Eyes on the Road, Hands on the Wheel: Thumb-based Interaction Techniques for Input on Steering Wheels

Iván E. González,[1*] Jacob O. Wobbrock,[2] Duen Horng Chau,[1] Andrew Faulring[1] and Brad A. Myers[1]

[1] Human-Computer Interaction Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213
{ieg, dchau, faulring, bam}@cs.cmu.edu

[2] The Information School
University of Washington
Mary Gates Hall, Box 352840
Seattle, WA 98195-2840
wobbrock@u.washington.edu

Abstract

The increasing quantity and complexity of in-vehicle systems creates a demand for user interfaces which are suited to driving. The steering wheel is a common location for the placement of buttons to control navigation, entertainment, and environmental systems, but what about a small touchpad? To investigate this question, we embedded a Synaptics StampPad in a computer game steering wheel and evaluated seven methods for selecting from a list of over 3000 street names. Selection speed was measured while stationary and while driving a simulator. Results show that the EdgeWrite gestural text entry method is about 20% to 50% faster than selection-based text entry or direct list-selection methods. They also show that methods with slower selection speeds generally resulted in faster driving speeds. However, with EdgeWrite, participants were able to maintain their speed and avoid incidents while selecting and driving at the same time. Although an obvious choice for constrained input, on-screen keyboards generally performed quite poorly.

CR Categories: H.5.2. Information interfaces and presentation: User interfaces — *Input devices and strategies*.

Keywords: Interaction techniques, selection techniques, text entry, text input, gestures, driving, steering, distraction, touchpad, StampPad, thumb-based input.

1 Introduction

The automotive experience is increasingly being augmented with computerized devices. Cars come with built-in navigation and information systems, satellite radios, multimedia systems, and complex air conditioning and heating systems. For example, the BMW iDrive system controls over 700 functions [4]. Drivers are also bringing complex devices into cars, including cellular phones, MP3 players, and PDAs. Not surprisingly, the increase in devices in automobiles has resulted in a need for users to interact with them while stationary or even while driving. Today, these devices are controlled using physical knobs, sliders, buttons, and virtual controls like on-screen keyboards and soft buttons. But these mechanisms often require users' visual attention and can be quite difficult, or even dangerous, to operate while driving.

A recent survey by Privilege Insurance in England showed that navigation systems can be distracting while driving [15]. According to the survey, 11% of motorists with a navigation system start their journeys without entering their route. Of these, 53% admitted that they had to take their eyes off the road in order to input destination details while driving. These examples illustrate a need for effective interaction methods that can be used with the multitude of devices present in an automotive setting.

Eventually, speech recognition may help address aspects of this problem, but for now, speech is not adequate in practice because it is susceptible to environmental noises and vulnerable to individual differences. It also requires the speaker to remember specific language syntax, which may add to cognitive load. Furthermore, it is poor for continuous control tasks such as scrolling through a long list or panning a map to a new location. Although speech may help in the future, for the purposes of this investigation, we examine direct manipulation.

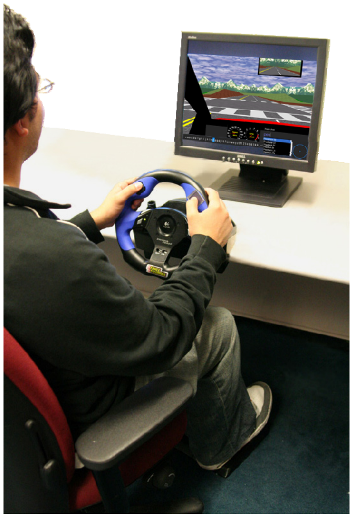


Figure 1. The first author performing input on a small touchpad with his right thumb during simulated driving. The touchpad is mounted in the surface of the wheel at the 2 o'clock position.

In this work, we focus on the common but complex task of *destination entry* for a navigation system. We conjecture that allowing users to keep their hands on the steering wheel might reduce the difficulty of destination entry when performed either by using text entry or by selecting from lists of street names. These techniques could also be used for selecting radio stations on satellite radio, which has over 170 channels, selecting songs from loaded MP3 players, or selecting contacts to dial from address books on cell phones. In the future, we expect that the automobile's control systems will be able to control handheld devices via Bluetooth or similar protocols. Therefore, a single input device that can be used for both built-in and brought-in devices may be useful.

In this paper, we present multiple thumb-based input techniques for destination entry using a small touchpad called a *StampPad* mounted where the right-hand thumb naturally grips the steering wheel (Figure 1). This use of the thumb keeps the driver in the preferred position with both hands on the steering wheel. We adapted popular interaction techniques from computers and consumer electronic devices for our thumb-based interaction techniques, which we evaluated in both stationary and simulated driv-

Graphics Interface Conference 2007
28-30 May, Montréal, Canada

[*] First author's current address: Microsoft Corp., One Microsoft Way, Redmond, WA 98052. ivangonz@microsoft.com

ing conditions. Techniques include "dialing," which is found on the Apple iPod, "clutching," which is common to laptop touchpads, "displacement," which is similar to how joysticks operate, and the EdgeWrite text entry method [24]. These techniques will be explained in more detail in later sections. The results of our study show that while none can really be considered safe, EdgeWrite performs much better than today's popular techniques such as on-screen keyboards.

2 Related Work

This work is related to previous studies of driving, text entry, and selection from lists. Each of these areas is reviewed below.

2.1 Some Previous Driving Studies

There has been a great deal of work on interaction with systems while driving. Studies have used either on-road driving or driving simulators with different levels of fidelity depending on the safety implications and level of maturity of the interface being evaluated.

The study by Dingus et al. [3] was the first to address the attentional demands of a navigation system. In that study, an electronic map displayed the position of the host vehicle and the location of the destination. The results showed that the attentional demands of most navigation tasks were similar to those of typical in-car tasks.

Schneider et al. [17] evaluated a system to reduce driving distraction during cell phone conversations. The system provided callers with a remote display of the driver's view and was evaluated using low fidelity and higher fidelity simulators. In our study, the same higher fidelity simulator is used.

Lee et al. [7] explored the effectiveness of a collision warning system while the driver interacted with an in-vehicle speech-based email system. That study used a driving simulator with a real vehicle cab and force feedback controls. They found that graded alerts were more trusted and had a greater safety margin, and that haptic alerts were perceived as less intrusive.

Arroyo et al. [1] designed a context-aware driving advisor and evaluated it in an instrumented research vehicle while on-road driving. They found that the frustration elicited by the driving advisor's negative feedback can be mitigated if the feedback is given in a scheduled manner and also includes positive feedback.

2.2 Text Entry

Many text entry systems in automobiles use on-screen keyboards, navigated with joysticks, bezel keys, arrow keys, remote controls, rotary knobs, or touch-screens. Text entry on handheld devices uses on-screen keyboards or gestural alphabets like Graffiti, tiny thumb keyboards like on Blackberry devices, or multiple tapping of cell phone keypads buttons. EdgeWrite [24] is a unistroke text entry method that has been implemented for handheld devices, joysticks, trackballs, touchpads, and other devices. However, one drawback of any gestural method is that it often takes longer to learn than most selection based methods.

2.3 Selection from Lists

Selection of items from a list has been studied since early graphical environments made selection possible with a pointing device. Beck et al. [2] compared different scrolling methods for lists, finding that familiarity with the list and its size were significant factors in selection time. A more recent investigation by Lehikoinen et al. [9] presented *BinScroll*, a rapid selection technique for alphanumeric lists based on binary search. Although these studies are 13 years apart, both recognize that selecting from a list is a recurring task that should be made more efficient. As computing devices have become more pervasive, the task of list selection has surfaced in new environments where traditional methods and devices are not an option. This investigation focuses on this challenge in the automotive setting.

2.4 Text Entry and Selection While Driving

Text entry while driving on a test track has been previously investigated. Tijera et al. [19] compared destination entry, cellular phone dialing, and radio tuning while driving on a test track with traffic. Four commercial navigation systems were evaluated: three visual-manual methods and a voice system. Mean entry times for the visual-manual methods tested ranged from 40 to 120 seconds. Dialing an unfamiliar 10-digit phone number on average took approximately 28 seconds, and tuning a radio using a seek function took on average 22 seconds. The mean number of lane departures was 0.9 when entering a destination, compared to ~0.15 when dialing a phone or tuning a radio. Notably, the speech input technique was associated with longer and more frequent glances away from the road to verify information about the destination.

Text entry and selection in automobiles has also been studied in a simulated environment. Holbrook [6] evaluated a Graffiti recognizer, a touch screen, and a remote control for selecting items from a list while simulated driving. In that study, all three input methods resulted in no apparent performance differences. The list selection was based on roughly three hundred items that were common knowledge to users (e.g., cereals, fruits). They were also divided into a two-level hierarchy. In contrast, our system tests a more difficult real-world task by having users select street names from lists of a more realistic size—about 3000 items—which is comparable to the number of streets in a medium-sized city (e.g., Pittsburgh, Pennsylvania).

3 Prototype Design and Implementation

3.1 Touch-Sensitive Steering Wheel Prototype

Many cars have buttons mounted on the steering wheel for controlling entertainment systems, cruise control, and environmental settings. Following this trend, we embedded a small Synaptics *StampPad* (2.8×3.2 cm) beneath the thumb's natural position on a Logitech Driving Force steering wheel for investigating thumb-based interaction techniques (Figure 2). Driving guidelines suggest that hands should be placed at the 2 o'clock and 10 o'clock positions when gripping a steering wheel [14]. Furthermore, the natural position of the thumb when grasping a device is often the best location for thumb-controlled input areas [5]. Accordingly, we placed the touch sensitive area at the 2 o'clock position for use by a right-handed user. The resulting prototype is a steering wheel with a touch-sensitive area flush with the surface and located where the right thumb grips the wheel. We built earlier prototypes using buttons and four-way directional pads, but these turned out to be more difficult to use.

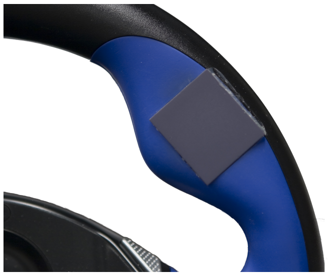
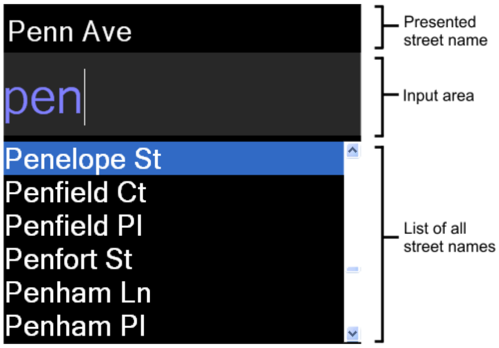


Figure 2. The Synaptics StampPad embedded flush with the surface of a Logitech Driving Force steering wheel.

The touch-sensitive steering wheel performed selection from a list of driving destinations displayed on the screen of the driving simulator (see section 3.2). The list-selection software supported selecting an item by either entering text or by direct-selection. Each of our seven destination entry techniques consisted of two components: an *interaction technique* (section 3.3) and a *keyboard technique* (section 3.4). The interaction technique determined how the user manipulated an on-screen selector, while the keyboard technique determined what kind of keyboard was used (if any) to enter text. For example, the "clutching" interaction technique could be used to select characters from an on-screen two-dimensional keyboard, which in turn sent key events to the list of driving destinations.

3.2 Destination List

With the on-screen list of destinations, the user moves a selector over the items in the list until the target is found. *How* the selector is moved is determined by the interaction technique and possibly an on-screen keyboard. Although a vertical list can have multiple levels to represent a taxonomy, the one used in our system consisted of a single-level list. Navigation systems today (e.g. Hertz NeverLost, Magellan PathMaster) select streets using single-level lists. Furthermore, hierarchical menus must allow users to select the individual items at each level, so single-level selection is still the most fundamental selection operation.



Penn Ave

pen

Penelope St
Penfield Ct
Penfield Pl
Penfort St
Penham Ln
Penham Pl

Figure 3. A presented target street ("Penn Ave"), current entry "pen", and current selection "Penelope St" as the first match for "pen".

The user changes the current selection by either moving the selector up and down, or, for some techniques, by text entry. When a text entry method is used to select from the list, the list displays the entered characters above it (Figure 3). Since the list of valid items is finite and known, the implementation automatically adds characters when the current prefix is unambiguous. For example, when the user enters the prefix "Penfi", the only matching items are "Penfield Ct" and "Penfield Pl", so the prefix "Penfi" is extended to "Penfield_" and a special tone is sounded to alert the user. After the prefix has been extended, the current selection is set to be the first match of the prefix. If the user backspaces after the prefix has been extended to "Penfield_", the system appropriately reverts to the prefix that *was present before the last character entry*, which would be "Penf".

3.3 Interaction Techniques

Interaction techniques allow the user to manipulate an on-screen selector by moving their thumb over the embedded StampPad. Thumb movement on the StampPad is translated to selector movement on the screen based on the nature of the interaction technique being used. In all interaction techniques, a "tap" on the touchpad enters the current selection.

Moving the selector through long lists is made practical with the implementation of transfer functions for some of the techniques. These functions map the finger speed on the touchpad to the speed of the selector.

3.3.1 Clutching

Clutching is familiar to most people from the way that a mouse is used on a desktop PC or a touchpad is used on a laptop. When clutching, users repeatedly stroke over the StampPad. Clutching allows for two-dimensional selector movement along either the horizontal or vertical axes.

We developed a transfer function through iterative pilot testing of the clutching technique by observing the fastest and slowest thumb-stroking speeds. We wanted to ensure that users could accurately select items when moving slowly while also being able to move quickly through a 3000-item list. The resulting transfer function is:

$$f(x) = \begin{cases} 0.07 \cdot x & 0 \le x < 20 \\ 1.2619^{x-20} - 1 & 20 \le x < 45 \end{cases}$$

The variable x represents the speed of the thumb on the StampPad measured in (pts/ms). The constants were derived empirically. The exponential growth of the second function branch allows for rapid scrolling through large lists with the constants chosen such that, at the fastest speed, only five strokes are needed to traverse the entire 3000-item list.

3.3.2 Dialing

Dialing has been recently popularized by the Apple iPod as a technique for selecting from song lists. This technique has also been used for text entry [16]. With the dialing technique, users move their fingers around the touchpad in a circular manner. To help users follow a circular path, we placed a plastic template with a round hole over the StampPad (Figure 4a). Dialing allows for one-dimensional movement of the selector by either clockwise or counter clockwise motion (Figure 4b).

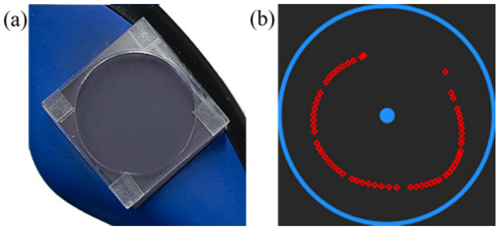


Figure 4. (a) A plastic template overlaid onto the StampPad for guiding circular movement. (b) Screen feedback when using the dialing technique.

The transfer function used for this interaction technique was modeled after the iPod, which operates at various speeds. Initially, the system operates at the slowest speed. Medium speed is triggered when the user maintains a speed of more than half a cycle per second for at least one full circle. The fastest speed is triggered when the user maintains a speed of more than one cycle per second for one full circle. The system can go from slow to fast without going through the middle speed. Through empirical testing we choose 20, 80 and 640 items per rotation for the slow, medium and fast speeds, respectively. A mathematical formulation of the transfer function is:

$$f\left(\Delta\theta_t\right) = \begin{cases} \frac{640}{2\pi}\left|\Delta\theta_t\right| & \exists k \ge 0 : \left(\left|\sum_{i=t-k}^{t} \Delta\theta_i\right| \ge 2\pi\right) \text{and} \left(\frac{\left|\Delta\theta_i\right|}{\Delta t_i} > 2\pi, \forall i = (t-k)..t\right) \\ \frac{80}{2\pi}\left|\Delta\theta_t\right| & \exists k \ge 0 : \left(\left|\sum_{i=t-k}^{t} \Delta\theta_i\right| \ge 2\pi\right) \text{and} \left(\frac{\left|\Delta\theta_i\right|}{\Delta t_i} > \pi, \forall i = (t-k)..t\right) \\ \frac{20}{2\pi}\left|\Delta\theta_t\right| & \text{otherwise} \end{cases}$$

In the equation above, $\Delta\theta_i$ represents the angular change by the finger's circular motion during time interval Δt_i. The current time step is t, and the number of scrolling items $f(\Delta\theta_t)$ only depends on the current angular change $\Delta\theta_t$. The branching conditions ensure that the specific speed level has been retained throughout a full circle, up to the current time interval.

3.3.3 Displacement

The displacement interaction technique is similar to a rate-controlled displacement joystick where the distance moved from a center position controls the speed of movement of the selector. This technique has been extensively studied in prior work [13,21].

Once a user touches the StampPad with his or her thumb, a dead-zone around this point is drawn (Figure 5a). The selector does not move until the user's finger exits the dead-zone. If the user exits the dead-zone at either the top or bottom sides of the square, then vertical movement commences and the system draws horizontal stripes to show the regions for the different speeds at which the selection can move (Figure 5b). The movement will stop once the user lifts their finger, and the technique is reset.

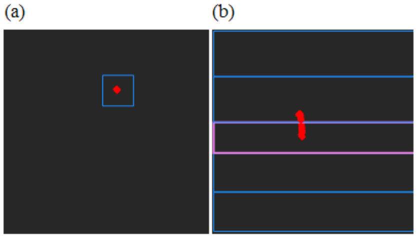


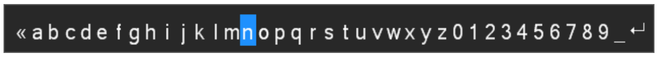
Figure 5. (a) The initial dead-zone region that is created when the user touches the StampPad. (b) The horizontal regions are drawn after the user has exited from the dead-zone.

Each horizontal strip has its own speed. The selector moves faster as the user crosses more regions in the same direction. The selector is moved one item per 10, 30, 120, or 150 ms, depending on the region in which the thumb lies. The direction of finger movement when first entering a region also determines the direction of selector movement. The dynamic assignment of speed and direction to regions supports the behavior of users that was observed during a pilot study. In the pilot study, each region had been fixedly assigned a speed and direction, but users constantly overshot their desired targets. The overshooting behavior has been previously documented with other rate-controlled joysticks [12]. The current technique solves this problem by dynamically assigning speed and direction based on thumb movement.

3.4 Text Entry

3.4.1 Linear Keyboard

The linear keyboard (Figure 6) is a common text entry method found in computer games and mobile devices. Our linear keyboard uses a horizontal layout of Roman letters, numbers, and ENTER, SPACE, and BACKSPACE. The interaction with the linear keyboard's selector is one-dimensional along the horizontal axis.



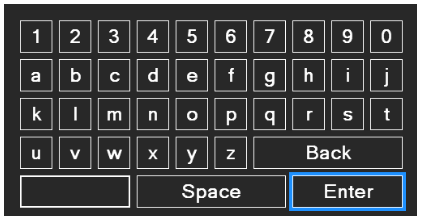
« a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9 _ ↵

Figure 6. The linear keyboard with the selector over "n".

There are many variations of the layout of the keys in the linear keyboard [11]. The one evaluated in this investigation is [BKSP][*a*…*z*][0…9][SPACE][ENTER]. Moving the selector to the right moves over the characters until the selector reaches ENTER ("↵"). Moving the selector to the left moves over the characters until the selector reaches BACKSPACE ("«"). In pilot tests of a linear keyboard that allowed the selection to wrap around the edges, subjects were confused, so we eliminated that behavior, choosing to "pin" the selector at either end. This also makes it very fast to get to ENTER to confirm one's entry, and backspace for corrections. In normal text entry, SPACE would be the most commonly entered character [21], but in destination name entry, spaces are rarely needed. We therefore put SPACE just to the left of ENTER, which allowed users to quickly reach SPACE and also to commit their selection after inputting a space by just moving one key to the right.

3.4.2 Two-Dimensional Keyboard

Two-dimensional selection keyboards (Figure 7) are common in video games, entertainment systems, multimedia players, and driving navigation systems. Our selection keyboard arranges the keys in a grid, allowing the user to move the selector (the halo) up, down, left, or right to reach the desired character. Common selection keyboard layouts are QWERTY and alphabetic. The alphabetic layout was chosen for this implementation since most navigation systems available today are based on alphabetic keyboards. For the sake of consistency with our linear keyboard, the selector does not wrap around the keyboard's edges.



1 2 3 4 5 6 7 8 9 0
a b c d e f g h i j
k l m n o p q r s t
u v w x y z Back
Space Enter

Figure 7. The 2-D selection keyboard with the selector over ENTER.

3.4.3 StampPad EdgeWrite

EdgeWrite is a stroke-based text entry method that recognizes characters based on the order in which the four corners of a square input area are hit (Figure 8a) [24]. In StampPad EdgeWrite (Figure 8b), letter segmentation is accomplished when the user's thumb is lifted from the pad.

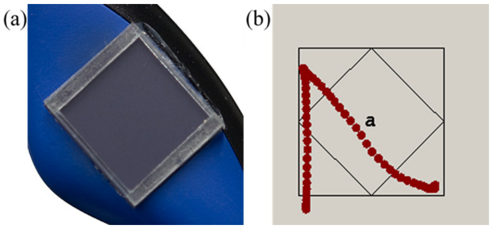


Figure 8. (a) StampPad with EdgeWrite plastic template. (b) The on-screen depiction of a trace of an EdgeWrite "a" on the StampPad.

The edges of the EdgeWrite plastic template aid in the thumb's movement between corners, providing stability. EdgeWrite's alphabet was designed to mimic Roman letter forms, resulting in an easily learnable alphabet. Studies of EdgeWrite have shown that it is 18% more accurate than Graffiti and just as learnable [22].

3.5 Implementation

Our destination list and all of these techniques were implemented in Visual C#. The interaction techniques communicated with Synaptics StampPad drivers to obtain finger-down, finger-up, and finger-move events. Note that for some techniques, the *absolute* position of the thumb on the touchpad had to be detected, not just its relative change in position. The interaction techniques converted the touchpad events into key events (letters or arrow keys) that were sent through the low-level keyboard input stream to the destination list. The keyboards and list techniques consumed the backspace, arrow, and ENTER events by moving the selector accordingly. For EdgeWrite, the character set that was used was limited to letters and numbers, since punctuation was not implemented in any of the other techniques.

4 EVALUATION OF DESTINATION ENTRY TECHNIQUES

After pilot testing all of the possible combinations of the interaction and keyboard techniques, we chose the seven that performed the best (see Figure 9). We conducted a two-session laboratory study to evaluate how these seven methods performed in (1) non-driving and (2) simulated driving situations. When not driving, participants used all seven methods while stationary. At the end of this session they also had some driving practice with the simulator *without* doing any destination entry. In a follow-up session, participants used all seven methods again, this time performing them while doing simulated driving.

	List (No Keyboard)	Linear Keyboard	2-Dimensional Keyboard
Clutching	✓	✓	✓
Dialing	✓	✓	×
Displacement	✓	†	†
EdgeWrite	✓	×	×

Figure 9. Combinations of visualization techniques and interaction techniques to create the seven "methods" under investigation. (✓ = tested, × = not possible, † = ruled out)

4.1 Method

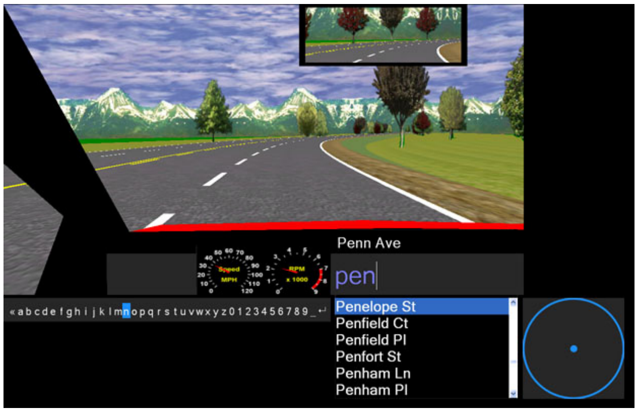
4.1.1 Participants

Four participants volunteered for this study. Importantly, these participants were already familiar with EdgeWrite from a previous study that tested EdgeWrite with an entirely different device (a cell phone with an isometric joystick [23]). This meant that they knew the EdgeWrite alphabet, allowing us to test their ability to physically *perform* the technique apart from learning the letter strokes. This issue is of less concern with the other methods, since direct list-selection and using on-screen keyboards was familiar to these participants from their daily use of electronics in general.

Participant ages ranged from 21 to 27, with an average age of 24. One participant was female. All participants were licensed drivers, and none of them had used or owned navigation systems. They were paid $15 for the first (non-driving) session. For the second (driving) session, they received $25 as the base payment, and for each method they received $3 bonus for driving safely—that is, without any crashes or centerline crossings. An extra $1 was given for driving safely with all methods.

4.1.2 Apparatus

Our Logitech Driving Force steering wheel was clamped on to the edge of a table in front of a 24-inch LCD computer monitor that served as the display for both the simulator and the interfaces for selecting street names (see Figure 1). The separate module consisting of the gas and brake pedals was positioned under the table. Performing the methods required the use of the steering wheel even for the non-driving session, so participants were seated in a chair in front of the steering wheel for both parts of the study. Two separate pieces of software were used in the study. One of them was the *StiSim Drive* simulator [18], which has been shown to correlate with real-world driving performance in a previous study by others [8]. The other is the user test software we wrote which handles the interaction with the StampPad and selecting of the street names for all of the destination entry techniques. *StiSim* generated simulated vehicle traffic, ambient sounds, scenery, and the inside of the car cabin. We placed the interface for each destination entry method near the speedometer (Figure 10) simulating a type of heads-up or high-dashboard display which are becoming increasingly common in cars [20].



Penn Ave

pen

« a b c d e f g h i j k l m n o p q r s t u v w x y z 0 1 2 3 4 5 6 7 8 9 _ ↵

Penelope St
Penfield Ct
Penfield Pl
Penfort St
Penham Ln
Penham Pl

Figure 10. A screenshot showing what the participants would see while driving and using the linear keyboard and the dialing method.

The interfaces of the seven destination entry methods varied in appearance, but they all were accompanied by the destination list shown in Figure 3. Displays placed to the side of the driver, such as where navigation systems usually are, would be expected to perform worse in all conditions, but perhaps not to an equal amount.

To make the simulation as realistic as possible, we enabled the simulator's sounds for the car, such as car engine noises, skidding squeaks, and crashes. The car engine noises are especially useful in signaling to the participants their current driving speed.

The top of the list showed the target street name that the participant should select (Figure 3). For techniques that entered one character at a time, such as EdgeWrite, the street name input box showed the characters that had been entered thus far. For methods that directly selected list items, e.g., using clutching to directly move the list selector, the input area showed only the street name in full after it was selected. The software did not show a new presented street name until a correct selection was made for the previous one.

The user test software also produced several sounds to signify events that happened during selection. A pleasant sound denoted a correct street name selection, while an error sound signified an incorrect selection. For methods that entered characters one at a time, if the participant had entered a sequence of characters that did not form the prefix of any of the available street names, a different error sound was heard and the color of the text changed to red.

4.1.3 Trials

A single trial consisted of entering one of the 3045 street names in the city of Pittsburgh, Pennsylvania. The street names contained only letters and spaces and users did not have to deal with capitalization. Each trial was timed. In a trial, the timer started after the target street name was shown and the participants thumb touched the StampPad. The timer stopped when the participant confirmed selection of the correct street name.

Fourteen mutually exclusive sets of 10 street names were carefully chosen from the 3045 street names. The non-driving session used sets 1-7. Similarly, the driving session used sets 8-14. The order in which the sets were presented was randomized for every participant. Two balancing criteria were used to create the destination sets. The first criterion balanced the sets for scrolling distance from the top of the destination list. We divided the list into 10 sections—the first section was closest to the top while the tenth section was the farthest. Each set contained one street name from each of the 10 sections. The second criterion balanced the number of keystrokes required to enter the street names, accounting for auto-completion. Each set contained two words requiring three keystrokes, three words requiring four keystrokes, three words requiring five key strokes, and two words requiring six keystrokes. In total, 45 keystrokes were required to optimally enter all 10 street names in a set.

4.1.4 Experiment Design

The experiment used a two-factor within-subjects design, with factors for *method* (1-7) and *driving state* (stationary, driving). For both driving states, the four participants performed all seven methods selecting 10 street names per method. Thus, there were 2×4×7×10 = 560 street names selected in all. Method order was counterbalanced to control for ordering effects. First, the order of the interaction techniques was counterbalanced with a Latin Square. Then, within interaction techniques, the various keyboard options were further counterbalanced. For example, all three clutching techniques were performed contiguously, but in different orders for each subject. The main dependent measure was the time to perform the selections. Other measures were specific to the driving condition, such as driving errors of various types.

4.1.5 The Stationary (Non-Driving) Session

In the stationary session, participants performed the seven methods using the steering wheel while seated but without the driving simulator. For each method, we first introduced the method to the participants and showed them how to perform it. They were told to be "fast and accurate" when selecting street names and began performing 15 street selections per method. (The first 5 were for practice; the other 10 were for testing.) For the 5 practice trials with EdgeWrite, participants were also given the EdgeWrite character chart, but they did not have the chart available to them when performing the 10 test trials. The other methods did not require any kind of charts or aides, since they were self-evident on the computer screen.

After finishing all seven methods, we asked participants to drive along a one-mile stretch of road to familiarize themselves with the *StiSim* simulator and the steering wheel. They did not perform any destination entries during this warm-up drive. The road was purposefully chosen to be challenging, having several sharp turns and oncoming traffic so that participants stayed alert during practice. The entire first session took about 1.5 hours.

4.1.6 The Simulated Driving Session

The simulated driving session took place either one or two days after the first session. For simulated driving, we used a test road that was 10 miles long with alternating gentle left and right turns every half mile (Figure 11). The road had two lanes in each direction, and was designed to be like a common countryside road. There were no other vehicles on the road. To make sure participants were comfortable using the simulator and could drive safely, they were asked to drive along the test road for 5 minutes while maintaining a speed of approximately 50 mph at the beginning of the session. The speed limit for the road was 55 mph, and the car could not go faster than 70 mph. All participants complied with the speed requirement and completed the practice drive without any safety violations.

After the practice drive, participants were informed about the four types of driving mistakes: (1) off-road accident, (2) centerline crossing, (3) road-side excursion, and (4) speeding. Participants were told that these were to be avoided during the upcoming trials, as driving safely was the most important goal in the study. To promote safety, participants were told that they would receive a $3 bonus for driving safely for each method. Then participants performed 15 street selections for each method while driving (5 practice, 10 test) in the same order as in the non-driving session. For each method, they started over at the beginning of the road.

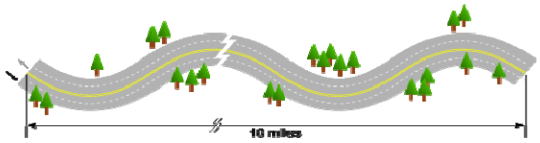


Figure 11. The test road used in the study. It had alternating gentle left and right turns every half mile.

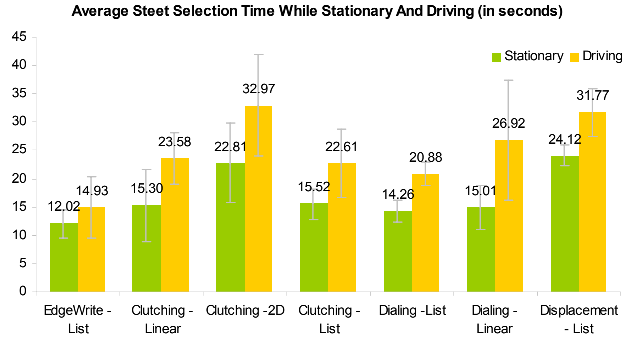
4.2 Quantitative Results

4.2.1 Data Analysis

Street selection times and driving performance were logged by our test software and the driving simulator. Street selection data were analyzed using a mixed model analysis of variance with fixed effects for *method* and *driving state* and a random effect for *participant*. This analysis method is more appropriate for our within-subjects study than a traditional ANOVA because individual error terms are synthesized for each participant [10]. Modeling *participant* as a random effect is necessary for this analysis because the participants represent a population sample. The model also takes into account the fact that measurements within participants are not independent. (For more information on mixed model analyses of variance, see Chapter 3 in [10].) We tested for the effect of *method order* on street selection time but found none, indicating adequate counterbalancing ($F_{6,46}$=0.85, n.s.).

4.2.2 Street Selection Results

Not surprisingly, participants performed all street selection methods more slowly while driving than while stationary. While stationary, EdgeWrite was the fastest, with an average selection time of 12.02s (σ=2.56). Displacement with list, which was about twice as slow as EdgeWrite, was the slowest overall, with an average selection time of 24.12s (1.90). While driving, EdgeWrite was also the fastest, with a selection time of 14.93s (5.49). Clutching with 2-D keyboard was the slowest, with selection time of 32.97s (8.98). Figure 12 shows the average time for each method while stationary and while driving.



$F_{1,39}$	0.72	**5.27**	**7.77**	*3.14*	**13.04**	*3.48*	**4.99**
p	0.4026	**0.0272**	**0.0082**	*0.0841*	**0.0009**	*0.0692*	**0.0313**

Figure 12. The average street selection time in seconds for each method while stationary and while driving. The bottom table shows the F- and p-values for contrast tests between stationary and driving conditions within each method (**bold** are significant, *italic* are marginally significant).

The main effect of *driving state* (stationary, driving) on street selection time was significant ($F_{1,39}$=33.91, p<.0001). There was also a significant effect of *method* on selection time ($F_{6,39}$=3.39, p=.0086), confirming that participants attained different speeds selecting with each method. However, there was no significant *driving state* × *method* interaction ($F_{6,39}$=0.75, p=.62), which means that methods relatively performed about the same from not driving to driving.

4.2.3 Method Contrasts

Having looked at the main effects, we now turn to contrasts for finer comparisons between individual methods. First, we consider how all clutching methods (list, linear keyboard, and 2-D keyboard) as a whole compare to all dialing methods (list, linear keyboard). This was found to be non-significant ($F_{1,39}$=2.04, p=.16). Second, we consider how the three clutching methods compare among themselves, finding that clutching with linear keyboard and direct-selection from the street list have indistinguishable performance ($F_{1,39}$=0.0067, p=.95). Clutching with a 2-D keyboard, on the other hand, was worse—significantly slower than both clutching with linear keyboard ($F_{1,39}$=4.56, p=.04) and clutching with list ($F_{1,39}$=4.31, p=.04). Thus, the 2-D selection keyboard did not work well with clutching. Furthermore, the two dialing methods (with list, with linear keyboard) performed indistinguishably ($F_{1,39}$=0.05, p=.83).

EdgeWrite was the fastest method on average (13.47s per street selection). However, it was not significantly faster than dialing with list, the second fastest method ($F_{1,39}$=0.41, p=.53). When compared to clutching directly on the street name list, EdgeWrite was not significantly faster ($F_{1,39}$=0.87, p=.36). However, EdgeWrite was significantly faster than the two slowest methods, clutching with 2-D keyboard ($F_{1,39}$=9.43, p=.0039) and displacement with list ($F_{1,39}$=11.85, p=.0014). EdgeWrite, in short, was the best on average, but not significantly better across both driving states than any but the two slowest methods.

4.2.4 Driving State × Method Contrasts

Clearly, all methods became slower from stationary to driving. To find out whether such decrease in performance was significant, we looked at the decrease in average street selection time from stationery to driving for each method. The table in Figure 12 summarizes these results for all the seven methods. It shows that all methods except EdgeWrite became slower when going from stationary to driving. In fact, four methods were significantly slower and two methods were marginally significantly slower. EdgeWrite seemed to be the only method that could sustain its performance when going from stationary to driving. This might be due to the fact that gestural methods can be done somewhat by feel, but selection-based methods require a visual focus of attention.

4.2.5 Analysis of Driving Measures

For the second session, we collected driving data in the form of number of accidents, speeding violations, centerline crossings, roadside excursions, and speed in miles per hour (mph). With the exception of speed, these data are highly skewed towards zero, since participants were motivated to avoid these violations. Therefore, performing ANOVAs on these data is not feasible. Moreover, the data are not amenable to a meaningful transformation to make performing ANOVA worthwhile because their counts are so low. We therefore conducted non-parametric Wilcoxon signed-rank tests on response pairs, but found that with only four participants, we lacked sufficient power to detect significant differences. Therefore, our discussion of the driving data examines means, distributions, and rank orderings of the methods. We first look at driving mistakes.

4.2.6 Driving Mistakes

Figure 13 shows averages for each of the four types of driving mistakes. Clutching with the 2-D keyboard had numerous driving mistakes and it was the only method that had all four types of mistakes, ranking worst in all of the categories. Note that off-road accidents happened only when the car completely left the road. Clutching with the 2-D keyboard and dialing with linear keyboard were the only two methods with such egregious mistakes.

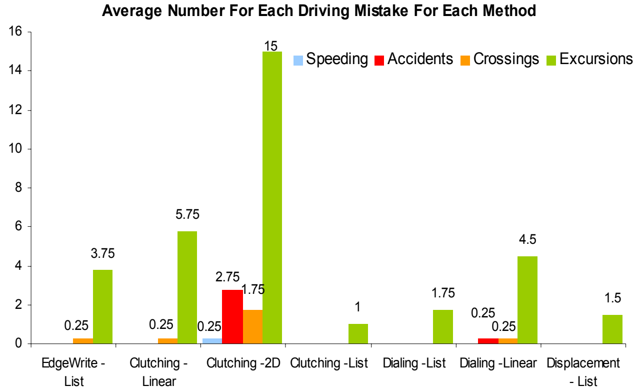


Figure 13. The average number of driving mistake for each method.

There were only a few speeding violations and centerline crossings for most methods, except for clutching with the 2-D keyboard. Road edge excursions, which were quite easy to commit, were the most common mistakes for all methods.

4.2.7 Driving Speed

The grand mean driving speed of all methods was 32.9 mph (σ=2.11). Figure 14 shows the average driving speeds for each method. Displacement with list had the fastest driving speed, and dialing with list had the slowest. The driving speed differences were found to be marginally significant with α=0.1 ($F_{6,8}$=2.12, p=.10). When we look closely at the ranking of driving speeds and compare it to the ranking of selection speeds, we find that the two orderings are exactly opposite—that is, the method with the fastest driving speed had the slowest selection speed, and vice versa. The exception to this is EdgeWrite, which had the third fastest driving speed and the fastest selection speed. Figure 15 depicts this pattern.

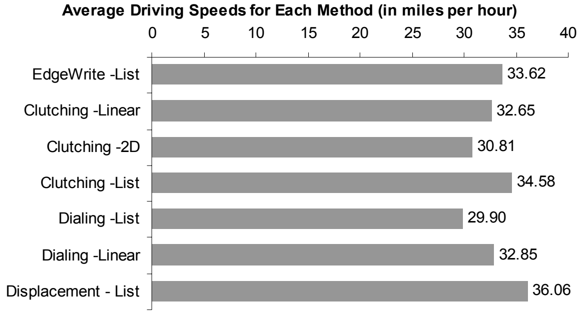


Figure 14. The average driving speed with each method (mph).

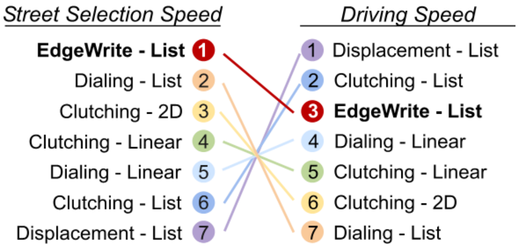


Figure 15. Generally, the method with fastest driving speed had the slowest selection speed, and vice versa. An exception was EdgeWrite.

4.3 Qualitative Results

At the end of the study, participants were asked to rank each of the seven methods with respect to ease of use, accuracy, speed, etc. On the whole, participants ranked the four non-keyboard methods more favorably than the three keyboard methods. This is interesting in light of manufacturers' general reliance on on-screen keyboards. However, one participant did rank dialing with the linear keyboard favorably. Among the four non-keyboard methods, participants seemed to have slight preference toward EdgeWrite and dialing directly with the street name list.

5 Discussion

On average, the fastest method for selection in the stationary condition was EdgeWrite, at 12.02s; the second fastest was dialing directly with the list, at 14.26s. In the driving condition, EdgeWrite was also the fastest, at 14.93s. The 2-D keyboard with clutching was on average the slowest in both the stationary and driving conditions. This is striking considering that most navigation systems currently use 2-D keyboards for destination entry. One possible explanation for this is the fact that our steering wheel prototype was not optimized for thumb movement alternating along both axes, and the 2-D keyboard is the only method that requires thumb movement along the top-left to bottom-right diagonal of the input region. The thumb's dexterity and rage of motion along the top-right to bottom-left diagonal is much better than along the other diagonal for right handed users. Since the touchpad is at a 45° angle on the steering wheel, it is possible that users in this study were able to move along the horizontal axis with less difficulty than along the vertical axis and this disparity possibly resulted in degraded overall performance.

6 Future Work

The primary issue with this study is that we still did not find a technique that had sufficiently small times or level of distraction to be recommended while driving. Furthermore, our evaluations would be better validated with test-track driving. However, we wanted to start with a simulator for our initial investigation of these techniques.

The scope of this study is limited to selection from a long list. Most navigation systems today require a number of steps before users are able to select from a list of items or enter their desired selection. A study that requires the user to interact with the system from an initial "power-on" screen could potentially yield different results. In some sense, this study investigated user input in only the most fundamental aspect of navigation system entry.

In future studies, we want to use eye-tracking technology to measure glances away from the road as an additional dependent measure. We would also like to test this system with non-expert EdgeWrite users and understand the learning curve of StampPad EdgeWrite while driving.

7 Conclusions

We have presented a system for selecting and performing text entry in an automotive setting that does not require users to take their hands off the wheel. We evaluated seven methods of street name entry using a hardware prototype and software techniques. Our study showed that using the EdgeWrite text input method on a steering wheel might be a promising means of input to navigation systems and other devices in automobiles.

This work is a first attempt to explore and evaluate possible emerging alternatives and ultimately technologies that are adopted in automotive settings require a very high level of maturity.

8 Acknowledgements

This work was supported in part by General Motors, Microsoft, Synaptics, a Philip D. Reed Undergraduate Fellowship, and the National Science Foundation under grant UA-0308065. Any opinions, findings, conclusions, or recommendations expressed in this material are those of the authors and do not necessarily reflect those of the National Science Foundation or any other supporter.

References

[1] Arroyo, E., Sullivan, S., and Selker, T. (2006) CarCoach: a polite and effective driving coach. Proc. CHI '06. ACM Press, 357-362.

[2] Beck, D. and Elkerton, J. (1989) Development and evaluation of direct manipulation lists (poster session). CHI Bulletin '89, 72-78.

[3] Dingus, T. A., Antin, J. F., Hulse, M. C., & Wierwille, W. W. (1989). Attentional demand requirements of an automobile moving-map navigation system. Transportation Research, 23A (4), 301-315.

[4] Hafner, K. (2002) Comforts of Home Yield to Tyranny of Digital Gizmos. New York Times, April 28, 2002.

[5] Hirotaka, N. (2003) Reassessing current cell phone designs: Using thumb input effectively. Ext Abs. CHI '03. ACM Press, 938-939.

[6] Holbrook, C. (2003) Input Methods for Notification Systems: A design analysis technique with a focus on input for dual-task situations. MS Thesis, Virginia Polytechnic Institute and State University.

[7] Lee J., Hoffman J., and Hayes E. (2004) Collision warning design to mitigate driver distraction. Proc. CHI '04. ACM Press, 65-72.

[8] Lee, H.C., Cameron, D. and Lee, A.H. (2003) Assessing the driving performance of older adult drivers: on-road versus simulated driving. Accident Analysis and Prevention 35 (5), 797–803.

[9] Lehikoinen, J. and Salminen, I. (2002) An Empirical and Theoretical Evaluation of BinScroll: A Rapid Selection Technique for Alphanumeric Lists. Personal Ubiquitous Comput. 6, 2 (Jan. 2002), 141-150.

[10] Littell, R. C., Milliken, G. A., Stroup, W. W. and Wolfinger, R. D. (1996) SAS System for Mixed Models. Cary, North Carolina: SAS Institute, Inc.

[11] MacKenzie, I.S. (2002) Mobile text entry using three keys. Proc. NordiCHI '02 , vol. 31. ACM Press, 27-34.

[12] Mithal, A.K. and Douglas, S.A. (1996) Differences in movement microstructure of the mouse and the finger-controlled isometric joystick. Proc. CHI '96. ACM Press, 300-307.

[13] Myers, B.A., Lie, K, and Yang, B. (2000) Two-handed input using a PDA and a mouse. Proc. CHI '00. ACM Press, 41–48.

[14] New Mexico driver's guidelines. http://www.state.nm.us/tax/pubs/dlm/dlm.htm

[15] Privilege Insurance. Unsafe use of navigation equipment. http://www.privilege.com/aboutus/Unsafeusenavigation.htm

[16] Proschowsky, M., Schultz, N., and Jacobsen, N. E. (2006) An intuitive text input method for touch wheels. Proc. CHI '06. ACM Press, 467-470.

[17] Schneider, M, and Kiesler, S. (2005) Calling While Driving: Effects of a Remote Traffic Display. Proc CHI '05. ACM Press, 561–569.

[18] Systems Technology Inc,- STISIM Drive. http://www.systemstech.com/content/view/23/39/

[19] Tijerina, L., Parmer, E., and Goodman, M.J. (1998) Driver Workload Assessment of Route Guidance System Destination Entry While Driving: A Test Track Study, Proc. of the 5th ITS World Congress.

[20] Wikipedia. Head-ups display. http://en.wikipedia.org/wiki/Heads_up_display

[21] Wilson, A.D. and Agrawala, M. (2006) Text entry using a dual joystick game controller. Proc. CHI '06. ACM Press, 475-478.

[22] Wobbrock, J.O., Aung, H.H., Rothrock, B. and Myers, B.A. (2005) Maximizing the guessability of symbolic input. Ext. Abs. CHI '05. ACM Press, 1869-1872.

[23] Wobbrock, J.O., Chau, D.H., and Myers, B.A. (2007) An Alternative to Push, Press, and Tap-tap-tap: Gesturing on an Isometric Joystick for Mobile Phone Text Entry. Proc. CHI'07. ACM Press. To appear.

[24] Wobbrock, J.O., Myers, B.A., Aung, H.H. and LoPresti, E.F. (2004) Text entry from power wheelchairs: EdgeWrite for joysticks and touchpads. Proc. ASSETS'04. ACM Press, 110-117.