Keyboard-Surface Interaction: Using the keyboard's surface as a pointing device

1st Author Name
Affiliation
City, Country
e-mail address

2nd Author Name
Affiliation
City, Country
e-mail address

3rd Author Name
Affiliation
City, Country
e-mail address

ABSTRACT

Pointing devices residing on the keyboard could reduce the time needed to perform mixed pointing and typing tasks, since the user's hand never leaves the keyboard area. While such devices are described in the literature, their implementations have had lower performance (higher movement and homing time) when compared to the mouse and trackpad. We introduce Keyboard-Surface Interaction (KSI), a technique that integrates a pointing device on the keyboard's surface. We first built a prototype using infrared markers to explore the interaction design space. We then evaluated this prototype in a pointing and typing test with 25 subjects whose performance and reported discomfort was better with our prototype than with a trackpad. Finally, we built a marker-less KSI-device and had 15 people evaluate it in a mixed pointing and typing task. This marker-less device achieves similar performance to a mouse and demonstrates the feasibility of a KSI-device under real world conditions.

Author Keywords

Mouse; trackpad; performance; discomfort; input device.

ACM Classification Keywords

H.5.2. Information interfaces and presentation: User Interfaces: Input devices and strategies.

INTRODUCTION

The computer mouse, introduced in 1964 by Douglas Engelbart, is the preferred pointing device for computer users. However, it is by no means perfect. In a study with 25 participants, we show that the main complaint from using a mouse is discomfort and fatigue in different parts of the arm, caused by having to move the hand between the mouse and the keyboard while performing mixed pointing and typing tasks. Also, manipulation of the mouse has been shown to create additional discomfort in the hand due to wrist extension. Frequent mouse usage is a risk factor for Carpal Tunnel Syndrome (CTS) and repetitive hand strain injuries [2,10]. In contrast, people who spend more time using the keyboard than the mouse at work, have significantly lower risk for CTS [2]. Several technologies have been developed that could potentially reduce this discomfort by eliminating the use of an interaction device like the mouse or trackball. Examples of such technologies include touch sensing technologies like the trackpad on laptops, touchscreens on smartphones and tablets, and mid-air interaction (*e.g.*, Leap Motion, Kinect, Senz3D).

The main barrier of these alternative technologies to becoming everyday pointing devices for traditional computing environments (*i.e.*, desktops and laptops) is poor performance. In this paper, we will refer to performance as the combination of movement time, homing time, and error rate. *Movement time* [30] refers to the time elapsed from the moment the user starts moving the pointer on the screen to when she clicks on a target. High movement time is undesirable because it implies that more time is required to click on a target making the whole action more inefficient. *Homing time* [5] is defined as the time elapsed between when a pointing target is shown on a screen and when movement of the pointer is detected. Homing time is composed of the reaction time of the user and the time it takes the hand to physically travel to reach the pointing device. Thus, a high homing time implies that more time is needed to reach the pointing device and results in more time being required to complete a pointing task making the whole task less efficient. *Error rate* refers to the quantity of errors made by the user, where an error is defined as clicking outside (missing) the target at least once. In addition to the performance metrics, we also consider *discomfort* as a measure of the physical effort and strain caused on different parts of the arm (muscles) from using a pointing device.

For mid-air interaction devices, low performance is explained by low resolution or low sampling rates. For example, the Kinect has a sampling rate of 30 samples per second, leading to a perceivable lag in responsiveness, and higher movement time. In contrast, modern mice and trackpads have sampling rates from 100 to 1000 samples *per* second, with no perceivable lag. Additionally, mid-air (and touchscreen) devices cause a feeling of heaviness in the arm [15] even for interactions with durations shorter than 5 minutes [4]. Touchscreen devices, and some mid-air devices, suffer from occlusion (users cannot see what they are pointing at) and the fat finger problem (when targets are smaller than the fingers, resulting in a higher error rate). Mid-air interaction and touchscreens have failed to become mainstream pointing devices for laptop/desktop computers due to poor performance and increased discomfort.

While mainstream pointing devices (trackpad and mouse) tend to have better performance, they also suffer from increased discomfort. This is partly due to wrist extension issues (bad for both devices but significantly worse for the trackpad [29]), and partly due to having to move one's hand between the keyboard and the trackpad or mouse during mixed typing/pointing interactions (increased homing time).

To address the performance and discomfort issues of these existing pointing technologies, we introduce keyboard-surface interaction (KSI). KSI allows the user to perform pointing tasks on the surface of the keyboard itself, using the user's hand as the pointing device. KSI allows the user to rest her fingers on the keys without pressing them. This allows users to switch between pointing/typing tasks more quickly and seamlessly, reducing homing time and decreasing perceived discomfort. We demonstrate that a KSI device can have better performance than a trackpad and be as fast the mouse, unlike previous similar approaches (e.g., [11]).

Although the idea of KSI-like devices has been around the HCI community for some time, previous instantiations have suffered from two main short-comings that we try to solve with our work. The first is lack of differentiation: we do not simply want to *merge* pointing and typing capabilities, we want to take advantage of the keyboard and its relatively smooth surface to *transform* it into an input surface for pointing. Further, we deliberatively want to take advantage of the keyboard due to its improved health outcomes for Carpal tunnel and strained hand disorders over devices like the mouse. The second problem has been evaluation: Most papers suggesting KSI-like devices have done so by hypothesizing its performance advantage without conducting any user studies leaving the community at large to wonder about the actual performance of a KSI device in the real world. This is somewhat understandable given the amount of effort and time it takes to build and test this kind of device.

We began our work to address these problems by first exploring the design space of KSI through our own prototype named Fingers, a glove with infrared markers that supports pointing and clicking on a keyboard surface. We optimized Fingers using a series of target acquisition tests (often referred to as Fitts' law-style studies), including type of movement (absolute tracking *vs*. relative motion) and invocation method (alt *vs*. tab key to toggle between typing and pointing). We evaluated Fingers performance and discomfort against mainstream pointing devices (trackpad and mouse) on a mixed typing and pointing task using a target acquisition test. We hypothesize that KSI will be more comfortable because it allows the hand to rest in a natural position on the keyboard and minimizes the effort to switch between pointing and typing tasks. Our study results provide evidence for this hypothesis: 1) for more experienced Fingers users, we found that Fingers decreases discomfort in comparison to both the trackpad and mouse. In addition, 2) Fingers has a comparable error rate and shorter homing time than the mouse and the trackpad. The results of our studies serve as a specification for future KSI devices, (independent of sensing technology) with respect to error rate, speed, and spatial resolution.

To further demonstrate the value of KSI we built a second KSI device that does not require the user to wear a glove, called IndexSense: a marker-less KSI device based on our KSI specification and the lessons learned from our initial system. IndexSense uses computer vision methods to make the user's hand a pointing device. A performance evaluation of IndexSense, showed that based on the total time required for pointing (including time to and from the pointing device), *IndexSense is as fast as the mouse*.

RELATED WORK

Since the introduction of the mouse, most of its variations have retained the same idea: the user's hand manipulates an object, and this object's movement is measured and used to control the position of the pointer on a computer's screen. The trackpad is the most common mouse alternative for laptop computers, and touchscreens have become the norm for smart phones and tablet devices. In this section, we examine other options for controlling a pointer on a screen.

Different parts of the body such as the head [27,35], eyes [35], mouth [35] and feet [35] have been used as pointing devices. Typically, they have been popular only among persons with disabilities or injuries, such as carpal tunnel syndrome. For typical users, there is no reason to use any of these devices, as they result in decreased user performance. This is due to the increased time and effort required to complete pointing and typing tasks, which leads to a less satisfactory user experience.

SwiftPoint [1] is a commercially available miniature mouse that can be used on the surface of a keyboard. An evaluation of SwiftPoint in a target acquisition test revealed that while it had good performance, it was not as good as the trackpad or mouse. Another approach for replacing the mouse and trackpad is to make the user's hand the pointing device. An example of this is Touch&Type [11], a touch sensitive keyboard. Here, the keys themselves are touch sensitive and a single finger or the entire hand can be used to steer the cursor. The device is close in performance to the trackpad, a promising result. However, the error rate is not reported, and the evaluation does not include a target acquisition test, making comparison to other devices difficult. A similar device is presented in [24], where users move their hand adjacent to the keyboard to imitate the operation of a regular mouse. A user "clicks" by moving his index finger up and down, despite the absence of any mouse. Results for the error rate, discomfort and acceptance were not reported in [24].

Another similar device is FlowMouse [34]. FlowMouse uses a 30Hz gray scale camera with a resolution of 640x480 to detect motion flow (speed and direction) and determine how to move the pointer. FlowMouse uses a button on top of the left trackpad button for clutching (turning on and off the device). FlowMouse does not rely on detecting the hand, and instead moves the pointer based on general movement in the camera's field of view; however, an evaluation shows that FlowMouse was significantly slower than the trackpad. Another significant disadvantage of this technique is that it will be sensitive to movement of the non-pointing hand or arm, as both introduce noise into the system.

Mid-air and touch screen devices also can leverage the user's hand as a pointing device. Mid-air interaction, which occurs above the keyboard, leads to a feeling of heaviness and fatigue [4,15] in the arms, making this method useful only for short-term use. Touch-screen devices are also used as alternatives to the mouse, however occlusion and the fat finger problem [26] make them a good option mostly for non-productivity related tasks, kiosk terminals [4] and single point interactions [12].

Other efforts to replace the mouse include concepts that fuse the mouse and keyboard. However, these have reported mostly in patent filings, and none have been commercialized or evaluated. One such example is the "computer keyboard pointing device" [6] that uses a small cross-shaped touch sensing device that is inserted in between the keys Y,U,H and J on a QWERTY keyboard. This concept requires the user to perform pointing tasks on a very narrow and specific area of the keyboard by repeatedly swiping her finger. Only horizontal or vertical movements can be performed, and not diagonal ones. This implies that the movement of the pointer is not smooth and instead has a taxicab geometry [21]; *i.e.*, movements are only vertical or horizontal. Similarly, in the "keyboard pointing device" patent [23], the user controls the pointer by pressing on keyboard keys (*e.g.,* arrow keys). This accessibility function is already mainstream on all major computer operating systems, however it is generally aimed at people with motor control impairments who cannot use the mouse or trackpad [36].

Another device that leverages the keyboard to support pointing is ThumbSense [28], an interaction technique where the user clicks and left clicks using the f and j keys while pointing is still done using the trackpad and the thumb finger. ThumbSense performs marginally better than the trackpad and demonstrates some of the benefits of keeping the hands of the user at all times in the keyboard area. However, keeping one's hands on the keyboard, limits the area of the trackpad that the thumb can reach comfortably. Many patents with variations on touch sensitive keyboards exist [9][8,14,18,22], however, to our knowledge, the patented devices are not available to the public nor is there any information about their performance or implementation.

More recently, some researchers have published enhancements to the keyboard including touch [3] and proximity sensing capabilities [32]. However, their goal was not to create an alternative to existing pointing devices, but to make the keyboard into a gesture sensing device. The low spatial sensing resolution of those approaches may not be accurate enough for pointing tasks. In [33] a keyboard cover is used to augment the keyboard with touch sensing. Although the authors mention its applications to pointing, there is no evaluation of pointing performance. In addition, the sampling rate is low enough that lag may be a problem for pointing; homing time (mode switching) is not reported.

Discomfort, fatigue and injuries

Different research studies have found associations between wrist and hand pain and discomfort, and duration of mouse [16] and keyboard [2] use. However, participants who spend more time using the keyboard than the mouse at work, had significantly lower risk for CTS [2].

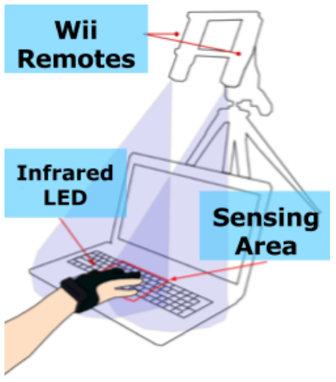
This suggests that pointing tasks performed directly on the keyboard may result in less discomfort, but possibly at the cost of worse performance for pointing tasks. Systematic evaluation comparing typical tasks (mixed typing and pointing) among pointing devices is rare in the literature and has focused primarily on performance. For example, Douglas [7] compares the performance of an isometric joystick (positioned under the 'J' key of a keyboard), to a mouse and trackpad, on a mixed pointing and typing task. This mixed task consists of first having the participant click on a target shown on the screen, and then as soon as the click occurs, a text box appears on the screen where the participant has to type a word shown on the screen as well. Once the participant types this word, the text entry box disappears and a new target for clicking appears on the screen. The isometric joystick decreased homing time, but increased movement time. As a result, performance was slower than using the mouse and trackpad. No measures of discomfort, fatigue or error rate were reported [7].

This body of work suggests that an ideal device would have the same low homing time as the joystick, but share the fast movement time of the mouse and the reduced discomfort of a keyboard. If such a device also had a low error rate, it could be competitive with the trackpad and mouse.

KEYBOARD-SURFACE INTERACTION

To achieve this ideal, we introduce keyboard-surface interaction (KSI). With KSI, the user performs pointing tasks while resting her hands on the keyboard's surface. The user's hand is the pointing device, and the surface of the keyboard is the sensing area in which the system detects pointing. The user can and should ideally keep her hands resting on the keyboard surface during pointing, to minimize discomfort, minimize hand movement (which we will show was an issue with mouse users in our performance study of KSI) and decrease risk for CTS by eliminating the need to use a mouse.

KSI has two modes: 1) typing mode, and 2) pointing mode. In typing mode, KSI does not recognize any hand movement as pointing. This mode was created to avoid unintentional pointing while typing. Switching between modes can be achieved by pressing a key once or making a gesture which switches from one mode to the other. In pointing mode, the user controls the movement of the pointer on the screen. A



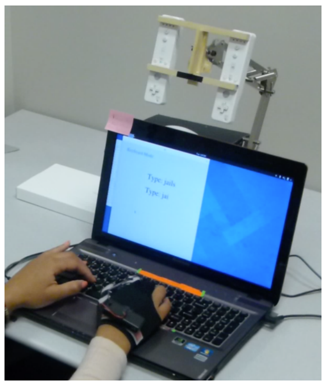


Figure 1. Left: Envisioned system. Right: Actual system and participant during a test

system can achieve this by tracking one of the fingers and mapping this movement to pointer movement.

EXPLORING THE DESIGN SPACE OF KSI

To explore the design space for KSI, we created a proof of concept device using infrared markers, we called this device Fingers. Fingers was developed in an iterative fashion, using a series of Fitts' law studies and prototypes to optimize its performance. Our iterative design process focused on improving error rate, homing time, movement time, and comfort. Our results are tightly related to the specifications of our sensing technology used. However, these specifications can be taken as the minimal design recommendations to obtain similar results even with a different sensing technology.

Overview of Fingers

The basic approach behind Fingers is hand tracking. Hand tracking can be difficult due to the multiple shapes that the hand can take and how quickly it can move. Moreover, the fastest marker-less hand tracking systems (Kinect devices, Creative Senz3D) operate at 30 samples per second, which is very low compared to the sampling rate of a commodity mouse (default of 100 samples per second). An alternative to hand tracking is measuring general movement flow [34], however it can easily get confused by other moving objects and body parts in view, which introduce noise.

Due to these limitations, we focused our efforts on a system that used markers to track the hand. Note that the use of markers is not inherent to Keyboard-Surface Interaction, but only to this particular implementation.

All of our development and evaluation of Fingers was conducted on a Lenovo laptop computer running Fedora 20, with a 2.2GHz Intel i7-2670QM processor, 16GB of RAM and a screen with 100px per square inch. The screen resolution was 1366x768 pixels. For all evaluations, we disabled the pointer acceleration feature following the standard on evaluating pointing devices [30].

Sensing

We used infrared LEDs as markers on the fingers of the user and tracked these LEDs using Wii Remotes (Figure 1), which include tracking capabilities. This is a simple and reliable

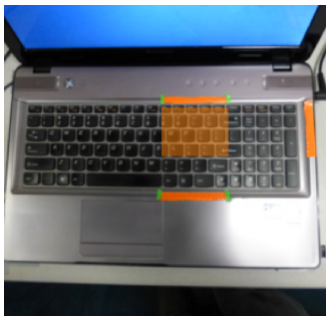


Figure 2. Fingers sensing area. Orange markers show the user the sensing area highlighted in translucent orange.

approach that allowed us to focus on other aspects of the system like evaluating and optimizing its performance. For tracking the infrared LEDs, we use Wii Remotes due to their price, readily available libraries, infrared tracking capabilities and high sampling rate (100Hz) compared to other infrared systems like the Kinect 1 and 2 (30Hz). Fingers has a resolution of 570x300 pixels over a sensing area of 3.3x2.5 inches (~140PPI) shown in Figure 2. This input surface is re-scaled so that every corner of the sensing area corresponds to a corner of the screen. By using a high power and high viewing angle infrared LED and the on-board tracking of the wiiRemote, we achieve perfect tracking accuracy for most movements and lose track momentarily for fast movements.

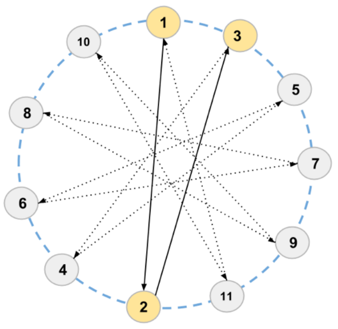


Figure 3. Target acquisition test: target location and order

We tested two movement-mapping strategies: *Relative* and *Absolute*. The *Absolute mapping* tracks a single LED located at the tip of the index finger. The movement of this LED is appropriately scaled and translated to move the pointer on the screen directly. The *Relative mapping* constantly aggregates the finger-tip's movement to the pointer's position on the screen only when the finger-tip is touching the keyboard's surface. This mapping is the same used by mouse and the trackpad. This mapping requires depth estimation to know when the keyboard surface was touched and uses a second Wii Remote and a simplified stereo vision model [25]. Depth estimation requires calibration for each user. For calibration, a participant moves her index finger while touching the desired sensing area over the keyboard while Fingers captures a 3D cloud of points from which the 3D position of the keyboard surface is estimated using least squares regression.

Fingers supports a sensing area of roughly 5 by 3 inches, acting as a trackpad. The area and size were selected empirically because it produces the best tracking, has an aspect ratio close to that of the screen, and is comparable in size to the trackpad's sensing area.

We used key presses for clicking and switching (between typing and pointing). By using key presses, the user's hand does not have to leave the keyboard and the potential for error caused by gesture recognition is eliminated. For clicking we chose the space bar due to its large size and easy access. We restricted clicking with the spacebar to be performed only with the non-tracked hand, to avoid mis-clicks caused by the movement of the hand while clicking and pointing with the same hand. We selected the switching key during our optimization process, as described next.

Optimization of Design Parameters

We conducted a series of studies to further optimize the performance of Fingers. These studies helped us to determine the best performing movement-mapping strategy (relative or absolute) and the fastest keys for switching between typing and pointing mode.

Method

Each study followed the same basic method, a Fitts' law study. Based on best practices [30], targets appeared on the screen as shown in Figure 3 with a fixed inter-target distance of 400 pixels. In addition, between consecutive clicks, participants were asked to type a short word (4-6 characters) that appeared on the screen. A similar test was used in [7] and we refer to this as the modified target acquisition test.

Each test consisted of 8 blocks of 36 targets each, 12 targets for each index of difficulty tested. In each block, a different sequence of index of difficulty (ID) was presented. We chose ID using Shannon's formulation [30] with index of difficulty of 3,4 and 5. We manipulated ID by modifying target sizes, keeping the distance between targets constant as suggested by [13]. To handle order effects, we randomized the presentation order of the blocks by index of difficulty.

Study 1: Absolute vs. Relative mapping

To compare Absolute mapping with Relative mapping, we used the Modified Target acquisition test. Twelve participants (5 males and 7 females) used Fingers in both the *Absolute* and *Relative* modes. To handle order effects due to the mapping used (*Relative vs. Absolute*), half the participants started with Relative mapping, while the other half started with Absolute mapping. We removed outliers that were more than three standard deviations from the mean, as described in [30].

Results: Movement time over the 8 blocks followed the power law of practice for both *Relative* (R^2=0.911) and *Absolute* (R^2=0.926), revealing strong learning effects. Thus, only the last block for each mode was used to compare movement time. In addition, movement time did not follow a normal distribution. This was confirmed with a Shapiro-Wilk normality test for *Absolute* (W=0.83, p-value<0.001) and *Relative* (W=0.76 p-value<0.001). *Absolute* movement time (Median time = 1.36 s) was faster than *Relative* (Median time = 1.13 s) with high significance on a Wilcoxon Signed Rank test p=0.00048, Z=-3.05 and size effect of 0.88).

One reason *Relative* movement time was slower than Absolute was that the depth estimation did not work as well when the participant moved her hands quickly. This made the detection of touch on the surface of the keyboard noisy.

When describing the study, we asked participants to rest their hands on the keyboard. This resulted in their hands being more relaxed and hence minimized trembling and other fatigue artifacts. Most participants initially were uncomfortable resting their hand on the keyboard. Most adjusted after the first block and all participants mastered it by the end of the study. Initially, some participants kept their index finger touching the keyboard while keeping their remaining fingers in the "air", much like one uses a trackpad. However, we prompted the participants to relax their hand and rest all of the fingers in their natural posture over the keyboard by demonstrating that this did not effect pointer use, unlike a trackpad.

Study 2: Switching key

To identify the best key for switching between typing and pointing, we used the Modified Target acquisition test, with participants completing 8 blocks for each key tested. We chose to compare the 'alt' key and 'tab' key since both are easily reachable (by the left hand's thumb and ring finger, respectively). As our goal was to provide guidance on which finger and keyboard area was best for switching, we did not address issues such as picking a rarely used key or key combination. Instead, we made sure that the typing tasks involved in this study did not require either switching key being tested. The same twelve participants from Study 1 completed the study but on a different day.

To handle order effects due to the key used (alt *vs.* tab), half the participants started with the alt key and the other half started with the tab key. Outliers more than three standard deviations from the mean were removed [30].

Results: We found that the tab key (Median time=0.27s) was faster than the alt key (Median time=0.33s). Homing time for the two keys did not follow a normal distribution, using a Shapiro-Wilk normality test (alt key W=0.73, p-value<0.001 and tab key W=0.75 p-value<0.001). The difference was significant using a Wilcoxon Signed Rank test (p-value=0.04 Z=2.01 with a size effect of 0.63).

Evaluation of performance and discomfort

Using the results from our previous studies, we evaluated the performance of Fingers against a trackpad and mouse. We used Fingers with Absolute mapping and the tab key for switching between typing and pointing,. We used the laptop's trackpad in the *Trackpad* condition and a standard optical Microsoft Mouse in the *Mouse* condition (with the acceleration feature deactivated for all devices). The study used the Modified Target acquisition test as in our optimization studies, repeated once with each device

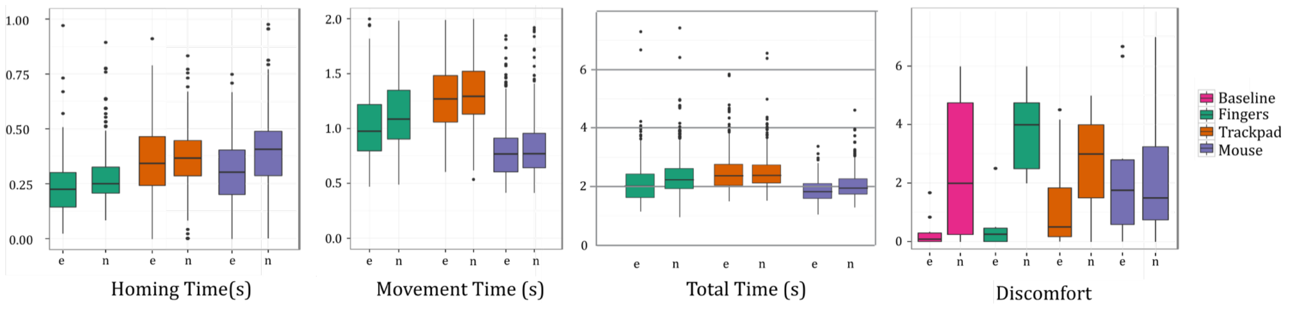


Figure 4: Box plots for the different performance metrics across devices, Novice (n) and Expert (e) users. The different units are: Time in seconds and discomfort total average score from 0 to 10.

(*Fingers, Mouse,* and *Trackpad).* The order of conditions was counterbalanced. In this test participants were asked to operate both the mouse and trackpad as they normally would with their own computers.

We recruited twenty-five participants: 10 experienced participants (from our earlier pilot studies on Fingers) and 15 novice participants. The experienced group (*Expert*) consisted of 4 males and 6 females, with ages ranging from 19 to 42. They had an accumulated experience with Fingers of 32 blocks, 16 from the absolute vs relative study and 16 from the switching key study. This final evaluation was conducted about two weeks after the prior studies. The novice group (*Novice*) consisted of 6 males and 9 females, with ages ranging from 18 to 47.

Participants filled out a demographic and a discomfort questionnaire. Discomfort is difficult to measure objectively. For example, measuring muscle fatigue requires special equipment to excite a muscle and measure its recovery time after exertion [19]. It also requires sensing equipment capable of measuring the forces exerted by the muscle [19]. Given these complications, we decided to use only subjective measures of fatigue, namely the perceived discomfort rating scale from [20]. Participants are asked about their current level of discomfort from 0 (None) to 10 (Extreme) for 6 different parts of the arm commonly used while pointing: hand, wrist, forearm, elbow, upper-arm and shoulder.

For each device, after a short practice, participants completed 8 blocks of tests and again filled out the discomfort survey. Participants rested 5 minutes or longer to remove any effects of fatigue before starting with the next device. Following testing with all 3 devices, participants were asked what their perceptions of Fingers. They received ($US) 20 for their time; the test took between 1-1.5 hours.

Measures

To assess performance, we computed *Homing time, Return time* and *Movement time* (in seconds). We define *Return time* as the time elapsed between when typing starts after the user has clicked on the on-screen target; it is a measure of the time it takes the hand to move from the pointing device to typing (*i.e.*, the opposite of homing time). We also calculated *Total time*, as the sum of *homing time*, *movement time* and *return time*. *Error rate* was calculated as the mean of the median number of errors for each index of difficulty for each device and user type. We removed outliers more than three standard deviations from the mean [30] and calculated median homing, movement and return time across the different targets and mean homing, movement and return time across different indexes of difficulty.

We calculated *discomfort* by averaging the 6 discomfort rates for the different parts of the arm and subtracting the baseline discomfort rate collected before the study began. The last step was performed to account for any discomfort unrelated to the experiment. We calculated average discomfort for each condition, for expert and novice users.

Finally, we measured *preference* by extracting commonly mentioned words and phrases from the open-ended question about what participants liked about Fingers. We grouped terms into themes. For example, 'responsive', 'effective', 'smooth', 'fast', and 'precise' were coded as *performance*. Other common themes (mentioned by more than six participants) were *movement time* and *reduced fatigue.*

Analysis Method

We tested for learning effect on homing time and movement time using fit to the power law of learning. Novice users' homing time showed a learning effect for Fingers (R^2=0.94), and for the mouse (R^2=0.71), but not the trackpad. Expert users did not show a learning effect for any measure. Thus, for novice users we only looked at the last block for homing time of Fingers and the mouse. For all other conditions, we computed the average across blocks.

A Shapiro-Wilk normality test showed that none of our movement time data followed a normal distribution, at a level α=0.05. Because the data was not normal, we used a Wilcoxon signed rank test in each case to measure whether there was a significant difference between the devices for each metric. All significance values presented are from these comparisons.

The median scores for homing time, movement time, and discomfort are shown in Figure 4. Bars with an *e* show expert performance while *n* indicates novice performance.

Results

Overall, Fingers has better performance with expert users than that of the trackpad, for both time to perform pointing actions and perceived comfort levels. When compared to the mouse, it has mixed performance with respect to the time to perform pointing actions, but has better perceived comfort levels. In addition, Fingers had the same error rate as both the trackpad and the mouse. Because there was no significant difference in the error rates across the devices (which varied from 0.033 to 0.09), we will not discuss error rate further.

Homing Time

Fingers was about a tenth of a second faster than the mouse and trackpad for experts, and novices achieved expert homing performance by the final block of the study. Expert median homing time was 0.23s, while mouse and trackpad both required over 0.31s (a significant difference: Z=-2.7 for trackpad; Z=-1.9 for mouse; p<.01 in both cases). For novice users, the learning effect for homing time was strong (power law of practice with R^2=0.94) and their performance was 0.27s for Fingers, 0.35s for the trackpad and 0.37s for the mouse by the final block (the block we will present the results from), as can be seen in Figure 4. As mentioned earlier, experts did not have a learning effect on any devices for homing time, thus we conclude that Fingers provides the best homing time.

When we examine in detail the distribution of homing time for the trackpad and the mouse, we found that there was an unusual peak centered over the zero bin in our histogram. This explains in part why the homing time for the mouse and trackpad does not follow a normal distribution. In the discussion section, we will describe why the peak occurred.

As described earlier, homing time is in part the time required to switch modes plus reaction time. When we asked participants what they disliked about Fingers, most participants (14/25) did not like to use the tab key to switch between typing and pointing. Thus, there may be further opportunity to improve homing time with an alternative key.

Movement time

In the case of *movement time*, the mouse was fastest, with a median time of 0.78s for novices and 0.73s for experts. Fingers had median movement time of 1.18s for novices and 0.96s for experts and the Trackpad had a time of 1.41s for novices and 1.33s for experts. The difference between the mouse and Fingers is significant (Z=3.4 for novices and Z=2.8 for experts, p<.001 in both cases). The difference between the trackpad and Finger is significant (Z=-2.3, for novices and Z=-2.7, p<0.01 in both cases).

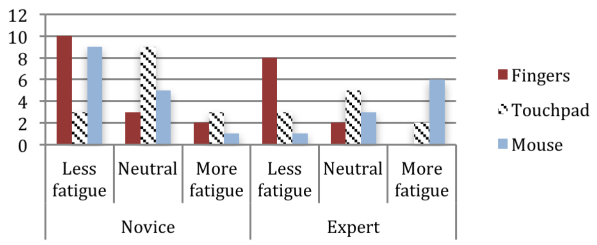


Figure 5. Summary of responses to the question: Please rate the level of fatigue induced by the device in this test.

Return time

For novice users, Fingers had the highest return time (0.79s), which is statistically comparable to the mouse (0.74s, p>0.9) but slower than the trackpad (0.65s, p<0.0005). For expert users, Fingers (0.73s) was statistically comparable to both the mouse (0.72, p>0.6) and the trackpad (0.67s, p>0.19).

Table 1. Total movement time across devices

	Fingers		Trackpad		Mouse	
	Novice	Expert	Novice	Expert	Novice	Expert
Total time	2.21s	1.97s	2.33s	2.29s	1.91s	1.77s

Total movement time

For novice users, Fingers had a statistically comparable *total* movement time (2.21s) with the trackpad (2.33s, p>0.05) although it was slower than the mouse (1.91s, p<0.01). For expert users, Fingers (1.97s) is statistically faster than the trackpad (2.29s, p<0.005), although slower than the mouse (1.77s, p<0.005).

Discomfort rate

Novices' discomfort did not differ across devices. However, in the post-study survey, they perceived both Fingers and the mouse as causing less fatigue (Figure 4, right).

In contrast, expert participants rated both Fingers (median discomfort=0.0) and the trackpad (median discomfort=0.4) as causing significantly less discomfort than the mouse (average discomfort=1.5) (Z=-2.3, p<.01 in both cases). In the post-study survey, expert users favored Fingers, rated the trackpad as neutral and rated the mouse as causing the most fatigue. Figure 5 shows the number of participants rating each device as less, neutral, or more fatiguing.

Discussion

The learning curve was fairly rapid, and only affected homing time, which suggests that Fingers was quick and natural to adopt. In expert use, Fingers outperformed the trackpad on every measure and outperformed the mouse on homing time and discomfort. Although it performed worse than the mouse on movement time, its gains in homing time allowed it to come within 9% of mouse performance on total movement time, with no increase in error rate. This performance comes at a relatively cheap cost (less than US$50 for our relatively simple marker-based prototype).

As mentioned in the results section, we observed some homing times of zero for the trackpad and mouse. This observation can be explained by some of our participants' behavior: In these cases, the participant reaches for the trackpad or mouse (and moves the pointer) with the right hand while still typing with the left hand. This only occurred if the last letters of the word on the screen to be typed were located on the left part of the keyboard. This behavior had a higher frequency for experts (3% of all blocks) than for novice users (1% of all blocks).

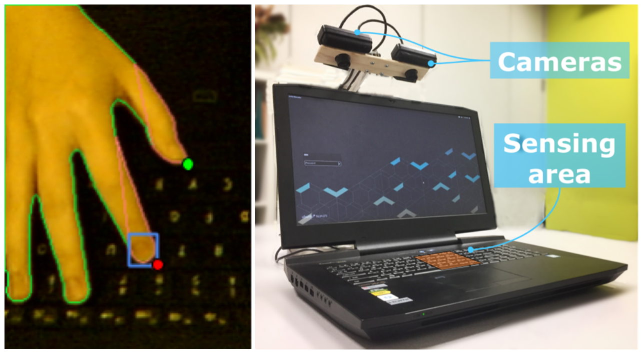


Figure 6. Left: Index finger tracking box, the blue box surrounds the tip of the index finger and the estimated tip is indicated in red. The thumb is correctly identified with the green dot. Right: Full IndexSense system

One unexpected result is that novice users did not perceive a difference in discomfort between Fingers and the mouse. We believe the novelty of using Fingers causes discomfort that is on par with using a mouse. In support of this hypothesis we observed that expert users rated Fingers to be more comfortable to use than the mouse. We hypothesize this is because users learned to use Fingers in a more optimal way.

As we learned from our qualitative survey, our participants did not like using the tab key for switching. This was surprising since the quantitative results for homing time are very good in general. Participants found the tab key confusing to use given its regular use for indenting while typing during daily computing tasks. This problem could be reduced in three different ways: 1) Creating a custom key for switching on KSI devices to avoid confusion. 2) Using the tab key to switch only from typing to pointing. To switch from pointing to typing, the user can simply start typing. 3) Eliminating completely the need for a switch. This may be possible by detecting dragging of the hand over the keyboard surface. This could not be achieved with Fingers due to our limited sampling rate, which reduces the precision of our depth estimation.

While participants did not complain in our surveys about the glove, the use of the glove on which the LED and battery are mounted could be a problem over the long run due to wear and tear, weight caused by the battery pack and heat from wearing a glove. Also, this approach would force users to wear the glove every time they wanted to perform a pointing task even for short tasks.

INDEXSENSE: A MARKER-LESS KSI DEVICE

Once we established the benefits of KSI, and to address the issues that arose with Fingers, we developed a marker-less, computer vision-based solution, IndexSense. IndexSense increases deployability and shows that KSI devices are viable pointing devices outside a lab environment.

Fingers achieved a spatial resolution of 140 pixels per inch (570x300 pixels, 100 frames per second). We used these metrics as our baseline goal for building a marker-less KSI device. Due to the speed constraint (wanting >=100fps), many commercially available sensors were discarded including depth sensing cameras, which at best can work at 60 fps. Although there are reports of the Kinect2 running at 120fps [31], depth sensing technology has a fundamental drawback that it cannot differentiate objects from cluttered backgrounds; the background must be empty space for it to work and recognize the user's hands properly. In our case the background is the keyboard's surface and hence depth sensing cameras are not suitable for our task.

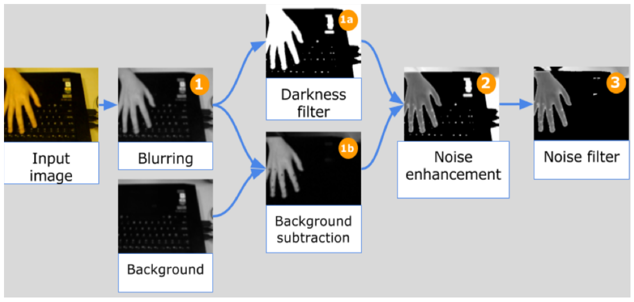


Figure 7. Hand recognition process.

For these reasons, we decided to implement our own computer vision-based system. For our implementation, we take advantage of having the keyboard's surface as a static background and to recognize the user's hand, find and track the index finger, and estimate whether this finger is touching the keyboard's surface.

All of our development and evaluation for IndexSense was conducted on a Bonovo (Systems76) laptop computer running Ubuntu 16.04, with a 4.5GHz Intel i7-7700k processor, 64GB of RAM. The screen resolution was 2722x1476 pixels. Our implementation used open cv 3.2.0 dev and python 2.7.

Sensing

We used two Sony PlayStation Eye cameras with a resolution of 320x240 and 189fps. The main stages of our vision method are calibration, hand recognition, index finger recognition and tracking. The setup can be shown in Figure 6 right. In the calibration stage, general parameters for functioning of the cameras are estimated. Hand tracking and finger tracking are performed first to determine if there is a hand and the image, and then the index finger-tip is located. Once the tip of the finger is located in the images from the two cameras, the difference in location is used to estimate depth using the same technique as Fingers.

Calibration

The goal of this stage is to minimize the effect of lighting on different skin tones. To do that, our system needs to be able to filter out the user's hand from the background. Here we use a key observation about using the keyboard of a laptop: the background is mostly the same with changes only due to illumination. Despite this advantage, due to shadows and other lighting artifacts, pure background subtraction does not always generate the desired results. To address this, first we obtain 20 images of the keyboard without the user's hands on it; this is done each time IndexSense is initialized. These images are then averaged and this averaged image is used for background subtraction. Once the background average image is collected, we have the user place their hand over a black fabric laid on top of the keyboard; we refer to this image as the calibration image. Next, the camera's gain is adjusted so that the gray scale histogram of the calibration image matches a pre-specified template on file. After that, the user is asked to twist and bend her index finger simulating normal positions and orientations of the finger that can make tracking of the index finger difficult. This gives us a lower-bound threshold for brightness. The next step is to find the upper-bound threshold that minimizes the difference between the background image initially collected and the calibration image. The last step is to locate the index finger in the calibration image and estimate its distance to the keyboard, providing a rough estimate of the finger's height. This height is used to re-estimate a plane of the keyboard already stored on file. To estimate the keyboard plane, we use the same depth estimation procedure used for Fingers.

Hand recognition

The hand recognition procedure is shown in Figure 7. The first step is to blur the image to reduce fine noise. After that, two different steps are taken in parallel. In step 1a, a darkness filter is applied, since the background is dark, most of the remaining image should consist of the hand of the user plus light reflected on some of the keys. In step 1b, the background is subtracted, and the resulting image contains mostly the hand of the user with some missing parts usually in the tips of the fingers due to shadows and darker skin tones. Some of these missing details of the hand contour are contained in the image resulting from step 1a. The images resulting from step 1a and 1b are subtracted, causing noise pixels to have a higher value than the hands. This subtraction results in shadows receiving a negative value (and are easily filtered out), and missing details of the hand from step 1b are recovered. Finally, noise is filtered out using the high threshold value estimated in calibration. After all steps, the hand, and more importantly, hand contour, is well defined.

Index finger recognition

After recognizing the hand, a contour is drawn around it and peaks and valleys are estimated over this contour. Peaks are identified as finger tips, and the peak closest to the upper horizontal part of the image is identified as the thumb. The next finger-tip found in the clockwise direction is identified as the index finger. After identifying the index finger, the next step is to identify a specific part of the index finger. Initially we tried to track the tip of the index finger, however tracking the tip can sometimes be unstable. For that reason, instead of trying to identify the tip of the index finger, we estimate a square that contains the entire upper part of the index finger and use the lower right corner of that square as our reference point used for depth estimation and 2d localization, as shown in Figure 6 left.

Index finger tracking

We track the index finger by using the estimated point of reference from the last step. We make use of the openCV Kalman filter to smooth the resulting movement estimate of the index finger. Additionally, we also perform visual tracking of the index finger using a kernelized correlation filter [17] to disambiguate situations in which the index finger cannot be identified properly. These situations usually occur when the thumb or middle finger are misidentified as the index finger. This causes the estimated position of the index finger to change abruptly. This problem is usually caused by fingers occluding or touching each other; when this is detected, IndexSense immediately requests the location of the index finger from the finger tracker. The finger tracker however loses track when there are fast movements; in those cases, the tracker relies on index finger recognition. Thus, both systems complement each other.

Performance Evaluation and Method

To evaluate the performance of IndexSense we repeated the same experiment conducted to evaluate Fingers and compared IndexSense with *Absolute* Mapping, Trackpad and Mouse. We had 15 participants for each experiment. For the first experiment, we had 8 female and 7 male participants and their age range was 18 to 28. Acceleration was disabled for all devices and the sizes of the targets were adjusted to keep the same index of difficulty as in the evaluation of Fingers, accounting for the higher screen resolution of the laptop being used in this study. We followed the same protocol as for testing Fingers, but did not ask participants to fill out the discomfort survey. We expect discomfort rates for IndexSense to be very similar to Fingers or better, since IndexSense does not require the user to wear a glove. The measures used were *Movement time*, *Homing time, Return time* and *Total movement time*. Like with Fingers, there was no significant difference in the error rates across the devices.

Data Preparation

After data collection, we filtered out outliers as described in [30]. Afterwards, we calculated the median homing, movement and return time across the different targets, and the mean times across different indexes of difficulty. Movement time, homing time and return time for both experiments showed a learning effect that varied from moderate (mouse R^2=0.64) to strong (IndexSense R^2=0.98). For this reason, we only used the last block of each condition for all subsequent analysis. As before, we used a Wilcoxon signed rank test in each case to measure whether there was a significant difference between the devices for each metric. All significance values presented are from these comparisons. In table 2 we show a breakdown of the results for each of the measures considered.

In summary, IndexSense had the highest (1.50s) Movement time. For Homing time, IndexSense had the lowest (0.23s). For the return time, IndexSense had the lowest return time (0.29s). Total movement time, IndexSense (2.04s) was faster than the Trackpad (2.27s, $p<0.01$) and as fast as the Mouse

(1.94s, p>0.3). Finally, for the error rate, IndexSense had 1.1% while the mouse and touchpad had 0.0%.

Table 2. Summary of measurements for all devices

	IndexSense	Mouse	Trackpad
Movement time	1.5s	0.76s**	1.3s*
Homing time	0.23s	0.45s**	0.37s**
Return time	0.29s	0.72s**	0.67s**
Total time	2.04s	1.94s	2.27**
Error rate	1.1%	0.0%**	0.0%**

** $p<0.01$, **$p<0.001$* when compared to indexSense,

Discussion

IndexSense's total movement time is statistically comparable to the Mouse. While our study is limited in terms of the tasks we assigned and the number of participants included, this result is very exciting. It highlights the opportunity to improve upon the mouse in terms of comfort, while maintaining the high performance that has been uniquely characteristic of the mouse. In addition, our results demonstrate that we can take advantage of a mostly unused interaction surface – the top of the keyboard.

It is worth noting that IndexSense has double the frame rate of Fingers, making it capable of keeping track of the index finger even for very fast movements. We correctly hypothesized earlier that IndexSense would have very low homing and return times. Thus, despite not having the best movement time, it has better performance than the trackpad and statistically comparable performance to the mouse. Although our implementation is not faster than the mouse, other implementations could potentially have higher performance. In addition, with more experience and practice, users of KSI devices could have improved performance and error rate, potentially even higher than with the mouse.

Implementing future KSI devices

KSI devices could be activated through gestures which could make homing time almost zero making it faster than the mouse. For example, assuming a homing time for IndexSense of 0.1 seconds and keeping movement time the same, IndexSense becomes 10% faster than the mouse.

Through our devices, Fingers and IndexSense, we have shown that KSI is promising, and we imagine a future in which we no longer need a trackpad or a mouse. This, would free up space on laptop computers and improve the human-computer experience.

During our evaluation of KSI devices, we did not test for dragging or right click; however, they are easy to implement. Right click could be supported by using the control key instead of the space bar or by pressing space bar + another key. Dragging can be implemented in the same way it is done with a mouse, by checking for how long the user has sustained a click (in this case with the space bar).

KSI devices should be investigated further with respect to acceleration features: Both the mouse and trackpad have an acceleration feature that increases the movement of the pointer on the screen relative to the acceleration of the device. This functionality decreases further total movement time while pointing by enabling long movement to be done very quickly while slow movements are still slow for the user (acceleration profile is not constant over the movement speed). For KSI devices with relative mapping, using the acceleration feature is possible since the movement of the pointer is an aggregate of the hand movement and not the hand position. This is not the case for the absolute mapping; however, absolute mapping allows for exceedingly fast hand movements so acceleration may not be necessary after all.

In our evaluation of KSI devices, we used an almost equal mix of typing and pointing that reflects previous studies. However, this does not tell us how a KSI device would perform at different ratios of typing *vs*. pointing. We expect that when users are mostly pointing they will experience decreased discomfort compared to the mouse and trackpad but no other gains in terms of performance. When users are mostly typing, we would not expect to see much difference compared to the mouse and trackpad since KSI devices do not interfere with typing. While we would expect modest gains, and at worst, no difference at these "extremes, further experimentation with KSI devices is needed to confirm this.

Finally, there is tremendous opportunity for re-imagining pointing entirely. In our KSI devices, we were limited by the current technology; however, we imagine a KSI device that uses the entire keyboard area. Not only could the user use both hands for performing interactions but specific regions of the keyboard could have special purposes. Positioning the finger in specific areas could, for example, change the painting tool in a painting application. By sliding the left hand over the top of the keyboard, the size of the painting tool can be changed. A KSI device could even support a mix of short mid-air gestures plus surface gestures to make our computing experience much more natural, compelling and equally important, easy on our hands.

CONCLUSIONS

In this paper, we introduced the concept of Keyboard-Surface Interaction, and evaluated it through Fingers, our proof of concept and IndexSense, our improved marker-less KSI device. Both devices have performance comparable to the trackpad, and IndexSense with absolute mapping has performance on par with the mouse. Results from discomfort surveys indicate that KSI devices induce less discomfort in users. This means that KSI devices can be a very good alternative to the trackpad and the mouse, if comfort is a priority. Furthermore, we have shown that the surface of the keyboard can be used to do much more than has been explored up to date. In this paper, we explored pointing tasks, but the potential is there to use the keyboard surface as a multi-touch sensing device on which to scroll, zoom and perform touch gestures, similar to a trackpad or touchscreen.

REFERENCES

1. Taher Amer, Andy Cockburn, Richard Green, and Grant Odgers. 2007. Evaluating swiftpoint as a mobile device for direct manipulation input. *Conferences in Research and Practice in Information Technology Series*, 63–70.

2. Isam Atroshi, Christina Gummesson, Ewald Ornstein, Ragnar Johnsson, and Jonas Ranstam. 2007. Carpal tunnel syndrome and keyboard use at work: a population-based study. *Arthritis and rheumatism* 56, 11: 3620–5. http://doi.org/10.1002/art.22956

3. Florian Block, Hans Gellersen, and Nicolas Villar. 2010. Touch-display keyboards. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10). ACM, New York, NY, USA*, 1145–1154. http://doi.org/10.1145/1753326.1753498

4. Sebastian Boring, Marko Jurmu, and Andreas Butz. 2009. Scroll, tilt or move it: using mobile phones to continuously control pointers on large public displays. *In Proceedings of the 21st Annual Conference of the Australian Computer-Human Interaction Special Interest Group: Design: Open 24/7 (OZCHI '09). ACM, New York, NY, USA,* 161–168. http://doi.org/10.1145/1738826.1738853

5. Stuart K. Card, William K. English, and Betty J. Burr. 1978. Evaluation of Mouse, Rate-Controlled Isometric Joystick, Step Keys, and Text Keys for Text Selection on a CRT. *In Human-computer interaction, R. M. Baecker and W. A. S. Buxton (Eds.). Morgan Kaufmann Publishers Inc., San Francisco, CA, USA* 21, 8: 386–392. http://doi.org/10.1080/00140137808931762

6. Donald S. Santilli. 1997. Computer keyboard pointing device. Retrieved from http://www.google.com/patents/US5675361

7. Sarah A. Douglas and Anant Kartik Mithal. 1994. The effect of reducing homing time on the speed of a finger-controlled isometric pointing device. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '94), Beth Adelson, Susan Dumais, and Judith Olson (Eds.). ACM, New York, NY, USA*: 411–416. http://doi.org/10.1145/191666.191805

8. Cliff Edwards. 2013. Gesture-enabled keyboard and associated apparatus and computer-readable storage medium. Retrieved from http://www.google.com/patents/US8432301

9. John Elias and Steven Martisauskas. 2015. Fusion keyboard. Retrieved from https://www.google.com/patents/US20130063286

10. Mircea Fagarasanu and Shrawan Kumar. 2003. Carpal tunnel syndrome due to keyboarding and mouse tasks: a review. *International Journal of Industrial Ergonomics* 31, 2: 119–136. http://doi.org/10.1016/S0169-8141(02)00180-4

11. W. Fallot-Burghardt, M Fjeld, C Speirs, S Ziegenspeck, H Krueger, and T Läubli. 2006. Touch&amp;Type. *Proceedings of the 4th Nordic conference on Human-computer interaction changing roles - NordiCHI '06*, ACM Press, 465–468. http://doi.org/10.1145/1182475.1182538

12. Clifton Forlines, Daniel Wigdor, Chia Shen, and Ravin Balakrishnan. 2007. Direct-touch vs. mouse input for tabletop displays. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '07). ACM, New York, NY, USA*: 647. http://doi.org/10.1145/1240624.1240726

13. Yves Guiard. 2009. The problem of consistency in the design of Fitts' law experiments: Consider either target distance and width or movement form and scale. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09). ACM, New York, NY, USA*, 1809–1818. http://doi.org/http://doi.acm.org/10.1145/1518701.1518980

14. William Hamburgen, Glen Murphy, Andrew Bowers, et al. 2014. Keyboard integrated with trackpad. Retrieved from http://www.google.com/patents/US8754854

15. Chris Harrison, Shilpa Ramamurthy, and Scott E. Hudson. 2012. On-body interaction: armed and dangerous. *In Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction (TEI '12), Stephen N. Spencer (Ed.). ACM, New York, NY, USA*: 69–76. http://doi.org/10.1145/2148131.2148148

16. Alan Hedge. 2003. Computer use and risk of carpal tunnel syndrome. *JAMA : the journal of the American Medical Association 290*, 1854–1855. http://doi.org/10.1001/jama.290.14.1854-a

17. Joao F. Henriques, Rui Caseiro, Pedro Martins, and Jorge Batista. 2015. High-Speed Tracking with Kernelized Correlation Filters. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37, 3: 583–596. http://doi.org/10.1109/TPAMI.2014.2345390

18. Satoshi Hosoya, Hiroaki Agata, and Fusanobu Nakamura. 2012. Touchpad and keyboard. Retrieved from http://www.google.com/patents/US20120306752

19. Peter W. Johnson, Steven L. Lehman, and David M. Rempel. 1996. Measuring muscle fatigue during computer mouse use. *Proceedings of 18th Annual International Conference of the IEEE Engineering in Medicine and Biology Society*, 1454–1455. http://doi.org/10.1109/IEMBS.1996.647501

20. Kihyo Jung. 2014. Effects of slanted ergonomic mice on task performance and subjective responses. *Applied Ergonomics* 45, 3: 450–455. http://doi.org/10.1016/j.apergo.2013.06.004

21. Eugene F. Krause. 1987. *Taxicab geometry: an adventure in non-Euclidean geometry*. Dover Publ., New York. Retrieved from https://cds.cern.ch/record/1547746

22. Glen C. Larsen and Steven N. Bathiche. 2010. Keyboard with a touchpad layer on keys. Retrieved from http://www.google.com/patents/US7659887

23. Goodman Michael Ken and Noorbehesht Charles Frederick, Raasch Farzad. 2000. Keyboard pointing device. Retrieved from

https://www.google.com/patents/US6100875

24. Pranav Mistry and Pattie Maes. 2011. Mouseless: A Computer Mouse as Small as Invisible. *In CHI '11 Extended Abstracts on Human Factors in Computing Systems (CHI EA '11). ACM, New York, NY, USA*, 1099–1104. http://doi.org/10.1145/1979742.1979715

25. Dynesh Nair. Simplified stereo vision system. Retrieved from http://www.techbriefs.com/component/content/article/23-ntb/features/feature-articles/14925

26. Alex Olwal, Steven Feiner, and Susanna Heyman. 2008. Rubbing and tapping for precise and rapid selection on touch-screen displays. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08). ACM, New York, NY, USA* 2008: 295–304. http://doi.org/10.1145/1357054.1357105

27. Prentke Romich Company. HeadMaster. Retrieved from http://www.indiana.edu/~iuadapts/technology/hardware/mice/headmaster.html

28. Jun Rekimoto. 2003. ThumbSense: Automatic input mode sensing for touchpad-based interactions. *CHI'03 Extended Abstracts on Human Factors in …*. Retrieved May 22, 2014 from http://dl.acm.org/citation.cfm?id=766031

29. Jenna M. Shanis and Alan Hedge. 2003. Comparison of Mouse, Touchpad and Multitouch Input Technologies. *Proceedings of the Human Factors and Ergonomics Society Annual Meeting. SAGE Publications* 47, 4: 746–750. http://doi.org/10.1177/154193120304700418

30. R. William Soukoreff and I. Scott MacKenzie. 2004. Towards a standard for pointing device evaluation, perspectives on 27 years of Fitts' law research in HCI. *International Journal of Human-Computer Studies* 61, 6: 751–789. http://doi.org/10.1016/j.ijhcs.2004.09.001

31. Andrea Tagliasacchi, Matthias Schröder, and Anastasia Tkach. 2015. Robust Articulated-ICP for Real-Time Hand Tracking. *Eurographics Symposium on Geometry Processing*, 101–114. http://doi.org/10.1111/cgf.12700

32. Stuart Taylor, Cem Keskin, Otmar Hilliges, Shahram Izadi, and John Helmes. 2014. Type–Hover–Swipe in 96 Bytes: A Motion Sensing Mechanical Keyboard. *In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14). ACM, New York, NY, USA*. http://doi.org/10.1145/2556288.2557030

33. Ying-Chao Tung, Ta-Yang Cheng, Neng-Hao Yu, and Mike Y. Chen. 2014. FlickBoard: enabling trackpad interaction with automatic mode switching on a capacitive-sensing keyboard. *Proceedings of the adjunct publication of the 27th annual ACM symposium on User interface software and technology - UIST'14 Adjunct*, 107–108. http://doi.org/10.1145/2658779.2658799

34. Andrew D. Wilson and Edward Cutrell. 2005. Flowmouse: A computer vision-based pointing and gesture input device. *In Proceedings of the 2005 IFIP TC13 international conference on Human-Computer Interaction (INTERACT'05), Maria Francesca Costabile and Fabio Paternò (Eds.). Springer-Verlag, Berlin, Heidelberg* 3585: 565–578. http://doi.org/10.1007/11555261

35. AbleTech Assistive Technologies Inc. Retrieved from http://abletech.ca/products/computer-access/specialized-pointing-devices/

36. MouseKeys. Retrieved from http://www.ssbbartgroup.com/blog/mousekeys-vs-tabbing/