SurfaceMouse: Supplementing Multi-Touch Interaction with a Virtual Mouse

[1]Tom Bartindale [2]Chris Harrison [1]Patrick Olivier [2]Scott E. Hudson

[1]Newcastle University
CultureLab, Kings Walk
Newcastle Upon Tyne, NE1 7RU, UK
{t.l.bartindale,p.l.olivier}@ncl.ac.uk

[2]Human-Computer Interaction Institute
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
{chris.harrison, scott.hudson}@cs.cmu.edu

ABSTRACT

We present SurfaceMouse, a virtual mouse for multi-touch surface computing. Although moving away from the direct touch manipulation paradigm, our system brings many significant benefits seen in absolute clutched devices to surface computing. Features include high and variable control device gains, several degrees of freedom in a single hand gesture, ability to target small GUI items, and a familiar method for reaching far areas of large displays. Importantly, this benefit is realized by leveraging what users already know and have tremendous experience with - physical mice. Results from our proof-of-concept evaluation reflect this; users were able to use and recognize our system without training or prompts. Being entirely virtual, SurfaceMouse can be implemented in existing systems with little more than a software update.

ACM Classification: H.5.2 [User Interfaces]: Input devices and strategies; Graphical user interfaces.

General terms: Human Factors

Keywords: Table, surface, multi-user, scrolling, cursor.

INTRODUCTION

Recent developments in touch surface technologies have seen a proliferation of proposals for new interaction techniques. These include multi-touch actions for the direct manipulation of interface elements, as well as gestures for command invocation and multi-element selection. Indeed, multi-touch interaction affords many new and interesting avenues for expressive input and collaborative interaction. However, little attention has been paid to how mice could operate and supplement interaction in touch-driven systems. This is unfortunate given the well-established benefits of the mouse and its familiarity among users, with potentially thousands of hours of training.

In this paper, we describe SurfaceMouse, a virtual mouse implementation for multi-touch surfaces. A key design objective was to leverage as much pre-existing knowledge (and potentially muscle memory) regarding mice as possible, making interactions immediately familiar. To invoke SurfaceMouse, a user simply places their hand on an interactive surface as if there was a mouse present (Figure 1). The system recognizes this characteristic gesture and renders a virtual mouse under the hand, which can be used like a real mouse. In addition to two-dimensional movement (X and Y axes), our proof-of-concept implementation supports left and right clicking, as well as up/down scrolling. Importantly, SurfaceMouse does not preempt any existing touch interaction or multi-touch gestures. It simply acts as an additional gesture.

MOTIVATION & BENEFITS

Although appearing at first glance to be a curious step back in surface computing, SurfaceMouse brings several significant benefits. Foremost, our approach provides an immediate, effective and intuitive means to provide high control-device (CD) gain on touch surfaces (vs. 1:1 of conventional direct finger input). The CD gain is easily controlled in software, and could be dynamically modified to aid different tasks or tailored to a user's motor ability. Additionally, because the cursor is decoupled from the finger/hand location, target occlusion can be dramatically reduced. It is well established that both of these attributes can improve pointing ease and accuracy.

Additionally, high targeting accuracy (plus left and right mouse buttons) could enable users to run existing desktop-class GUI applications. These tend to be unwieldy on touch surface due to their small interface elements (i.e., too small or dense for "big" fingers). Although not ideal, backwards

Figure 1. SurfaceMouse (left), Physical Mouse (right).

TEI'11, January 22–26, 2011, Funchal, Portugal.

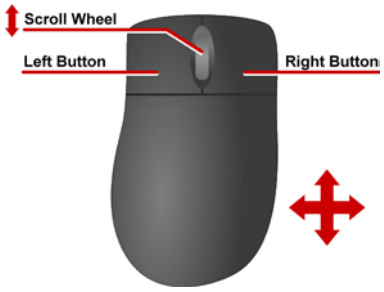

Figure 2. Actions available in our prototype implementation.

compatibility of software is a significant real-world issue given the huge volume of existing software. Moreover, an increasing number of computer systems are hybrid – running a conventional desktop operating system, but containing screens capable of multi-touch input (e.g., Windows 7 tablet computers). SurfaceMouse could help bring these two interface paradigms closer together and yield a more seamless user experience.

This ability to interact with smaller targets also means applications can make use of smaller GUI elements, allowing for more productive use of this screen space. This is especially important as screen real estate on many multi-touch platforms is at a premium. This issue is further compounded in multi-user environments. Conversely, large screen interaction can be troublesome due to the fact that users may not be able to reach all areas of the screen [1,21]. SurfaceMouse can be used within this scenario to enable in-direct interaction over larger physical distances.

As noted previously, SurfaceMouse immediately enables input with several degrees of freedom: continuous motion in two dimensions, two mouse buttons, and up/down scrolling (Figure 2). This level of expressivity is rare for a single-handed gesture. This success relies heavily on the years of training an average user has with physical mice. SurfaceMouse simply co-opts this knowledge and applies it to a new (albeit virtual) form. We suspect that if different movement/buttons binding were employed, interaction would prove less intuitive to users.

Finally, the lack of a physical device has several smaller benefits. First, SurfaceMouse can be deployed and updated entirely in software. Second, multiple users can invoke SurfaceMice for collaborative purposes, a central tenet of surface computing. When interaction is complete, SurfaceMice disappear, an important aspect when designing public displays, where physical objects can be removed or damaged easily. Third, the lack of physicality also allows SurfaceMouse to provide special button mappings to left- and right-handed users (handedness is automatically detected). Tabletop systems with physical mice are typically shaped for the right hand, which can be uncomfortable for left-handed users to manipulate.

RELATED WORK

As touch screens entered mainstream, pointing performance relative to conventional interaction became a popular research subject. The seminal paper by Sears and Shneiderman [20] showed correctly designed high precision capacitive touch screens could yield touch selection precision comparable to that of a mouse, for targets as small as 4 pixels. However, below four pixels, mice significantly outperformed touch selection. Vision-based multi-touch technologies have traditionally suffered from poorer accuracy due to factors such as limited camera resolution, illumination artifacts, and shadows.

For example, Forlines et al. [6] showed error rates for diffuse illumination-based [14] touch interaction were nearly twice that of mice, even when using relatively large targets (e.g., dimensions of 16 and 32 pixels). Wang and Ren [24] conducted a thorough exploration of contact point properties for FTIR surfaces [7]. This work found that different contact profiles occur as a consequence of varying degrees of arm extension. Similarly, Holz and Baudisch [9] found finger angle led to targeting variances on capacitive touch screens.

No matter what the sensing technique, the fact is that our fingers are simply larger than a digital cursor. This inherently reduces accuracy, both by having more contact area and often by introducing occlusion at the point of contact (see e.g., [4,9,23] for more discussion on the "fat finger problem"). Consequently, many techniques have been proposed for higher precision selection. These include visual enhancements [1,2,4,17,23], gesturing schemes [2,3] and bimanual manipulation [12,15]. There has also been work addressing issues regarding reach on large interactive surfaces [1,5,16].

The fluidity and precision of mice has been of interest to tabletop researchers previously. Hartmann et al. [8] experimented with placing physical mice and keyboards on tabletops to supplement interaction. Many compelling interactions were introduced, most of which are immediately applicable to this work. There has even been new work in extending the touch functionality of physical mouse [22].

Finally, closest to this work is SDMouse [13], which provides a partial, but powerful mouse implementation using cording gestures. For example, the index finger can be tracked on screen (the cursor rendered beneath it). To left click, the user presses their index and middle finger against the surface; to right click, the index and ring finger could be used. SurfaceMouse, although similar in appearance, has a significantly differently character of operation. In particular, we decouple cursor location from hand position in order to reduce occlusion and obtain a CD gain greater than 1:1, offering improved pointing accuracy. Secondly, rather than using cording gestures, we directly emulate conventional mouse controls (e.g., left click with a button). The graphic of a mouse provides a useful and omnipresent visual affordance. Moreover, users can use any fingers they like to operate these functions as per their preference. This further aided our objective of not requiring the user to learn anything new.

IMPLEMENTATION

Our initial prototype was written in C# (WPF) for use on a Microsoft Surface, which uses diffuse illumination tracking [14] (Figure 3). However, many other touch technologies are applicable, including FTIR [7], capacitive [18], IFSR [19], and optical fiber [10] technologies. In general, any interactive surface that allows for multiple points of interaction and a measurement of the area of contact should be sufficient.

When a new contact is detected, the system scans over all present blobs, finding the largest and makes the assumption it is a wrist contact. The system then checks that exactly five finger-sized blobs are present in a suitable configuration near the presumed wrist blob. If all criteria are met, SurfaceMouse is invoked. Figure 3 illustrates the high level process. To provide visual confirmation and ongoing feedback, a mouse image is rendered below the hand (appropriate for the user's handedness). This virtual mouse tracks with the hand. Simultaneously, mouse movement events are created using the relative motion of the wrist.

In addition to relative movement, SurfaceMouse also features two buttons and a scroll wheel, all of which function like their physical counterparts and track appropriately with the hand. We chose not to map button clicks to particular fingers (e.g., middle finger controls the scroll wheel). Anecdotally, there appear to be a wide variety of finger-to-function mappings. For example, some people use their index finger for left click and scrolling, and their middle finger for right click). To account for this, our system triggers a function when a blob appears in the active region of a particular function (e.g., the left button area), allowing for any digit to be used.

SurfaceMouse becomes inactive only after the wrist is lifted from the interactive surface. This allows fingers to be lifted for actions like clicking. When the user removes all contacts from the surface of the display, SurfaceMouse does not immediately disappear. Instead, a timeout is used to increase robustness when tracking is accidentally lost and also to allow for clutching, just like a real mouse.

FUNCTIONALITY

Hand Size

Our recognition algorithm requires discrimination between wrist, finger and non-SurfaceMouse contacts (e.g., tangibles, arms). To inform our size and shape constraints, we measured a variety of hands. As a result, our system is robust in its wrist and finger discrimination, and we can use the inferred hand size to create a virtual mouse appropriately sized to the user (e.g., for children).

Orientation

SurfaceMouse calculates orientation of the hand using the ovoid wrist blob. An orientation adjustment can be made on a personal basis to provide custom alignment with the fingers. This feature allows users to walk up to the surface and begin interaction with the interface regardless of their orientation relative to the surface. This is important in collaborative settings on horizontal surfaces, where people traditionally "sit around the table." Note that only the wrist blob is used for translation and rotation tracking. This allows for accurate use without fingers needing to be on the surface.

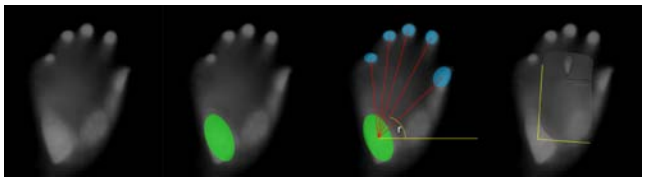

Figure 3. Left to right: 1) Infrared camera image of SurfaceMouse gesture; 2) Palm detection; 3) Finger tip detection; 4) Size, orientation, and handedness calculation.

Handedness

The system changes the horizontal orientation of the virtual mouse depending on whether a left or right hand is used. This hand feature is achieved by using the maximum and minimum angle of all finger blobs within the boundary area, compared with the center of the wrist. If this angle is above or below a certain value, handedness can be inferred. If the angle is ambiguous, our system defaults to right-handed mode (~80% of the population). Finally, handedness adaptation allows a single user to wield two mice, which is valuable in certain use contexts [11].

Scroll Wheel

In addition to virtual mouse buttons, we also include a virtual scroll wheel. This feature is accomplished by simply taking the relative movement of a finger blob when it appears in the scroll wheel area.

Microsoft Windows Driver

As a true test of our prototype's ability to perform as a mouse, we created a Microsoft Windows driver for our system. Since SurfaceMouse implements a standard array of mouse features, we simply send events through the standard mouse API to the OS. We chose to make a Windows driver, as our Microsoft Surface system is also able to run a conventional desktop version of Windows. In our proof-of-concept evaluation, described subsequently, this allowed users to interact with a standard desktop web browser using SurfaceMouse.

Multi-User

An additional advantage of using multi-touch technology is its innate ability to support multiple users. We capitalize on this ability, and allow each user to wield a SurfaceMouse. Although this feature was implemented, it was not possible to enable it outside of our simple sandbox as most operating systems only allow one mouse pointer.

EVALUATION

We evaluated our proof-of-concept SurfaceMouse for general usability compared to conventional touch selection on a Microsoft Surface. In order to get a preliminary understanding of the sensitivity and usability of our approach, we presented our prototype to nine beta testers (three female, average age 28) who had not seen or used the system, but were familiar with using a physical mouse.

First, they were presented with a standard desktop web browser, and asked to navigate through a variety of content using direct finger touches as the only interaction. Next, a short introduction was given outlining the gesture required to produce SurfaceMouse. Users then were presented with a simple sandbox environment in order to become familiar with the technique.

Then, using our SurfaceMouse mouse driver for Windows, testers were then asked to repeat the browsing task and freely navigate the web. The mouse driver displayed an image of SurfaceMouse on top of all windows (see Video Figure). The movement and acceleration profiles were set to the standard Windows values. During the session, users were encouraged to provide commentary on their impressions. Questions were also asked to elicit feedback, especially about how it compared to the earlier touch-driven browsing task.

Reactions were positive. People consistently used words like "natural" and "intuitive" to describe the experience. Participants immediately recognized what it was and how to use it without prompts. In regards to how it compared to touch-based interaction, at least half noted that they felt less frustrated when clicking on hyperlinks. This was because of the visual feedback afforded by the mouse pointer and the accuracy gained by indirectly controlling it. Scrolling without having to move the hand (i.e., retarget) was also seen as beneficial. Finally, the fact that SurfaceMouse disappeared when not needed was also liked. However, as expected, users missed the physicality afforded by a real mouse, and consequently, felt less in control of the pointer.

CONCLUSION

We presented SurfaceMouse, a virtual mouse implementation for multi-touch surface computing. It may seem odd to introduce a pointing device made for "desktop" computing into the new and exciting space of finger and touch manipulation. However, mice have well-established benefits that could be used to great effect in touch-driven environments and potentially solve several outstanding interaction problems, including reach and precision targeting. We discussed these and other benefits at length and also described unique interactions our approach enables. We concluded with a brief overview of qualitative feedback from our pilot testing.

REFERENCES

1. Abednego, M., Lee, J., Moon, W., and Park, J. I-Grabber: expanding physical reach a large-display tabletop environment through the use of a virtual grabber. *Proc. ITS '09*. 61-64.
2. Albinsson, P. and Zhai, S. High precision touch screen interaction. *Proc. CHI '03*. 105-112.
3. Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tadler, P. Bederson, B., and Zierlinger. A. Drag-and- Pop and Drag-and-Pick: Techniques for Accessing Remote Screen Content on Touch and Pen-operated Systems. *Proc. Interact '03*. 57–64.
4. Benko, H., Wilson, A. D., and Baudisch, P. Precise Selection Techniques for Multi-Touch Screens. *Proc. CHI '06*. 1263–1272.
5. Echtler, F., Huber, M., and Klinker, G. Shadow tracking on multi-touch tables. *Proc. AVI '08*. 388-391.
6. Forlines, C., Wigdor, D., Shen, C., and Balakrishnan, R. Direct-touch vs. mouse input for tabletop displays. *Proc. CHI '07*. 647-656.
7. Han, J. Y. Low-cost multi-touch sensing through frustrated total internal reflection. *Proc. UIST '05*. 115-118.
8. Hartmann, B., Morris, M. R., Benko, H., and Wilson, A. D. Augmenting interactive tables with mice & keyboards. *Proc. UIST '09*. 149-152.
9. Holz, C. and Baudisch, P. The generalized perceived input point model and how to double touch accuracy by extracting finger-prints. *Proc. CHI '10*. 581-590.
10. Jackson, D., Bartindale, T., and Olivier, P. FiberBoard: compact multi-touch display using channeled light. *Proc. ITS '09*. 25-28.
11. Latulipe, C., Mann, S., Kaplan, C.S., & Clarke, C.L. symSpline: symmetric two-handed spline manipulation. *Proc. CHI '06*. 349-358.
12. Lepinski, J., Grossman, T., Fitzmaurice, G. The design and evaluation of multitouch marking menus. *Proc. CHI '10*. 2233-2242.
13. Matejka, J., Grossman, T., Lo, J., and Fitzmaurice, G. The design and evaluation of multi-finger mouse emulation techniques. *Proc. CHI '09*. 1073-1082.
14. Matsushita, N. and Rekimoto, J. HoloWall: designing a finger, hand, body, and object sensitive wall. *Proc. UIST '97*. 209-210.
15. Moscovich, T. and Hughes, J.F. Indirect mappings of multi-touch input using one and two hands. *Proc. CHI '08*. 1275-1284.
16. Nacenta, M. A., Sallam, S., Champoux, B., Subramanian, S., and Gutwin, C. Perspective cursor: perspective-based interaction for multi-display environments. *Proc. CHI '06*. 289-298.
17. Olwal, A., Feiner, S., and Heyman, S. Rubbing and tapping for precise and rapid selection on touch-screen displays. *Proc. CHI '08*. 295-304.
18. Rekimoto, J. SmartSkin: an infrastructure for freehand manipulation on interactive surfaces. *Proc. CHI '02*. 113-120.
19. Rosenberg, I. and Perlin, K. The UnMousePad: an interpolating multi-touch force-sensing input pad. *Proc. SIGGRAPH '09*. 1-9.
20. Sears, A., Shneiderman, B. High precision touchscreens: design strategies and comparisons with a mouse. *International Journal of Man-Machine Studies*, 34(4), April 1991, 593-613.
21. Shoemaker, G., Tang, A., and Booth, K. S. Shadow reaching: a new perspective on interaction for large displays. *Proc. UIST '07*. 53-56.
22. Villar, N., Izadi, S., Rosenfeld, D., Benko, H., Helmes, J., Wes-thues, J., Hodges, S., Ofek, E., Butler, A., Cao, X., and Chen, B. Mouse 2.0: multi-touch meets the mouse. *Proc. UIST '09*. 33-42.
23. Vogel, D. and Baudisch, P. Shift: a technique for operating pen-based interfaces using touch. *Proc. CHI '07*. 657-666.
24. Wang, F. and Ren, X. Empirical evaluation for finger input properties multi-touch interaction. *Proc. CHI '09*. 1063-1072.
25. Wu, M. and Balakrishnan, R. Multi-Finger and Whole Hand Gestural Interaction Techniques for Multi-User Tabletop Displays. *Proc. UIST '03*. 193-202.