more dimensions;
- *ink,* i.e., draw a line;
- *text,* i.e., enter text;
- *value*, i.e., specify a scalar value.

They then proceeded to enumerate a relatively comprehensive set of techniques and technologies capable of articulating each of these basic primitives.

Buxton (1983) introduced a taxonomy of input devices that was more rooted in the human motor/sensory system. The concern in this case was the ability of various transducers to capture the human gesture appropriate for articulating particular intentions. Consequently, input devices were categorized by things such as the property sensed (position, motion, or pressure), the number of dimensions sensed, and the muscle groups required to use them.

Recent research at Xerox PARC has built on this work (Card, Mackinlay and Robertson,1990; Mackinlay, Card and Robertson, 1990). Their taxonomy captures a broad part of the design space of input devices. The model captures both the discrete and continuous properties of devices, (unlike that of Buxton, which could only deal with the latter).

Together, this collected research begins to lay a foundation to support design. However, none of the models are complete in themselves, and there are still significant gaps. One is the lack of a vocabulary that is capable of capturing salient features of interactive techniques and technologies in such a way as to afford finding better matches between the two.

In what follows, a model (first suggested in Buxton, Hill and Rowley, 1985) is developed which provides the start to such a vocabulary. It takes the form of a simple state-transition model and builds on the work mentioned above. In particular, it refines the notion of device state introduced in the PARC model of Card, Mackinlay and Robertson (1990) and Mackinlay, Card and Robertson (1990).

3. A THREE-STATE MODEL

The model can be introduced within the context of a direct manipulation interface, such as that of the Apple Macintosh. Consider moving the mouse without the button pushed. One way to characterize the state of the system at this point is as *tracking.* If, however, we point at an icon, depress the mouse button and move the mouse while holding the button down, we have entered a new state, *dragging.* These simple states are represented in Fig. 1.

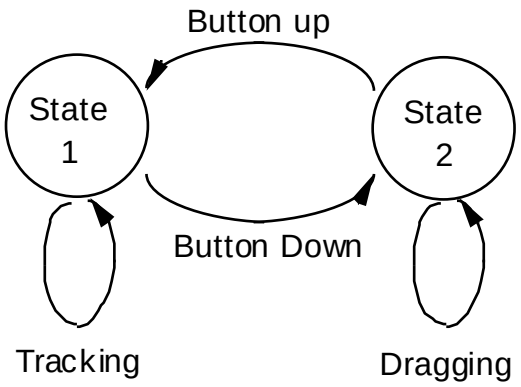

Figure 1. Simple 2-State Transaction
In State 1, moving the mouse causes the tracking symbol to move. Depressing the mouse button over an icon permits it to be dragged when the mouse is moved. This is State 2. Releasing the mouse button returns to the tracking state, State 1.

Consider now the situation if a touch tablet rather than a mouse was connected to the system. For the purpose of the example, let us assume that the touch tablet is capable of sensing only one bit of pressure, namely touch or no-touch.

In this case we also get two states, but only one is common to the previous example. This is illustrated in Fig. 2. The first state, (State 0), is what we will call *out of range*, (OOR). In this state, any movement of the finger has no effect on the system. It is only when the finger comes in contact with the tablet that we enter the State 1, the tracking state seen in the previous example.

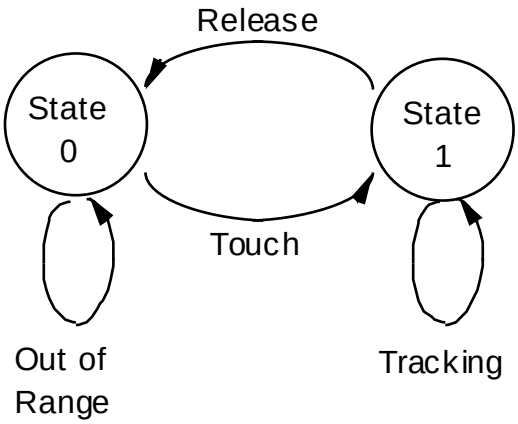

Figure 2. State 0-1 Transaction

Assume a touch tablet. In State 0, moving the finger is out of range (OOR), so has no effect. When the finger is in contact with the tablet, the tracking symbol follows the finger's motion (State 1: tracking). The system returns to State 0 when the finger releases contact from the tablet surface.

Each example has one state that the other cannot reach. Lifting a mouse off of one's desk is not sensed, and has no effect. No interactive technique can be built that depends on this action. Consequently, State 0 (the OOR condition) is undefined. Conversely, without some additional signal, the touch tablet is incapable of moving into the dragging state (State 2). To do so would require a signal from a supplementary key press or from a threshold crossing on a pressure-sensitive tablet (Buxton, Hill and Rowley, 1985).

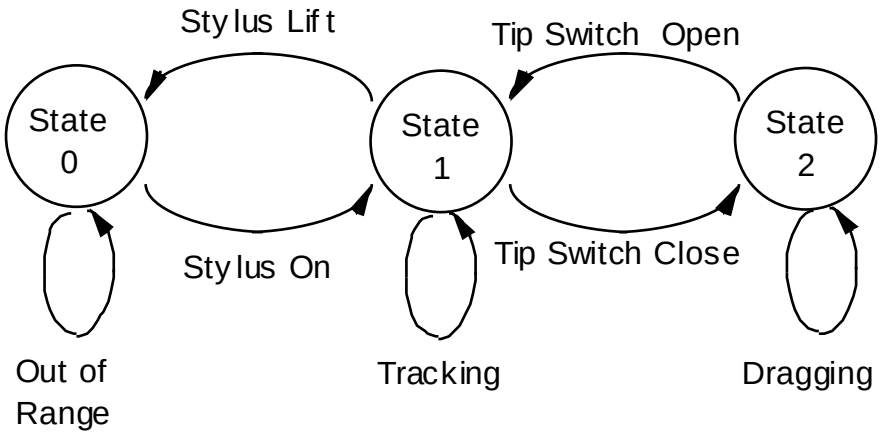

Figure 3. State 0-1-2 Transaction

Assume a graphics tablet with stylus. In State 0, the stylus is off of the tablet and the tip switch in its open state. Moving the stylus has no effect since it is out of range (OOR). When the stylus is in range, the tracking symbol follows the stylus' motion (State 1: tracking). Extra pressure on the stylus closes the tip switch, thereby moving the system into State 2.

There are, however, transducers that are capable of sensing all three states. A graphics tablet with a stylus would be one example.[1] This is illustrated in Fig. 3.

The three states introduced in the above examples are the basic elements of the model. There can be some variations. For example, with a multi-button mouse (or the use of multiple clicks), State 2 becomes a set of states, indexed by button number, as illustrated in Fig. 4.

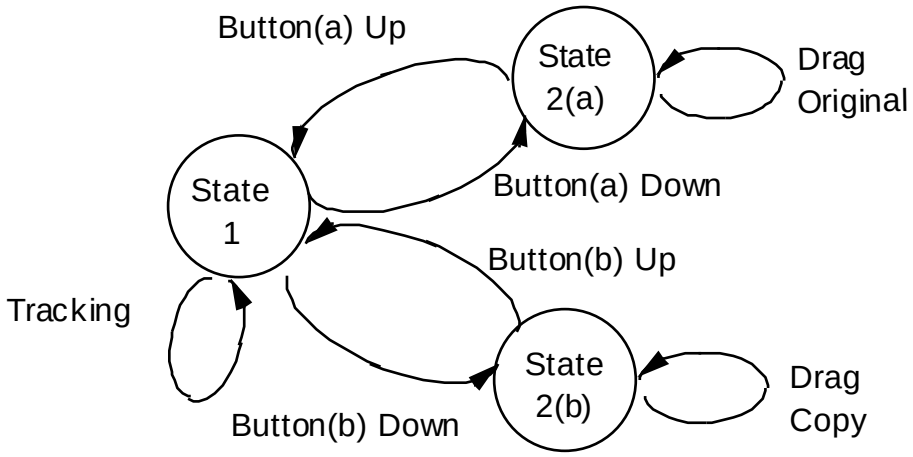

Figure 4. State 2 Set

With a multi-button mouse, for example, multiple State 2's are available. For example, selecting an object with button$_a$ may cause the selected object to be dragged, whereas selecting the object with button$_b$ may mean that a copy is dragged. Multiple State 2s can also be realized by multiple clicks on the mouse, as with the Apple Macintosh, where single clicks are used to select and double clicks to "open."

While the model is simple, it will be shown how important properties of devices and interactive techniques can be characterized in terms of these three states, and how this representation can be used in design. Weaknesses of the the model and areas for future research will also be discussed.

Note that in what follows, states will be referred to by number (State 1, for example) rather than by description. The consequences of the action performed in a particular state can vary. For example, State 2 could just as easily been "inking" or "rubber banding" as "dragging." The ordinal nomenclature is more neutral, and will be used.

4. DEVICES AND TRANSACTIONS: TABULATING ATTRIBUTES

Two tables are presented. The first summarizes the demands of a number of transaction types expressed in terms of the states and state transitions that they require from a supporting transducer. The second summarizes the capabilities of a number of input devices, expressed in terms of this same type of state information. By comparing the two tables, a simple means is provided to evaluate the match between transducers and transactions.

DOCKET ALARM

Explore Litigation Insights

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

Real-Time Litigation Alerts

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

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

Advanced Docket Research

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

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

Analytics At Your Fingertips

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

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

API

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

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

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

FINANCIAL INSTITUTIONS
Litigation and bankruptcy checks for companies and debtors.

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