Using IDs to Improve Web Navigation with a Keyboard

Hooman Baradaran & I. Scott MacKenzie
Department of Computer Science and Engineering
York University
Toronto, Ontario, Canada, M3J 1P3
{hooman,mack}@cse.yorku.ca

Abstract

We evaluated two variations of a keyboard-only web navigation technique. "ID Navigation" assigns a unique, incremental identifier (ID) to each interactive web page element. The ID is displayed as a tiny label beside each element when an activation key is pressed. One variation used numeric IDs, the other used a set of IDs limited to four sequential keys each associated with a fixed finger. The time to select elements using a mouse, TAB-key navigation, and the two new techniques was analysed using a keystroke-level model and measured in an experiment. Selection time increased linearly with the number of elements for TAB-key navigation and was constant for ID navigation and the mouse. ID navigation was significantly faster than TAB-key navigation however the mouse was fastest overall, as expected. Numeric ID navigation was the faster of the new methods and was better liked by the users.

1. Introduction

Web browsers are a classic example of "direct manipulation interfaces". Shneiderman [14] coined this term in the 1980s to characterize the emerging style in computing, as character-based key-entry gave way to graphical point-and-click interfaces. The "directness" in these systems comes by way of the mouse, where users manoeuvre an on-screen tracker (via the mouse) to an element of their choice and then select the element by clicking a mouse button. Simple as this is, there are a number of deficiencies. One is the constant need to "home" the hand between the keyboard and the mouse. Another is the valuable desk space ("real estate") the mouse occupies beside the keyboard. Notably as well, some users have a preference (or need), and some environments mandate, a keyboard for interaction. And, so, keyboard-only interaction is of interest even today.

We studied a method of keyboard-only web navigation where unique, incremental identifiers (IDs) are assigned to interactive elements on the page. The IDs appear when a modifier key ("activation key") is pushed. This is followed by selection with a few keystrokes. Similar methods are available as *Firefox* extensions (e.g., *Mouseless Browsing* [11]), however no empirical evaluations exist. Based on pilot tests on a modified version of this extension, we improved the interaction technique and undertook what to our knowledge is the first empirical test of such a technique. The method is defined and compared to traditional TAB-key and mouse navigation using both a predictive model and an empirical study.

Web interfaces were designed as point-and-click graphical user interface systems; and, so, HTML elements [12] are naturally manipulated by pointing devices. This works well when simply browsing through web pages on a desktop computer with a mouse. However, point-and-click

interaction does not work well in all situations. Today, a growing number of applications use web-based interfaces and when using web interfaces for data entry, continuous switching between the keyboard and mouse is required.

Individuals with neuromuscular impairments often cannot use fine movements with a mouse (or use a mouse at all) due to spinal cord injuries, brain damage, and such. This is one of the reasons web accessibility guidelines [1,6] require the possibility of web browsing using a keyboard in lieu of a pointing device.

Additionally, experienced users and touch-typists may simply prefer to use the keyboard to reduce the time lost due to "homing". So, an efficient keyboard navigation method is particularly important for expert users. While keyboard navigation is possible in current web browsers, it is generally slow and inefficient [13].

2. Background and Related Work

Navigating web interfaces using the keyboard in most browsers requires a huge number of TABs, also known as TAB-chains. To reach a desired element, users press TAB to advance through all other selectable elements between the element currently focused and the target element. There are two types of "selectable elements": widgets (buttons, text-input elements, drop-down lists, check-boxes and radio buttons, etc.) and hyperlinks.

Since the natural flow of TAB-chains is one directional, navigating using the TAB key is laborious. In the worst case – advancing to the previous element – focus advances through every element in the page and all focusable widgets in the browser. While most browsers allow reverse navigation using a modifier key, such as SHIFT, this is not obvious to the user. The reverse TAB-chain may be large too and holding an extra key makes the task even more cumbersome. This can even cause repetitive strain injury [3].

Another issue with TAB navigation is the order of elements. By default, TAB advances over elements according to their order in the HTML file. This is often different from their order on the page. Defining a custom TAB order may help; however, this is typically either not done or not done correctly. For example, modifications to a page may result in incorrect TAB orders which is far worse than the longer but correctly ordered default TAB order.

Developers of some web-based applications have tried to improve keyboard navigation using client-side scripting to capture keyboard shortcuts. This works well for web applications used regularly (e.g., email) but does not eliminate the need for a more generic improvement for keyboard navigation.

KeySurf [15] is a similar technique that uses labels to facilitate element selection for disabled users. It predicts elements the user is most likely to select next. While this is beneficial for sites visited regularly, a simple incremental approach to creating labels is more predictable in general. Another limitation of KeySurf is that it relies on all letters of the alphabet rather than a limited set of characters or keys.

3. ID Navigation

The idea behind keyboard web navigation, henceforth "ID navigation", is to assign a unique, incremental ID to each interactive element. ID navigation extends the idea of mnemonics used in desktop applications to an arbitrary number of elements by using incremental IDs instead of prefixes. The ID is displayed as a tiny label next to each element, thus only a single visual scan is required to find the element itself. To minimize screen real estate and avoid clutter, the ID pops up only when a modifier key is pressed.

Initially, we assume keyboard focus is on the actual web page, not on the browser, and that the keyboard is either unused or not used for text entry. For example, step keys may scroll the page or change a value in a drop down list, or an action key may toggle a check box; but there is no numeric or text input. Thus, numeric key sequences are available to navigate between widgets based on assigned IDs.

Assigning IDs as incremental digits (instead of using an element's name or probability of selection) allows users to scan the labels in the

same top-to-bottom, left-to-right direction as text on a page. For a typical web site, elements at the top of the page are used for navigation and are generally repeated on all pages throughout the web site. Since these are the first elements to have IDs assigned to them, they get short, easier-to-remember IDs and are unchanged across all pages. This makes it easier for users to memorize them for future visits. Where the elements do not appear in the same order as in the HTML file, IDs are still assigned to related elements (such as list items) in sequence.

ID navigation is meant to improve the existing keyboard navigation technique, not replace it. It is to facilitate faster movement between elements that are too far apart for efficient TAB access. For example, the user can use TAB navigation when filling out a web form's inputs one by one and use ID navigation to "jump" to a different part of the page.

3.1 Numeric ID Navigation

With the original *Firefox* extension [11], typing a numeric ID followed by a short timeout or an action key switched focus to the target element. This method required 2-3 keystrokes for selection: 1 or 2 numeric key presses for the ID and 1 key press (or a timeout) for selection. Clearly, this was less burdensome than traditional TAB-chain navigation.

Natural numbers are especially useful when a numeric keypad is accessible to the user. Additionally, this allows the technique to easily extended to speech recognition systems. With a limited vocabulary of only 10 digits, the chance of error during speech recognition is low. Even when the numbers are not read one digit at a time, it is easy for a simple speech recognition system to detect the spoken numbers.

The main problem with the original extension was that the IDs were either shown continuously, which caused clutter, or were "toggled" using a key. If toggled, users had to hit a toggle key to bring up the IDs, and then perform the selection. This is slow and increases the visual and cognitive burden.

3.2 Ambiguity with Text-input Controls

Text-input[1] elements pose a challenge for ID navigation. When focus is on a text-input element, a method is required to distinguish between the control ID and the data entered into the element.

One solution is to use the numeric keypad exclusively for navigation and the main number keys for number entry. This is a problem for users who prefer the numeric keypad for entering numbers, and for those who use a laptop. Numeric keypads are awkward to use on notebook computers because they are integrated in the main keyboard and are accessible only through a mode shift. Additionally, this method only works when IDs are numeric.

An alternative is to use a modifier key in text-input elements to process the numeric keys for navigation. In the original extension, the CONTROL key was assigned to toggle numeric entry. We modified the technique to show the IDs only while an activation key is held, allowing users to hold the modifier while entering an ID regardless of the past state or the keyboard focus. This results in a more consistent interaction.

3.3 Beyond Numeric IDs

While numbers are incremental and easy to follow, the IDs can use any character in the keyboard. This is particularly useful if a numeric keypad is not available, as for laptops.

Using ID Navigation with a limited number of keys can greatly benefit disabled users who use special input devices such as chord keyboards [7].

We developed a 4-finger variation of ID navigation where four fingers are used on four adjacent keys. With enough practice this method has great potential for touch typists. We picked J, K, L and semicolon (;) as these keys form the normal home position for the right hand when touch typing. Also, the J key has tactile feedback, making it easier to put the fingers back in

[1] For keyboard navigation, drop-down lists are like text-input elements because key sequences are used to scan through available values. This is useful for keyboard navigation, especially for larger drop-down lists.

position. A small pilot study showed using the symbols J, K, L and ";" in labels was confusing, so we used numbers 1-4 on the labels instead. See Figure 1.

Figure 1: ID Navigation with four fingers

A similar group of keys would be A, S, D and F using the left hand. This allows full navigation with the left hand alone. However, not all users may comfortably hold the activation key (e.g., ALT) with the left thumb while typing the labels with their other four fingers.

3.4 Predicting Using a Keystroke-Level Model

The selection time to move from one widget to another for the techniques discussed above can be predicted using Card et al.'s Keystroke-Level Model (KLM) [4]. In its general form, the model gives

$$T_{Execute} = nT_K + nT_P + nT_H + nT_D + nT_M + nT_R \quad (1)$$

where the total execution time is predicted by summing primitive operations. T_K is the time for a keystroke or button press and is calculated for any key including mouse buttons and modifier keys. T_P is the time to point to a target with the mouse. T_H is the homing time for the hand to switch between the keyboard and mouse. T_D is the drawing time (unused in this paper). T_M is the mental preparation time. T_R is the system response time. n is the number of repetitions of each operator.

To simplify the calculations and consider the web interface alone, we assume a page fits in a single view port and no scrolling is needed. Given this, the number of page elements is the same as the number of visible elements.

Predictions are developed below for a single selection of a focusable element on a web page using three different navigation methods.

3.4.1 Using the mouse

$$T_{Execute} = T_H + T_M + T_P + T_K \quad (2)$$

where

- T_H is the homing time. It equals 0.4 s [4] for switching from text-input widgets and 0 for other widgets where there is no need to use the keyboard.
- T_M is the mental preparation time to execute pointing. We set T_M for all devices to 1.35 s, as given by Card et al. [4].
- T_P is the pointing time to select the new widget with the mouse. It ranges from 0.8 to 1.5 s [4].
- T_K is the time to press the mouse button. Based on Card et al. [4], T_K ranges from 0.08 to 1.20 s; however, since this range was estimated from participants' typing speed rather than a mouse click, we chose the lower end of 0.08 s.

Given this, the estimated execution time to switch between widgets using the mouse is between 2.23 and 3.33 s.

3.4.2 Using the keyboard: Tab chains

$$T_{Execute} = T_M + n_{TAB}\, T_K + T_M + T_K \quad (3)$$

where

- T_M is the mental preparation time, as explained above. Based on heuristics for placing M operations [4], T_M is calculated once before the first TAB and once before selecting the desired element.
- n_{TAB} is the number of TABs to reach the target element. It is between 1 and n where n is the number of TAB-sensitive interactive elements. For a page with n elements, the average is:

$$n_{TAB} = (n + 1) / 2 \quad (4)$$

- The first T_K in eq. (3) is the time to press TAB.

It ranges from 0.08 to 1.20 s. The second T_K is the keystroke (typically ENTER) to select an element once focused. It is included for elements that need selection, like buttons and hyperlinks. Arguably, users who prefer keyboard navigation have above-average typing speeds, and since all strokes are on the same key, a lower T_K than given by Card et al. [4] may be warranted.

Given the above model components, the estimated time to switch between elements using TAB-key navigation is from $2.78 + 0.08n_{TAB}$ to $3.9 + 1.2n_{TAB}$ s, where n_{TAB} can be estimated by the average number of elements on the page using equation 4.

The model assumes elements are traversed in the normal order and does not consider reverse traversal using SHIFT.

3.4.3 Using the keyboard: ID Navigation

$$T_{Execute} = T_M + n_K T_K + T_R \quad (5)$$

where

- T_M is the mental preparation time to enter the ID of the element, as described above.
- n_K is the number of keystrokes to enter the ID and execute the action. Assuming a 2-digit ID, two keystrokes are needed to enter the ID and if enabled, a third key (e.g., ENTER) to confirm selection. Plus, there is an additional keystroke for the activation key.
- T_K is the keystroke time, as explained for TAB-chain navigation.
- T_R is the response time. If a timeout is used instead of a key press, $T_R = 0.5$ s, otherwise zero is used. In practice, if the timeout is used, it should be user configurable.

The estimated time to switch between elements using ID navigation, using an activation key, is between 1.59 to 4.95 s. The timeout option is only explained for the purpose of comparison with the original method.

3.4.4 Comparing the estimated execution times

Table 1 summarizes the model predictions for each navigation method. While model constants from Card et al. [4] may be inaccurate, the importance of the predictions lies in the relative comparisons they afford. The selection time for TAB navigation is a linear function of the number of TAB-sensitive elements. For both mouse and ID navigation, $T_{Execute}$ is constant regardless of the number of elements.

Table 1
Summary of Model Predictions (s) by Method

Method	Model Prediction (s)	
	Min	Max
Mouse	2.23	3.33
Tab		
5 elements	3.18	9.90
30 elements	5.18	39.90
60 elements	7.58	75.90
ID Navigation		
key selection	1.59	4.95
0.5 s timeout	2.09	5.45

Schrepp and Fischer's study [13] of the parent GOMS [5] module for the mouse vs. keyboard web navigation also reports linear results for the keyboard (using TAB) and constant results for the mouse. Furthermore, web-authoring statistics by WebWatch [8] and Google [2] show an average of 60 elements per web page and at least 20 elements for over 97% of web pages. Thus, the predictions for TAB-chain navigation in Table 1 may be on the low side. While the results for the mouse and ID navigation techniques are similar, an empirical comparison of the techniques is needed to test the predictions.

4. Method

An experiment was designed to compare two ID navigation methods (numeric and 4-finger) with the existing method (TAB) as well as the mouse as a baseline condition. To encompass the range of interactions usually found on web pages, two tasks were used. The first task was to measure the time to select a random target button from a set of buttons on a web page. The second task required the participants to enter text into form elements. As with our pilot studies, both tasks were limited

to a fixed view port. The experiment also included random web surfing to practice ID navigation techniques as well as a questionnaire in the end.

4.1 Participants

Sixteen participants (10 male, 6 female) performed the experiment. They were in their 20s (except one who was in early thirties) and all were relatively experienced computer users capable of using a mouse and a keyboard comfortably. Two participants were left-handed but used the mouse with their right hand. Thirteen participants indicated they were laptop users. Of these, 3 participants indicated they used an external mouse instead of the built-in pointing device. No participant had previous experience with any technique similar to ID navigation. One participant had used the "Find As You Type" [9] feature in *Firefox* which selects hyperlinks by typing their text on the keyboard.

4.2 Apparatus

The experiment was performed on a Linux-based machine in our local computer lab. Mozilla *Firefox* was used because it provides good keyboard support to access browser features, move between pages, etc. It is also easily modified and expanded using extensions.

We made a new extension for this experiment with significant improvements based on our pilot studies and user feedback. To improve the response time and minimize clutter, we presented the IDs in a different "layer" on top of the original web page. The computer in the experiment was fast enough to show the IDs without delay. The IDs popped up while an activation key (ALT) was held.

Previously, the selection was done by either hitting the ENTER key or after a delay of 500 ms after the last ID character was entered. Both methods caused high error rates in pilot studies. In this experiment, selection was done as soon as the activation key was released. Elements such as hyperlinks or buttons were clicked on selection.

To avoid mistakes, we introduced a delete key to clear the current selection before releasing the activation key. Users were able to hit delete, then type a new ID without releasing ALT. Participants were given the option of changing the default activation (ALT) or delete key (Z). Except one participant who chose the back-quote (`) key to delete, no participant changed the defaults.

We improved highlighting by using a box with a transparent yellow background and a red border instead of the default grey dashed border used in *Firefox*. This highlighting was only shown *during* selection to differentiate between an active selection and a previously selected element. As soon as the selection was confirmed (by releasing the activation key), the extra highlighting disappeared and the selected element was focused with the default dashed border.

The four input techniques were tested on two tasks: target selection ("Task 1") and data entry ("Task 2"). This was done using interactive web pages written in Javascript and stored on the local computer to avoid communication delays.

The web page used in Task 1 (target selection) consisted of a START button, a message box for instructions, and 25 target buttons. To minimize the visual scan time for all input techniques, the buttons were labeled with 25 words corresponding to 25 letters of the alphabet (A to Y) in alphabetical order. Figure 2a shows the experiment web page when the activation key is held down and the numeric IDs are shown.

Initially, only the START button was enabled on the page. Once participants clicked the START button, it was disabled and the target buttons were enabled with the button participants were expected to press shown in red and bold. Once a target button was selected, the message box indicated if the correct button was pressed. The participants were then instructed to start the next trial when ready.

The target button, the button actually pressed and the selection time were logged for each trial. For all keyboard techniques, the selection time started from the first key event after the START button was pushed. When the mouse was used, selection time started when the pointer moved beyond a certain threshold (in any direction) after the START button was clicked.

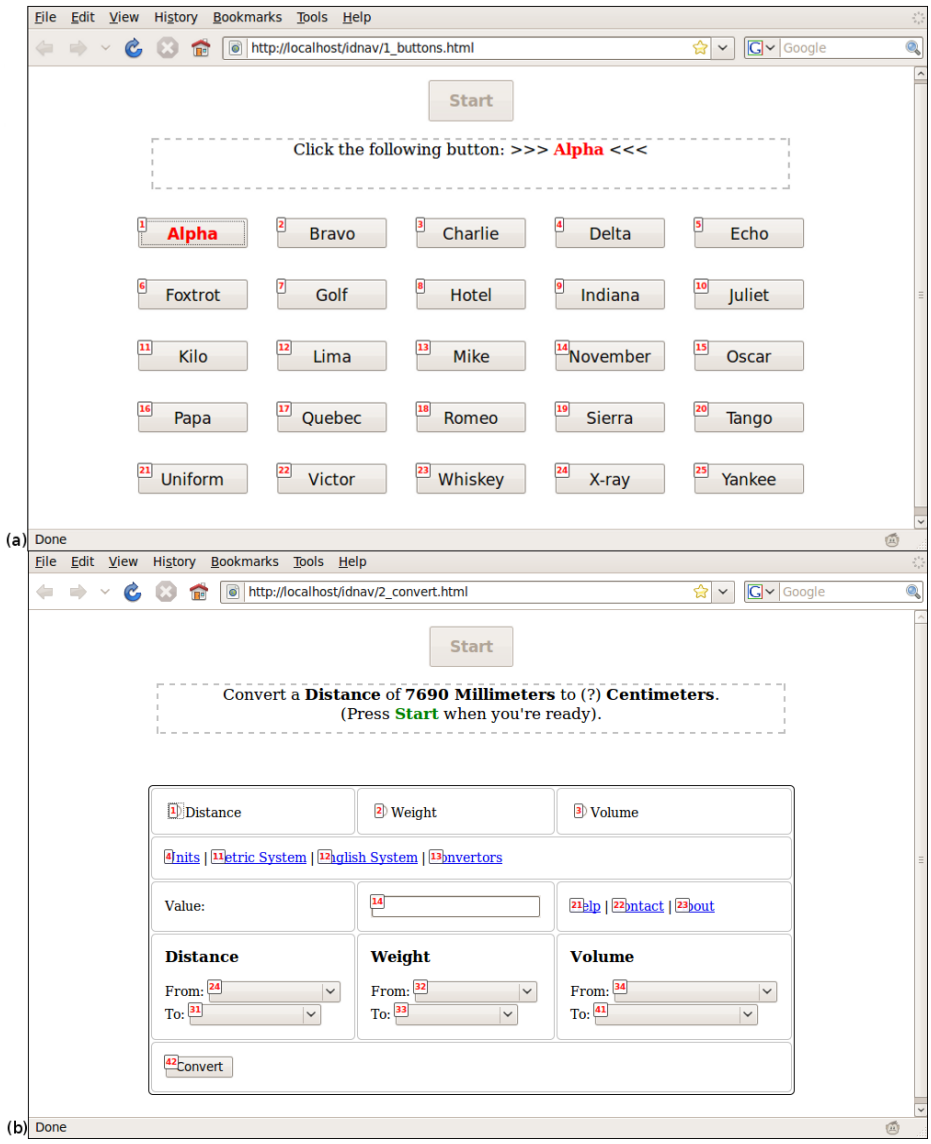

Figure 2. (a) Experimental interface during Task 1 (target selection). The activation key is down causing the numeric IDs to appear beside each focusable web page element. (b) The interface during Task 2 (data entry). The activation key is down and 4-finger ID Navigation is enabled.

In Task 2 (data entry), participants were presented with a form to convert a value from one unit to another. They had to select the type of conversion (distance, weight or volume), enter the value, select units from a corresponding drop-down menu and press a convert button. We also added a number of hyperlinks in two parts of the form to require participants to "jump" to a different section. Figure 2b shows the web page when 4-finger ID navigation is used with the IDs shown (i.e., the activation key is down).

4.3 Procedure

The experiment was performed in two one-hour sessions per participant. To avoid confusion due to the similarity of the ID navigation techniques, each session included one ID navigation technique (numeric or 4-finger) and either the mouse or TAB technique. The sessions were performed with a break of at least two hours but

no more than two days. We believe this was long enough to avoid fatigue and confusion between ID navigation techniques, yet short enough for participants to remember the first session when filling out our questionnaire.

At the beginning of each session the appropriate ID navigation technique was explained to participants and they were asked to practice the technique on random web sites. They were asked to browse different sites and try the technique on various pages and page elements including different types of form elements. A couple of participants who weren't familiar with TAB navigation were asked to practice that method as well. Once participants felt comfortable with the techniques, they performed a few test trials of the experiment using both techniques for the current session. Both tasks were completed for one technique, then after a break of a few minutes the other technique was tested.

Participants were instructed to press the START button and proceed as quickly and accurately as possible. They were told that at the end of each trial, they could stop and relax before pressing START again. If an error was made, such as an incorrect button selection or a mistake in the conversion input, it was indicated in the message box but the participants continued without redoing the trial. At the end of each block of trials, participants were asked to take a short break and continue when ready. To force breaks and avoid fatigue, the START button remained disabled for five seconds after each trial block ended.

To keep TABS within the web page, we initially focused the START button. After START was pushed and disabled, the first target button (or form element) was focused. As this is slightly different from normal TAB behaviour of the browser (where the address bar is focused), it was explained to the participants so they would not have to visually scan for the focus indicator.

4.4 Design

Since the interactions are substantially different between Task 1 and Task 2, they were treated as two separate experiments.

4.4.1. Task 1: Target selection

Task 1 was a 4 × 6 × 12 within-subjects design. The factors and levels were as follows:

Input Technique	ID navigation (numeric), ID navigation (4-finger), TAB, Mouse
Block	1, 2, 3, 4, 5, 6
Trial	12 buttons (all odd/even buttons in alternating blocks) in random order

The dependent variables for Task 1 were time to select a target button (selection time) and the mean error rate (%). The total number of trials was 4608 (16 participants × 4 input techniques × 6 blocks × 12 trials).

To minimize skill transfer, the input techniques were assigned to participants using a Latin square such that each session included only one ID navigation technique.

4.4.2 Task 2: Data entry

Task 2 was a 4 × 6 × 4 within-subjects design. The factors and levels were similar to Task 1 (target selection) except there were only 4 trials in each block. Each trial was a conversion task presented in random order.

The dependent variable for Task 2 was "task completion time": the time to perform a single conversion task starting when the first form element (conversion type) was accessed. The total number of trials was 1536 (16 participants × 4 input techniques × 6 blocks × 4 trials).

5. Results and Discussion

In Task 1 (target selection), the grand means for the dependent variables were 1.37 s for selection time and 0.85% for error rate. In Task 2 (data entry), the grand mean for task completion time was 12.2 s.

5.1 Selection Time (Task 1)

As expected, TAB navigation was the slowest technique with a mean selection time of 2.2 s. The selection time was 40% less for 4-finger ID navigation (1.31 s), 74% less for numeric ID navigation (1.12 s) and 80% less for the

mouse (0.87 s). The differences were statistically significant ($F_{3,15}$ = 95.8, p < .00001). See Figure 3.

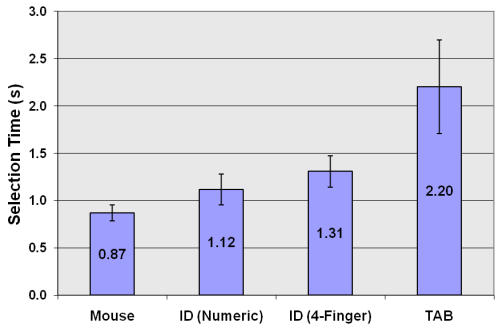

Figure 3. Mean selection time by input technique (Error bars: ±1 *SD*)

Based on selection time, ID navigation is clearly a good alternative to TAB navigation. The mouse was only 22% faster than numeric ID navigation, while both techniques were at least twice as fast as TAB navigation. Additionally, since the number of elements in our test web page was about half the estimated average for a normal web page [2,8], the difference may be even higher for real web pages.

There was a significant effect of block on selection time for all techniques ($F_{3,15}$ = 20.8, p < .00001). See Figure 4.

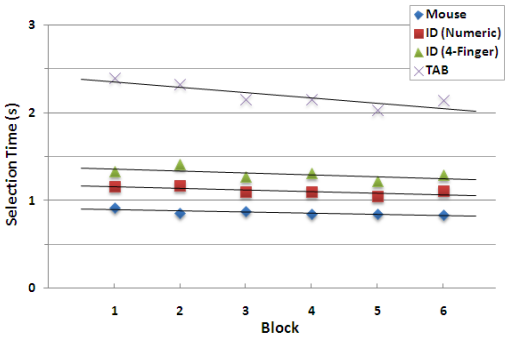

Figure 4: Selection time by block

Numeric ID navigation was 14.5% faster than 4-finger ID navigation. Not everyone who participated in the experiment was comfortable typing with all four fingers and some could only use their index and middle fingers. Thus we expect touch typists to perform faster with 4-finger ID navigation.

5.1.1 Order of the buttons

We expect the order of the buttons (their TAB order on the page) to effect the selection time for TAB navigation more significantly than for other methods. However, TAB order is not the same as the pointing distance for the mouse condition. For example, button "Charlie" in Figure 2a was closest to the START button, while "Alpha" had the smallest TAB order. To reveal these distinctions, we consider the effect of button "row" on selection time. The average TAB order of the buttons in each row is relative to their average pointing distance from the START button.

Figure 5 shows the relationship between selection time and row. While the effect of row on selection time was significant for all techniques ($F_{3,15}$ = 85.8, p < .00001), the linear relationship for TAB chains is clearly much steeper than for all other techniques. The slope of the regression line for TAB (0.55) was 89% steeper than for both the mouse (0.06) and numeric ID navigation (0.06). 4-finger ID navigation was also significantly less steep than with TAB navigation but slightly steeper than with the other two techniques (0.14). This is because the number of digits in the IDs grows faster for this technique. This requires more keystrokes for buttons farther down the page.

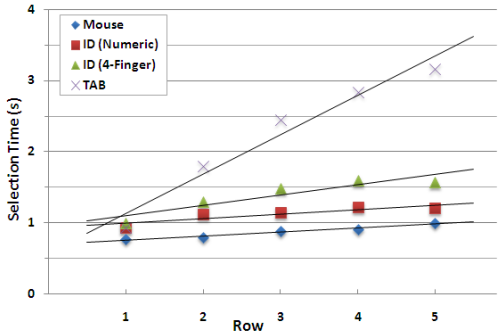

Figure 5. Selection time by row and input technique

In our pilot studies, we observed an interesting strategy of pressing and holding the TAB key until the target was almost reached. To allow participants to use the techniques as they normally would, we allowed this during the experiment. Interestingly, it did not seem to help as in many cases participants overshot the target and had to backtrack using SHIFT. This reverse

traversal was unnatural and seemed to take much longer. In this case, some participants even had to visually scan for the SHIFT key.

5.1.2 Comparing the observations with the predictions

Our observations for both traditional methods fall close to the lower end of the predictions from our keystroke-level model (Table 2). The mean selection time for the mouse was only 1% less than predicted. Optimal performance for the mouse was expected as the buttons were relatively large. This reduces the actual pointing time according to Fitts' Law [10], but is not accounted for in the pointing operator in Card et al.'s [4] original model. The empirical result for TAB navigation was 10% less than the minimum predicted selection time for a page with 25 elements. This is likely because the predictions assume one TAB press at a time, while some participants chose to hold TAB. Additionally, since the predictions are based on typing speed of the participants, repeatedly pushing a single button should be faster. Thus, for users with disabilities or poor keyboarding skills, the selection time using TABS might be even longer.

Table 2
Model Predictions vs. Observations

Method	Model Prediction (s)		Observation (s)
	Min	Max	
Mouse	0.88	1.58	0.87
Tab 25 elements	2.47	18.15	2.2
ID Navigation (2-digit IDs + activation key)	0.24	3.6	1.12 / 1.34

Note: The initial mental preparation time is not included in this table since it was not measured in the experiment.

For both ID navigation techniques, our observations fall within the range of the predictions and below the average. The mean selection time for numeric ID navigation was 33% less than the average and the mean selection time for 4-finger ID navigation was 22% less. These predictions did not take into account the implementation-specific response time needed to load the IDs. With a faster low-level implementation of ID navigation, we can expect better results.

The model assumes expert users and does not take into account "user unpredictability" due to factors such as fatigue and other surrounding factors. While this was a controlled experiment, we could not avoid fatigue or quantitatively measure it. In an initial pilot of this experiment, the performance decreased towards the end of the experiment due to fatigue. To minimize fatigue in the actual experiment, we used more blocks with less trials per block.

Overall, the model provided very good predictions of the *relative* performance of all input techniques. The linear relationship between selection time using TAB and the position of buttons is also clearly visible.

5.2 Error Rate (Task 1)

With invalid button selections as the only errors, mouse navigation was the most accurate with no errors. The next most accurate technique was numeric ID navigation (0.4%) followed by TAB navigation (1.0%). 4-finger ID navigation was the least accurate method with an error rate of 1.9%. See Figure 6. The differences were statistically significant ($F_{3,15} = 6.61$, $p < .00001$).

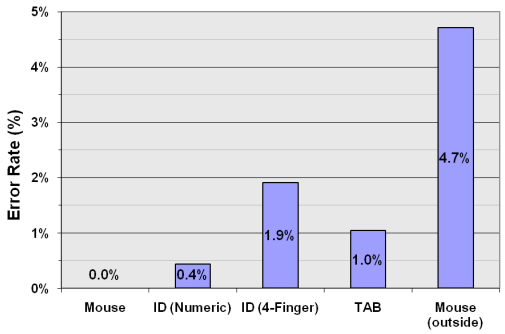

Figure 6. Error rate by input technique

Clicking the mouse in the space between buttons occurred with reasonable frequency, thus the error rate for the mouse above is artificially low. On most real web pages, elements are typically smaller and closer together, so the true error rate for the mouse would likely be higher. On the other hand, having small, tightly packed elements should not affect keyboard navigation techniques. For the purpose of comparison, we recalculated the error rate for the mouse by counting clicks between buttons. In this case the

error rate was 4.7% and significantly higher than all keyboard navigation techniques (the right-most column in Figure 6).

In contrast to our pilot study (10% error rate), the new implementation of ID navigation was 96% more accurate. This was achieved by releasing the activation key instead of using a timeout to confirm selections, as well as other improvements in the implementation. While we also implemented a method to correct selection mistakes, participants rarely made corrections.

5.3 Data Entry Time (Task 2)

Even in Task 2 (data entry), the mouse was the fastest method with a mean task completion time of 9.8 s. The task was completed in 11.7 seconds using numeric ID navigation, 12.1 seconds using TAB-key navigation, and 15.0 seconds using 4-finger ID navigation. The differences were statistically significant ($F_{3,15} = 45.5$, $p < .00001$). See Figure 7.

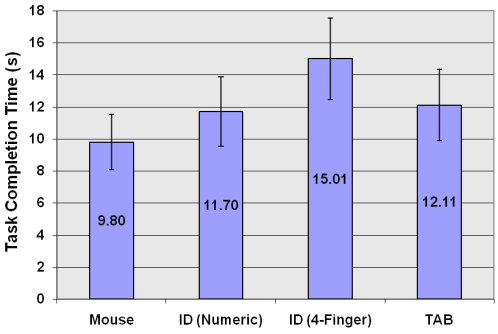

Figure 7. Mean Task 2 completion time by input technique (Error bars: ±1 *SD*)

Keyboard methods are favoured with an increasing number of text entry elements. With the addition of a single text entry field in Task 2, the relative difference between the mouse and numeric ID navigation (16%) was less than the difference during Task 1 (22%).

While numeric ID navigation was 74% faster than TAB in Task 1 (target selection), the difference between the two in Task 2 was not significant ($F_{3,15} = 1.45$, $p > .05$). In a web form with many data entry elements accessed in sequence, TAB is the ideal method to traverse the elements. That is why ID navigation is meant to improve TAB navigation, not replace it.

The slow task completion time for 4-finger ID navigation was mainly due to the numeric nature of Task 2. Most participants confused actual number keys (1 to 4) with the labels used for ID navigation. Some even wanted to use the first four number keys to navigate.

There was a significant effect of block on task completion time for all techniques ($F_{3,15} = 26.1$, $p < .00001$). See Figure 8.

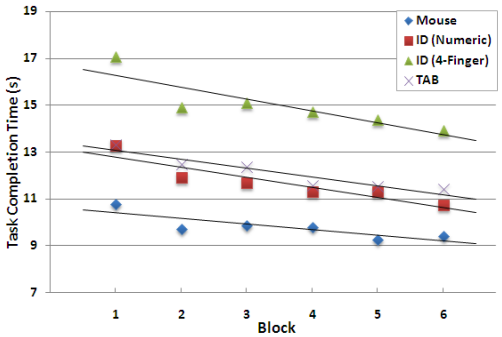

Figure 8: Task 2 completion time by block

5.3.1 Use of a numeric keypad

During the experiment, participants were instructed to use their preferred numeric entry method. All participants used the numeric keypad to enter numbers in the form and all but one used it for ID Navigation as well. Based on these results, *exclusive* use of the numeric keypad for ID navigation is not recommended, since it is already the preferred choice of numeric entry for many users. When used with an activation key however, the numeric keypad was a fast choice despite the small homing time necessary to switch between the main keyboard and the numeric keypad.

5.4 Questionnaire

In a questionnaire, participants were asked to rank the four techniques based on their personal preference. Overall, 14 participants preferred the mouse as their first choice and 15 chose ID navigation as their second choice (or better). TAB and 4-finger ID navigation took the 3rd and 4th spots with more participants in favour of TAB. See Table 3.

Table 3
Overall Ranking of all Four Methods

Method	Rank			
	1	2	3	4
Mouse	**14**	1	1	0
ID Navigation (numbers)	2	**13**	1	0
ID Navigation (4-finger)	1	0	**6**	**9**
Tab	1	1	**9**	**5**

Furthermore, we asked participants to rank all methods only when constant data entry was needed. The results were similar but less unanimous. While 11 participants ranked the mouse as their first choice, 7 preferred a keyboard method. See Table 4.

Table 4
Rankings for Tasks Requiring Data Entry

Method	Rank			
	1	2	3	4
Mouse	**11**	4	1	0
ID Navigation (numbers)	3	**8**	4	1
ID Navigation (4-finger)	0	1	**6**	**9**
Tab	4	2	**6**	4

On a 7-point Likert scale (1 = least comfortable, 7 = most comfortable), participants ranked their level of comfort with all four methods. The mouse was the most comfortable method with an average rating of 6.7, followed by numeric ID navigation (5.4) and TAB (4.8). 4-finger ID navigation was the least comfortable method (3.4). See Table 5.

Table 5
Perceived Level of Comfort with all Methods
(1 = least comfortable, 7 = most comfortable)

Method	# Responses per Rank							Mean Rank
	1	2	3	4	5	6	7	
Mouse			1			1	14	**6.7**
ID Nav'n (numbers)		1		2	4	7	2	**5.4**
ID Nav'n (4-finger)		3	6	4	3			**3.4**
Tab	1	1	2	2	3	4	3	**4.8**

We asked participants if they would enable ID navigation on their own browser. Of the participants who answered yes, 8 preferred numeric IDs while 2 preferred custom IDs. We also asked participants to suggest alternate IDs. Most wanted to keep numeric IDs, some suggested the left-hand version of our 4-finger method (A, S, D, F) and some suggested 4-finger combinations on the numeric keypad such as (7, 8, 9, +) and (NUM_LOCK, /, *, -).

We asked participants to compare our highlighted selection indicator to *Firefox*'s default dashed border. Nine participants preferred the improved method, however only two felt the improved feedback was "required".

Finally, we asked for suggestions to improve ID navigation. One participant thought the large number of digits in the labels (4-finger method) was confusing. On the other hand, a number of participants preferred a numeric method with a smaller subset of keys.

6. Conclusion and Future Work

We have shown that ID navigation can reduce the selection time, compared to TAB navigation, to a constant time. ID navigation can improve existing keyboard navigation techniques and even complement pointing devices for expert keyboard users.

We found numeric ID navigation far superior in performance to the 4-finger method, especially during a task involving numeric entry. However, the 4-finger method should not be abandoned as it may be the only choice in certain applications such as chord keyboards. Additionally, with enough practice, the 4-finger method may be more comfortable for the touch typist.

In the future, it may be helpful to compare alternate choices of labels as well as special input devices for the 4-finger method. It may also be worthwhile to compare ID navigation with slower pointing devices used in notebook computers (e.g., touchpads and pointing sticks).

7. Acknowledgements

We would like to thank all participants for their time, effort, and helpful suggestions. This research is sponsored by the Natural Sciences

and Engineering Research Council of Canada (NSERC).

8. About the Authors

Hooman Baradaran is a graduate student at York University. Scott MacKenzie is a professor of computer science at York University.

9. References

1. US department of justice. Section 508 of the federal rehabilitation act. http://www.section508.gov/.
2. Google code: Web authoring statistics. http://code.google.com/webstats/.
3. Amadio, P. C., Frymoyer, J., Szabo, R. M., and King, K. J. *Repetitive stress injury*. JBJS, 2001.
4. Card, S. K., Moran, T. P., and Newell, A. The keystroke-level model for user performance time with interactive systems. *Communications of the ACM, 23*, 7 (1980), 396-410.
5. Card, S. K., Moran, T. P., and Newell, A. *The psychology of human-computer interaction*. Hillsdale, NJ: Erlbaum, 1983.
6. Chisholm, W., Vanderheiden, G., and Jacobs, I. Web content accessibility guidelines 1.0. *interactions, 8*, 4 (2001), 35-54.
7. Dix, A. J., Finlay, J., Abowd, G. D., and Beale, R. *Human-computer interaction* (3rd ed.). London: Prentice Hall, 2004.
8. Kelly, B. WebWatching UK universities and colleges. 1997. *Ariadne Magazine (Web Version)*. http://www.ariadne.ac.uk/issue12/web-focus/.
9. Leventhal, A. Mozilla keyboard feature: Find as you type. http://www.mozilla.org/access/type-ahead/.
10. MacKenzie, I. S. and Buxton, W. Extending Fitts' law to two-dimensional tasks. *Proceedings of the ACM Conference on Human Factors in Computing Systems*, (New York: ACM, 1992), 219-226.
11. Noe, R. Mouseless browsing extension. http://www.rudolf-noe.de/index.php?/content/view/14/26/.
12. Raggett, D., Le Hors, A., and Jacobs, I. HTML 4.01 specification. *W3C Recommendation REC-html401-19991224, World Wide Web Consortium (W3C), Dec,*, (1999).
13. Schrepp, M. and Fischer, P. A GOMS model for keyboard navigation in web Pages and web applications. *Lecture Notes in Computer Science*, (Berlin: Springer, 2006), 287.
14. Shneiderman, B. The future of interactive systems and the emergence of direct manipulation. *Human Factors and Interactive Computer Systems: Proceedings of the Nyu Symposium on User Interfaces, New York, May 26-28, 1982*, (Intellect Books, 1984).
15. Spalteholz, L., Li, K. F., Livingston, N., and Hamidi, F. Keysurf: A character controlled browser for people with physical disabilities. *Proceedings of the 17th International Conference on World Wide Web*, (New York: ACM, 2008), 31-40.