Automatically Adapting System Pace Towards User Pace -Empirical Studies

Andy Cockburn, Alix Goguey, Carl Gutwin, Zhe Chen, Pang Suwanaposee, Stewart Dowding

Highlights

Automatically Adapting System Pace Towards User Pace – Empirical Studies

Andy Cockburn, Alix Goguey, Carl Gutwin, Zhe Chen, Pang Suwanaposee, Stewart Dowding

- Provides empirical evidence that user preferences for system pace (interface conditions that vary only in the duration of interface timeouts) covary with user pace.
- Reveals characteristics of user performance that can be automatically measured by a system as a basis for automatically adapting system pace.
- Shows that users converge their rate of interaction towards that of the system.
- Empirically demonstrates that fast-paced users prefer an adaptive system pace to a 'one size fits all' static pace.

Automatically Adapting System Pace Towards User Pace – Empirical Studies

Andy Cockburn[a], Alix Goguey[b], Carl Gutwin[c], Zhe Chen[a], Pang Suwanaposee[a], Stewart Dowding[a]

[a] *University of Canterbury, Christchurch, 8041, Canterbury, New Zealand*
[b] *Université Grenoble Alpes, 700, avenue centrale, 38400 Saint-Martin-d'Hères, France*
[c] *University of Saskatchewan, 110 Science Place, Saskatoon, Canada*

Abstract

An interactive application's overall *pace of interaction* is a combination of the user's pace and the system's pace, and if the system's pace is mismatched to the user's pace (e.g., timeouts or animations are too fast or slow for the user), usability and user experience can be impaired. Through a series of four studies, we investigated whether users prefer systems where the system's pace better matches their own pace. All of the studies used common drag-and-drop interactions with hierarchical folder widgets, in which a folder would expand when the cursor hovered over it for a timeout period. If the system pace in these interactions is too fast (i.e., the timeout is too short), then the user's performance and subjective experience is likely to be impaired because of unintended expansions; and if the system pace is too slow (i.e., the timeout is too long), then performance and experience could be impaired by unnecessary delay before folders expand. The first experiment was designed to validate the premise that fast-paced users prefer a fast system pace to a slow one (and the inverse for slow-paced users), and results confirmed this premise. The second study used the first experiment's data to look for measures of user pace that could enable automatic adaptation of system pace, and also examined whether partic-

Email addresses: andy.cockburn@canterbury.ac.nz (Andy Cockburn), alix.goguey@univ-grenoble-alpes.fr (Alix Goguey), gutwin@cs.usask.ca (Carl Gutwin), zhe.chen@canterbury.ac.nz (Zhe Chen), pang.suwanaposee@pg.canterbury.ac.nz (Pang Suwanaposee), stewart.dowding@canterbury.ac.nz (Stewart Dowding)

ipants adjusted their pace towards that of the system. The study found reliable measures of user pace and showed that participants do entrain to the system's pace. The third and fourth studies examined whether users would prefer a system that adapted its pace to the user over a system that used a static baseline pace. Results indicated that a majority of fast-paced users preferred the adaptive interface, but that slow-paced users generally preferred the static baseline interface. We discuss several design implications, including opportunities for systems to improve user experience for fast users by automatically adapting system pace to user pace.

Keywords: Interface pace, adaptation, timeouts, user preferences.

---

1. Introduction

Interaction with computer-based interfaces involves a dialogue between the user and the system in which both parties contribute to the overall *interaction pace*. On the human side (*user pace*) some users will be slow, perhaps due to careful contemplation of each action or to unhurried input device manipulation; other users will be fast, making decisions and manipulations much more quickly. Furthermore, an individual's preferred interaction pace may vary due to factors such as fatigue, workload, illness, or stress.

On the system side of the interaction, many factors within the operating system and the application can influence the *system pace*. On desktop computers, examples include the animated appearance and disappearance of windows, menus, and control panels (e.g., the Windows Start Menu and Mac OS Dock), pop-up tooltips and hotkeys that appear after a hover delay, and accessibility features such as screen readers, which have a speech rate and a hover timeout delay before screen reading initiates. Mobile operating systems and applications also make extensive use of timeouts to discriminate between taps and long-presses, to interpret intentions (e.g., dragging an icon between homescreen pages), and to control input transfer functions (e.g., the iOS keyboard delete key, which increases the deletion rate the longer it is held down).

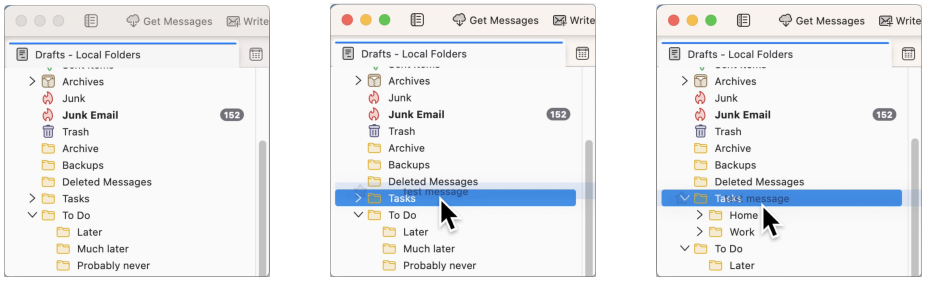


Figure 1: Thunderbird's drag-and-drop folder expansion – folders automatically expand (right) when an object is dragged over a parent item (middle) for longer than a timeout.

Importantly, while preferences for system pace are likely to differ between users (who may themselves work sedately or quickly), system pace is almost always set to a fixed value by the designer in a 'one size fits all' manner. In a few cases designers provide configuration facilities that allow elements of system pace to be customised – but customisation features are known to be seldom used [33, 27, 34], leaving most users with the default set by the designer.

Instead of using fixed (or even user-customisable) system pace, systems could be designed to automatically adapt in order to better match the user's pace. Exemplifying this type of behaviour, Giacolone [18] describes a gambling machine interface that adapts to the user's rate of play. When the user is slow to press the machine's "deal" button, an animation depicts hands slowly dealing cards to the user, but the animation is faster when the user presses the button quickly: *"if one wants to play at a leisurely pace, the game will proceed at its normal play rate, but if the player's excitement level increases and he demonstrates a desire to play faster, the game play rate will be automatically increased"* [18, col. 2:8-11].

There are at least two reasons (further reviewed in Section 2) suggesting that users might appreciate systems that automatically adapt their pace to better match that of the user. First, findings from the communication studies literature indicate a variety of positive outcomes, including liking between interlocutors, when people adapt their patterns of communication to better match one another – the same may be true for human-computer communication. Second,

from a cognitive-motor processing perspective, individuals' reaction times vary (typically characterised by an ex-Gaussian distribution), and therefore a system that requires users to interact at a 'one size fits all' rate is likely to be too fast for some users and too slow for others.

Conversely, however, there are also reasons for suspecting that automatic system pace adaptation may be undesirable. In particular, any change in system pace necessarily introduces an element of inconsistency into the interaction, and the absence of consistency is a well known source of interaction problems (e.g., [38]).

Motivated by the possibility to improve interaction through automatic adaptation of system pace, we conducted a series of empirical studies that examined one specific element of system pace – the timeout that is widely used in hierarchical-folder widgets, such as the Windows File Explorer, the MacOS Finder, and many email clients and IDEs. As exemplified in Figure 1, during drag-and-drop actions, hierarchy widgets commonly use a built-in timeout to determine when a hierarchical item will expand to show its children: items automatically expand when the cursor hovers over them for longer than a timeout period. The length of the timeout can have an important effect on user experience: if the timeout is too short, then unintended item expansions will occur, which is likely to be frustrating; and if the timeout is too long, the user will need to wait for expansions to occur, which is also likely to cause frustration.

We report on four studies that examine issues related to interaction pace, focusing on the potential for improving user experience by automatically measuring the user's pace and adapting system pace accordingly rather than using a 'one size fits all' setting for system pace. The first study (previously published as [20]) tests a basic premise underlying the entire investigation – hypothesis H_1, that user preferences for system pace covary with a measure of their user pace (determined by the time taken to complete a set of initial drag-and-drop trials). Results confirm that fast-paced users prefer a fast-paced interface to a slow one, and that slow-paced users prefer the inverse. The second study reports on new analyses of the low-level data from the first study that investigates

whether in-situ user performance characteristics can be used to *automatically* determine the user's pace, without need for an artificial set of tasks. Results indicate that the time between the cursor entering a target item and the click to select it provides a good indicator of the user's pace and of their preference for fast/slow interfaces. The new analyses also show that users tend to entrain to the system's pace – that is, users interact faster than normal when using a fast interface, and slower when using a slow interface.

Studies three and four tested hypothesis H_2, that users would prefer an interface that dynamically adapted its timeout-based pace towards that of the user to an interface that used a traditional static timeout. Results indicate that there are differences based on whether users are fast or slow: automatic adaptation was preferred by the majority of fast-paced users, but not by the majority of slow-paced users. We conclude that adaptation is likely appropriate for fast users but inappropriate for slow ones – and our results also highlight the importance of various nuances in the design and conduct of experiments relating to interaction pace.

The contributions of this work are as follows:

1. We provide empirical evidence that user preferences for *system pace* (interface conditions that vary only in the duration of interface timeouts) covary with *user pace*;

2. We reveal characteristics of user performance that can be automatically measured by a system as a basis for automatic adaptation of system pace;

3. We show that users converge their rate of interaction towards that of the system;

4. We empirically demonstrate that fast-paced users prefer an adaptive system pace to a 'one size fits all' static pace.

2. Background

Two main areas of related work provide the motivation and background for this research. First, we review findings from communication studies indicating that the convergence of communication patterns between interlocutors is correlated with positive affective outcomes, such as affinity and pro-social behaviour. Second, we review HCI literature that address issues related to interaction pace.

2.1. Inspiration from studies of communication convergence

Communication patterns may *converge* towards one another, increasing similarity or synchronisation – for example, a speaker may talk faster and more energetically than they would normally when conversing with an energetic fast speaker. But patterns may also *diverge* away from one another, such as a person using clipped short sentences to respond to a speaker who is perceived to be verbose when time is short. Convergence and entrainment have been examined and demonstrated across many aspects of speech and communication, including the pitch and loudness of speech (e.g., Levitan and Hirschberg [30]), pronunciation (e.g., Pardo et al. [42]), and the use of gestures (e.g., Chartrand and Bargh [6]) or of lexical constructs (e.g., Bradac et al. [1]). Interested readers are directed to Giles et al. [19], Pardo et al. [42] and Lewandowski and Jilka [31] for more extensive introductions related to these general communication effects.

Our particular interest is on temporal aspects of convergence (i.e., the pace of the interaction dialogue between the system and the user). In communication studies, temporal aspects of convergence have been widely studied through the analysis of speech rate effects. Jungers et al. [28], for example, showed that experimental participants adapted their rate of speech to converge towards that of fast or slow speakers in audio recordings, and results from Schultz et al. [47] show similar effects in scripted turn-taking dialogues.

In a prisoners' dilemma study, Manson et al. [35] observed that speech rate convergence indicated pro-social behaviour (i.e., greater cooperation on the prisoners' dilemma task) and that participants evaluated each other more positively

when their speech rates converged. Manson's findings are echoed in other studies of different communication modes, including the following: Chartrand and Bargh [6] observed greater liking between partners who mimic one another's gestures; van Baaren et al. [50] found that larger tips are given when a confederate waitress mimicked her customers' orders than when she did not; and Pickering and Garrod [44] showed that mutual understanding is enhanced when language is adapted to increase similarity in grammatical structures and word use.

In summary, previous work from communication studies indicates a variety of positive outcomes from communication convergence, including pro-social behaviour and affinity between people.

2.2. Interaction Pace in HCI

During human-computer interaction, both the user and the system contribute to the overall interaction pace. The variability of cognitive and motor aspects of *user pace* has been well documented since early HCI research. Card et al. [2] introduced the notion of 'slowman', 'middleman', and 'fastman', with a wide range of time estimates for even the most basic actions – for example, estimates for the time taken to press the space bar in reaction to the appearance of a symbol varied from 105 ms to 470 ms. While the ideal model of the distribution of individuals' reaction times remains an active area of research, an ex-Gaussian (right skewed) model is prominent [e.g. 24, 22, 46].

Early studies relating to *system pace* focused on the implications of system delays (for example, see [48]). Although there have been relatively few studies of pace convergence in HCI, Shneiderman's review [48] mentions the possibility that the user may converge towards a system's fast pace: *"As users pick up the pace of a rapid interaction sequence, they may learn less, read with lower comprehension, make ill-considered decisions, and commit more data entry errors"* [48, p. 266][1]. Design guidance relating to system delays is now routinely in-

[1] Jakob Nielsen recently made similar points in favour of slow interaction, although with

cluded in undergraduate HCI courses, with approximate threshold times of less than 0.1 seconds for the user to feel that the system is reacting instantaneously, less than 1.0 seconds for the user to maintain an uninterrupted train of thought, and less than 10 seconds to maintain the user's attention [39].

System delays are often caused by compute-bound processes (such as searching a large data structure) or by limitations in network bandwidth, latency, or jitter. While these limitations still influence interaction, hardware improvements typically reduce their magnitude and frequency. However, unlike the delays that are imposed on the system by processing or transmission requirements, interaction effects with temporal properties such as animations and timeouts are often intentionally engineered into systems. These intentionally engineered timeout effects are the focus of our work.

In the following paragraphs we first review the state of the art in contemporary user interfaces, demonstrating the widespread use of temporal effects in current interfaces. We then briefly review HCI literature investigating effects related to interface pace adaptation.

2.2.1. Interaction pace features in contemporary interfaces

We see four main ways that current interfaces are designed to incorporate elements of system pace: animation and motion effects, information rate effects, input device transfer functions, and temporal effects to discriminate user intentions with overloaded input. Other system factors may also influence the overall pace of interaction, such as the availability of interface shortcuts, but in general shortcut facilities are designed to enable the user to obtain the fastest pace possible, rather than the system imposing some element of pace on the user.

Animations are widely used to accompany the appearance and disappearance of basic interface controls such as windows and menus. Animations provide several potential advantages to the user, including softening visual effects

tongue firmly in cheek. `https://www.nngroup.com/articles/slow-ui/`.

that might otherwise be perceptually jarring, and providing a spatial cue to the source and destination of objects when they appear or disappear. For example, in the default configuration of Microsoft Windows 10 the Start Menu appears with a fast-in-slow-out animation, windows zoom and fade when they appear/disappear from the taskbar, and menus are animated. Operating systems typically provide personalization options to reduce or disable animations, and some provide expert configuration options to set the duration of animations (e.g., Windows allows users to edit the MenuShowDelay value in the Windows system registry). Animations such as those described above are typically of short duration at around 300 ms. Interested readers are directed to Chevalier et al. [8] and to Hudson and Stasko [25] for high quality reviews of animation effects.

Information rate effects determine the amount of information presented to the user per unit time, and they can be related to animation effects. These effects are commonly used to manipulate the difficulty of computer games. For example, a game might present waves of items to contend with in each level (e.g., asteroids or enemies), manipulating the number of items and the time available – Denisova and Cairns [12] discuss the use of 'time manipulation' techniques to increase game immersion, and Vicencio-Moreira et al. [51] describe in-game methods for adjusting difficulty to balancing skill across players. Visualisation techniques such as Rapid Serial Visual Presentation also manipulate information rate to assist activities such as rapid comprehension or search [11, 52].

Input device transfer functions are used to translate low-level signals received from input devices into output control effects displayed on the screen. For example, transfer functions determine the mapping between mouse and cursor movement, and on mobile devices they determine how swiping gestures influence scrolling movement. These functions are often sophisticated, attempting to appropriately amplify the user's input when it appears that the user wishes to move quickly, yet diminish input when it appears that the user wants precision. Transfer functions can therefore influence the system's pace of interaction. For example, a low-acceleration scrollwheel transfer function will require the user to

repeatedly rotate the wheel to move through a long document; but conversely a high acceleration transfer function risks a twitchy behaviour that results in extensive over-shooting. Operating systems and input device vendors typically provide configuration interfaces that allow the user to directly control transfer function behaviour for cursor movement and scrollwheel operation. Interested readers are directed to Quinn et al. [45] for a discussion of transfer functions in touch scrolling and to Casiez et al. [4] for their use with mouse input devices.

Temporal discrimination with overloaded input is the area examined in our experimental work. Input devices offer a limited vocabulary of possible actions for the user to express their intent – for example, a mouse will typically have only two or three buttons, a scrollwheel, and a displacement sensor, and a touchscreen may report only the coordinates of one or more contacts. To increase the user's ability to express varied intentions designers often exploit the temporal components of user actions to discriminate intent. For example, using a mouse, two successive clicks are only interpreted as a double-click if they occur within a timeout period. If this time period is too short for the user then they will fail to reliably double-click, but if it is too long then the system may misinterpret separate manipulations as a single double-click action. Temporal discrimination is also widely used on mobile devices – for example, on the iOS homescreen a single tap on an icon opens the object, a long press (a press longer than a timeout) posts a context menu, and a very long press (longer than another timeout) enters a reconfiguration mode.

Related timeout methods are also commonly employed in interfaces when the input mechanism used for a particular action is temporarily unavailable because it is consumed by an ongoing user action. For example, in hierarchical browsers a mouse left button click is used to expand a parent object and reveal its children. But during drag-and-drop actions the left button must remain pressed because releasing drops the dragged item onto the underlying object. Therefore, designers use a hover timeout to resolve the problem of overloaded input – if the cursor hovers over a parent item for longer than the timeout then the underlying item expands to show its children. This hover-expansion

technique (also named 'spring-loaded folders' [17, 10]) is widely used across platforms and applications, including Windows Explorer, Mac Finder, most email clients, and many programming IDEs, and similar timeouts exist in a wide range of interfaces, including dragging items across homescreen pages on mobile devices.

System timeout values such as these influence a system's pace of interaction. In general, long timeouts permit slow-paced interaction, but they also risk frustrating users who want to proceed more quickly. For example, a user who wants to drag an icon across several homescreen pages on their phone must wait for the timeout to expire at the edge of each page. Conversely, short timeouts permit fast interaction but risk frustrating the user by incorrectly identifying their intention – the user might accidentally change to the next homescreen page as a result of briefly dragging an item near the edge of the screen.

To gain insight into the timeout values used in current software systems, we used a screen recorder to inspect the timeouts used for hover expansion in the Thunderbird email client and in the Mac Finder. The Thunderbird timeout value was fixed at 1000 ms, but the July 2023 release (version 115) reduced the value to 170 ms, creating frustration for users, including one of the authors, due to unintended folder expansions while filing email[2]. The Finder's value varies depending on which view is enabled. By default, the timeout is 600 ms in the column view, and 1000 ms in other views, although the user can configure the timeout using the 'spring-loaded delay' setting in System Preferences (the range of values available with the setting is 200-1200 ms in the column view, and 500-1200 ms in other views). In many systems the timeouts are configured in a 'one size fits all' manner, and even if customisation facilities are provided they are known to be seldom used [33, 27, 34]. Consequently, users who would prefer to proceed more quickly or more slowly are likely to be constrained by an invariant system pace.

[2] `https://www.reddit.com/r/Thunderbird/comments/162lh4m/auto_expand_subfolder_while_dragndrop_over_it/`

2.2.2. HCI research and interaction pace

As mentioned above, Shneiderman considered interaction pace within his seminal review of system delays [48], and Dix [13] directly examined interaction pace in the context of computer-mediated collaboration between people. A few authors have contemplated how users might benefit from interfaces that promote slower, reflective, laid-back and restful interactions in mobile search [26] and in a music player that adapts to the user's pace [15]; similar issues were also discussed at a DIS conference workshop [40]. Others have scrutinised how aspects of performance and satisfaction change as users are prompted to alter their pace of interaction, demonstrating a strong increase in errors in a game [36] and in abstract pointing tasks [53].

HCI researchers have also examined issues related to communication convergence, particularly in speech interaction. In an early and comprehensive study Oviatt et al. [41] demonstrated that 70-95% of child participants (aged 7-10) quickly converged their rate of speech towards that of an animated agent. More recently Dohsaka et al. [14] examined convergence in the opposite direction – where the duration of the agent's speech pauses converged towards that of the human – with Likert item responses suggesting positive subjective outcomes from system convergence. Related positive findings have been demonstrated for automated speech rate adaptation in synthesised Mandarin speech [7]. Other recent studies have examined speech convergence and entrainment effects to promote positive turn-taking behaviour in human users and to build positive social responses to the system [29, 32].

The previous HCI study most closely related to ours is the PhD thesis work of Yu [55]. The thesis examines the effects of timing on users' perceived control when interacting with intelligent systems, and it examines related effects including pace entrainment between humans and their systems. The key finding from three controlled experiments was that consistency in system timing is beneficial for subjective experience, enhancing the user's sense of control, and reducing perceived stress and effort; inconsistent timing, such as that provided by adap-

tive system pace, had negative effects, including elevated stress and effort, and harming subjective experience.

3. Study 1: User Pace and Preference for a Fast versus Slow Interface

With the exception of the findings of Yu [55], the review of prior work suggests that users' preference for interfaces may be enhanced if the system's pace is more similar to the user's own pace. Furthermore, there may be opportunities to improve users' overall preference for interfaces by having the system pace converge towards that of the user. However, Yu's findings suggest that the pace inconsistency of an adaptive system could negate any benefits of automatic pace adaptation.

We therefore conducted an initial study to validate the underlying premise that fast users would prefer a fast system pace to a slow one, and that slow users would prefer a slow system pace to a fast one. The hypothesis is formally expressed as follows:

H_1 User preferences for system pace (as exhibited through system delay timeouts) covary with user pace — faster users show stronger preference for shorter timeouts, and slower users show stronger preference for longer timeouts.

If supported, this hypothesis suggests that system designers could improve user preferences by matching the system's interaction pace (e.g., timeout duration) to the user's pace of interaction, in a form of interface pace convergence. However, there are at least three reasons to think that the hypothesis may not be supported. First, the measure of user pace may be insensitive, providing poor distinction between users. Second, measured user pace may be a poor predictor of user preferences for system pace – for example, all users might prefer faster (or slower) system response, regardless of their user pace. Third, the hypothesised effect (if any) may be sufficiently small to make it impractical to demonstrate at reasonable experimental scale.

To briefly summarise the method, participants first completed an initial series of drag-and-drop actions that did not involve the timeout-based hierarchical folder-expansion feature, with the data used to characterise their pace of interaction. Each participant then completed a series of hierarchical drag-and-drop tasks using two interfaces – one fast and one slow – that differed only in the hover timeout required before a hierarchical item would automatically expand to reveal its children. They then selected which of the two interfaces they preferred. Finally, they used a slider to configure and test their preferred timeout for a final series of drag-and-drop tasks.

3.1. Task Interactions

We based Study 1 (and Studies 2-4) on drag-and-drop behaviours similar to those widely used in hierarchical file and email interfaces. Subjects had to drag a series of boxes onto targets located in a hierarchical structure. At the beginning of each task the structure was fully contracted, and when the cursor hovered over a hierarchical item for longer than a timeout period the item automatically expanded to reveal its child items. Each time an item expanded, any previously expanded item at the same hierarchy level was contracted, so at most a single item at each level of the hierarchy could be expanded at once. The expansion/contraction of hierarchical items was not animated. The item beneath the cursor was highlighted light blue (see Figure 1), and the blue highlighting flickered if the item under the cursor could be expanded.

We chose to base the experiments on hierarchical drag-and-drop for four main reasons:

1. *Familiarity* – the participants would be familiar with these behaviours from everyday computer use and should therefore be able to understand task requirements without extensive training.

2. *Simple interface actions that are readily modelled* – the participants' actions in completing the tasks include simple manipulations that are amenable to well-validated models such as Fitts's Law [16].

3. *Ease of experimental control* – related to the previous point, system properties within these behaviours can be precisely configured, including the distance that an icon must be dragged, the target size, and the hover timeout before an item will expand.

4. *Ecological validity* – several common interfaces (including the Mac OS Finder, Windows Explorer, and most email clients) include system-imposed hover timeouts for hierarchy expansion during drag-and-drop, and some provide configuration controls for customizing the timeout duration.

Each task was cued by showing a blue icon containing a number series at the top of the display. The number series, such as "3.1.2", indicated the required target. The hierarchical structure was shown immediately beneath the cued item, initially showing the fully contracted view of sixteen items, enumerated 1 – 16 (see Figure 2b, which shows a partially expanded hierarchy).

3.2. Timeout setting and pilot studies

The key manipulation in the study was the duration of the hover timeout. If the timeout was too short (i.e., the system pace was too fast for the user), then unintended items would expand while the user dragged the item towards the target. And if the timeout was too long (i.e., the system pace was too slow for the user), then users would be overly delayed, potentially causing frustration.

The choice of timeout values for the 'fast' and 'slow' interfaces was critical for our experiment. Ideally, across the full set of participants roughly half would prefer the 'fast' interface and half the 'slow' interface. And inappropriate timeout selection would create ceiling or flooring effects, such as the strong majority of participants preferring the slow interface if the fast timeout was much too fast.

We conducted a series of small pilot studies using the method described below, but varying only the fast and slow timeout values. The first pilot study (n=15) used values of 450 ms and 900 ms, with 100% of participants preferring 450 ms. The second pilot study (n=5) used 400 ms and 800 ms, and again 100%

preferred the faster condition. The third pilot (n=12) used 200 ms and 800 ms, with 70% preferring the slow condition. A final pilot study (n=10) used 250 ms and 750 ms, with a 50:50 split in preferences. We therefore used these values of 250 ms and 750 ms for Study 1.

3.3. Procedure

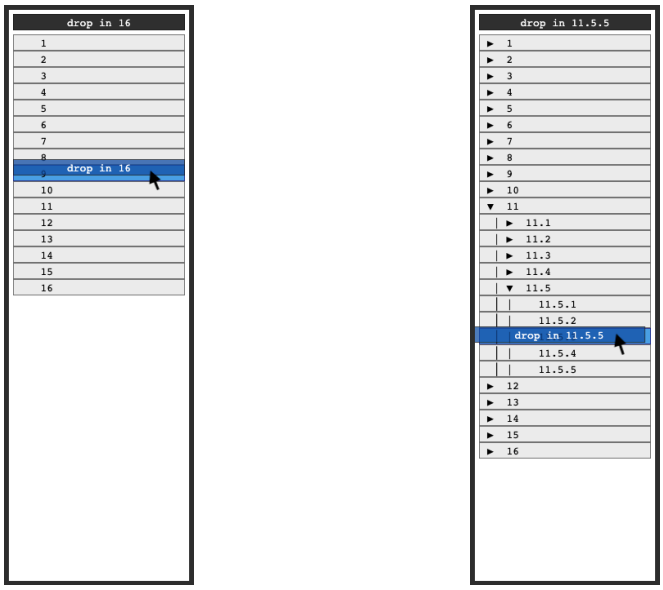
We conducted Study 1 on Amazon Mechanical Turk. Each participant proceeded through three experimental stages: 1. user pace determination; 2. system pace experience and preference; 3. setting and experience of user-tailored system pace.

3.3.1. Stage 1. User pace determination

After collecting background demographics (age, gender, pointing device, frequency of computer use and game play), participants completed a set of one-dimensional drag-and-drop trials. A web page instructed participants that they needed to drag a blue box containing a number onto the target grey box showing the same number. Participants clicked a button labelled 'Start Tasks' at the bottom of the instruction page when ready to continue.

Sixteen grey target boxes were vertically arranged below the blue draggable box, enumerated 1 – 16 (see Figure 2a). The dragged box moved with the cursor and turned green while the cursor was over the target. Participants completed each trial by dropping the dragged box while it was green. Correctly completing one trial caused the next target to be immediately displayed in the blue box at the top of the display. If the box was dropped onto the wrong target, the same blue box was animated moving back at the top of the list to notify participants that they needed to redo the trial (preventing participants from racing through the experiment without regard to accuracy).

Each participant completed 20 drag-and-drop trials, consisting of four familiarisation tasks (dragging items to randomly selected odd-numbered targets) and 16 controlled tasks (two repetitions for each of the even-numbered targets).



(a) Stage 1 task (no hierarchy).

(b) Stage 2 & 3 task (hierarchy).

Figure 2: Drag-and-drop tasks. The item was torn off from the top and dragged to the numerical target.

Data from the 16 controlled tasks was used to determine each participant's user pace, as determined from their mean time on error-free trials.

3.3.2. Stage 2. System pace experience and preference

On completing the final task in stage 1, a web page displaying instructions was automatically shown for the next stage. In stage 2, participants completed trials with two different settings for system pace, and then chose their preference. The instruction page included a sample hierarchy and instructed participants to drag the blue box to the target identified by its number string, such as '5.2.3', meaning that the item should be dragged to item 5 at the first level, item 2 at the second level, and items 3 at the third level (see Figure 2b). Participants were required to complete four familiarisation trials on the instruction page, each involving dragging the box to a target at the third level of the hierarchy. In these familiarisation trials, each level of the hierarchy opened after a hover timeout of 1000 ms.

Having completed the four familiarisation trials, a 'Next' button appeared, and clicking it advanced to a new page informing participants that they would use two systems – 'System A' then 'System B' – to complete two sets of hierarchical drag-and-drop trials. Participants clicked a button labelled 'Start System A Tasks' when ready to proceed. After completing the set of trials with System A, they then clicked a button labelled 'Start System B Tasks' to begin the second set of trials.

Similar to the previous stage, the blue box to be dragged was shown at the top of the display, immediately above sixteen grey box items, numbered 1 – 16 and prefixed with a symbol '►'. Consistent with many commercial interfaces, the ► symbol indicted that the box contained children. At the start of each trial all items within the hierarchy were collapsed. When the dragged blue box hovered over a hierarchical parent item for longer than the timeout period, the item would expand to reveal its hierarchical content (see Figure 2b). Only one item at each level of the hierarchy could be expanded at any time, so when a hierarchical item was expanded, any previously expanded item at that level was automatically contracted. Each of the sixteen top-level items contained five child items (enumerated 1-5), and each of the child items contained between one and five grand-child items – item $n.1$ contained four items, $n.2$ three, $n.3$ two, $n.4$ one, and $n.5$ contained five items. This structure was used to reduce target item spatial stability when unintended items were expanded (as normally occurs in real data hierarchies) – if all items had the same number of children then the location of a target would remain constant when one item expands and another contracts, for example, item 10 would remain spatially stable while the user dragged downwards to replace the five children of 2.1 with five children of 2.2. As at most only one item at each level of the hierarchy could be expanded, the maximum number of items displayed at once in the hierarchical view was 26 (16+5+5), plus the dragged item.

The dragged box turned green when it was over the final target. Dropping the box while green completed the trial, causing the next trial to be immediately displayed. Dropping the box on the wrong target caused the trial to begin anew,

with the hierarchy fully contracted. If the cursor was moved outside of the list while dragging the box, the same blue target item was re-displayed at the top of the list (preventing participants from avoiding the expansion feature).

Each participant completed 16 trials with each of the two systems ('A' and 'B'), starting with four familiarization trials, and then 12 trials comprising one selection at each of the top-levels targets in the range 3 – 14. Each second and third level target was always item 5. We avoided using the top-most and bottom-most top-level items because they are least likely to induce unintended expansions (e.g., there is no expandable item beneath item 16, so the user can slowly approach item 16 from below without risking unintended expansion, unlike other locations). We always used the last item (item 5) at the second- and third- level because it is the most likely item to suffer unintended expansions – for example, dragging off the bottom of target item 6.5 or 6.5.5 risks the unintended expansion of item 7.

Having completed all of the trials with both Systems A and B, participants were asked to respond to the following forced-choice question:

If completing these tasks again, I would prefer to use:
☐ System A ☐ System B

The only difference between System A and System B was the timeout duration used to trigger hierarchy expansion. The two settings were *fast* (250 ms) and *slow* (750 ms). For half of the participants, 'System A' was fast and 'System B' was slow, with the inverse for the other half.

3.3.3. Stage 3. Setting and experience of user-tailored system pace

Up to this point in the experiment the participants had not been informed about the use of different timeouts in the systems. A web page instructed participants that Systems A and B differed only in the length of the hover timeout used for item expansion. The web page included a slider showing the time settings used for Systems A and B (see Figure 3). A slider handle allowed users to set the timeout for an upcoming set of trials with System C. The web

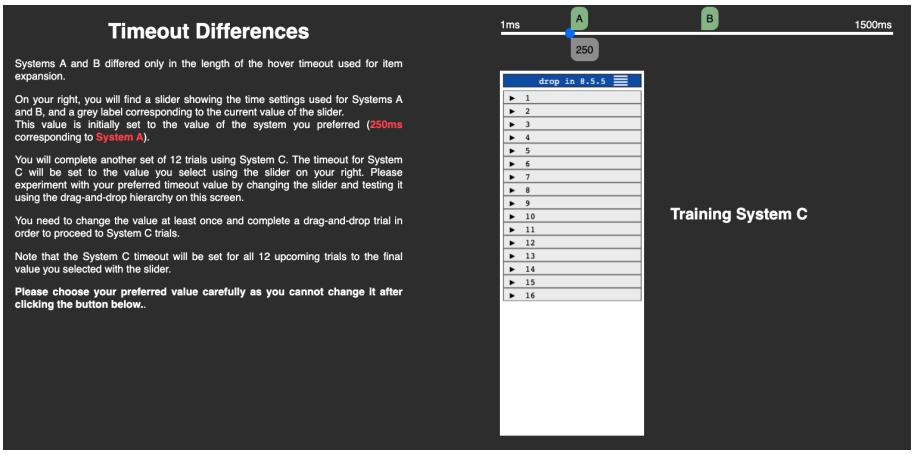


Figure 3: Timeout configuration page and sandbox hierarchy, used to set timeout in Stage 3 of Study 1.

page also included an interactive 'sandbox' version of the System C interface, showing the same hierarchy as that used in Stage 2.

Participants were instructed that they would complete a final set of drag-and-drop trials using 'System C', and that they should first use the slider to set their preferred timeout value in the range 1 – 1500 ms. The timeout was initially shown to match that of their preferred option (i.e., that used by their preferred of System A or B). Once the participants had manipulated the slider at least once, and completed at least one selection using the sandbox interface, a button at the bottom of the page was enabled stating 'Start System C Tasks'.

Finally, participants were asked to type any final comments they might have about the experiment. The study terminated with a web page that thanked them for their time.

3.4. Subjects and Apparatus

Study 1 involved 208 participants on Amazon Mechanical Turk. Conditions for inclusion were that each crowdworker had to be based in the United States, with a HIT approval rate greater than 90%, with at least 1000 approved HITs, and with no prior participation in the study (these conditions are used across

all of the crowdworker studies 1, 3, 4, and 6). The participants' mean age was 36.6 years (s.d., 10.5, min. 18, max. 70); 61.5% self reported as male, 37.5% as female, and 1% declined to answer. Participants were asked to report the input device used for the study, with 83% reporting that they used a mouse, and 17% using a trackpad. Operating system use was divided between Microsoft Windows (89%), MacOS (8%), and Linux (3%).

The experiment took approximately 10 minutes to complete, with participation rewarded with a payment of USD$2.50 (apportioned from an hourly rate of USD$15 per hour).

The experiment ran within each participant's web browser, allowing precise control of the drag-and-drop interface, including control of the hover expansion timeout with millisecond granularity. Software was written in HTML/CSS/JS, using the Paper.js library and a Firebase database to log all user actions, including the time taken to complete each of the trials and any errors made.[3] The software automatically detected whether the zoom level of the browser window was sufficient to display the full height of the expanded menu without scrolling, and if not, it prompted users to zoom out until the condition was satisfied.

3.5. Design

The hypothesis H_1 (that fast users have stronger preference for fast timeouts, and slow users have stronger preference for slow timeouts) is tested through two main analyses.

The first analysis compares the proportions of participants who choose the *fast* system in preference to the *slow* system (when choosing between System A and System B) across three quantile bands of user pace classification (Q_1 (fast), Q_2 (medium), and Q_3 (slow)) based on their performance in the stage 1 tasks. The dependent measures for this analysis are as follows:

[3]The software can be accessed and run at `https://www.csse.canterbury.ac.nz/andrew.cockburn/AdaptivePace/`. Anonymised data and analysis scripts for all studies is available in supplementary materials for this paper, including data for an additional related study (n=250) that is not reported in this paper.

1. *user pace classification* – the classification of each user's pace was based on their stage 1 data, dividing participants into three quantile bands based on their mean drag-and-drop completion times (Q_1, the fastest 33⅓% of participants; Q_2, the second quantile, representing the middle third of participants; Q_3, the third quantile, representing the slowest third of participants). We used three quantiles (rather than two or other value) for three reasons: first, to provide a simple and natural alignment with our hypothesis; second, to provide a clear separation between fast and slow categories; third, to ensure a large sample within each of the fast and slow categories.

2. *system pace preference* – the participant's binary preference choice of either the *fast* or *slow* interface (i.e., the preference of either System A or System B after stage 2 of the study).

The hypothesis is tested using a χ^2 test of proportions. The hypothesis requires that a higher proportion of participants who are classified within 'Q_1 (fast)' choose the *fast* interface than do those who are classified within 'Q_3 (slow)'.

The second analysis examines the relationship between the participants' user pace (as indicated by their mean performance in the stage 1 trials) and the timeout value that they select using the slider for System C. The dependent measures for this analysis are as follows:

1. *user pace* – each participant's mean time on stage 1 trials.

2. *System C timeout setting* – the time that each participant selects as desirable for their stage 3 trials with System C.

In this second analysis, support for the hypothesis requires a positive correlation between user pace and System C timeout setting – fast-pace users with a low mean time on stage 1 trials should set a low timeout value for System C, and slow pace users should set a high timeout value for System C.

Additional secondary analyses are also conducted to further characterise the results, including analyses of the effects of participant gender and gaming experience, as well as participants' comments.

3.6. Study 1 results

3.6.1. User pace characterisation from stage 1 trials

The analysis of results requires first determining each participant's *user pace* from their performance in stage 1 trials. We only included data for correct selections because including the time for errors could misrepresent a 'rushing' user who is fast but inaccurate as having a slow pace of interaction.

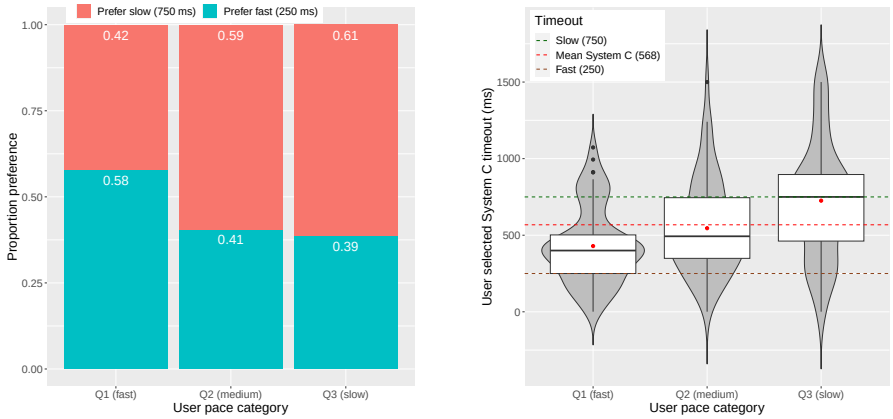
Participants' mean time for stage 1 trials ranged from an extremely fast 656 ms to a sedate 4100 ms, with an overall mean of 1544 ms (s.d., 624, 95% CI [1458, 1631]).

We computed quantile values that split the participants into three equally sized pools according to their mean performance on stage 1 trials, classifying users as Q_1 (fast), Q_2 (medium), and Q_3 (slow). The fastest 33⅓% of participants were categorised as Q_1 if their mean trial time was less than 1219 ms; the middle 33⅓% were classified as Q_2 if their mean was in the range 1219-1567 ms; and the slowest third of participant were categorised as Q_3 if their mean was greater than 1567 ms.

3.6.2. Preference for fast or slow system A/B, across user pace characterisation

Although not part of our hypothesis, there was a general preference for the first system experienced, with 59% of participants choosing System A (used first) over System B ($\chi^2 = 6.58, p = .01$). Similar preference rates for the first condition in forced-choice experiments have been observed in other studies (e.g., Harrison et al. [21]).

Figure 4a summarises the main result, showing the proportion of users in each classification quantile that selected the fast (250 ms timeout) or slow (750 ms timeout) System A/B as their preferred interface. Importantly, 58.0% of users who were classified in the Q_1 quantile selected the fast 250 ms system



(a) Proportion of participants choosing the fast (250 ms) or slow (750 ms) system A/B, across classification quantile.

(b) Distribution of user-selected System C timeouts across quantile.

Figure 4: Main results from Study 1 across 33⅓ quantiles of user pace (Q_1 fast to Q_3 slow). In all box-whisker plots red dots show the mean, black dots show outliers, the middle bar shows the median, hinges show first and third quartiles, and whiskers extend to max/min value within 1.5× the inter-quartile range from the hinge.

as preferred to the slow 750 ms system. This preference for the fast interface decreased to 40.6% for the medium-paced Q_2 users, and to 38.6% for the Q_3 users. A χ^2 test of proportions shows that data as extreme as this sample should seldom occur in the absence of an underlying effect: $\chi^2 = 6.35, p = .042, \omega = 0.17$[4]. This supports H_1, with further analyses of H_1 below.

3.6.3. System C timeout setting across user pace

The second main analysis concerns the relationship between *user pace* and participants' preferred slider setting for the System C timeout. The mean value that participants set across quantiles is summarised in Figure 4b, from a low of 429 ms (s.d., 250 ms) for Q_1 participants, through a mean of 546 ms (s.d., 302 ms) for Q_2, to a maximum mean of 726 ms (s.d., 375 ms) for Q_3 participants. Analysis of variance for this data yields $F_{2,205} = 15.77, p < 10^{-5}$. In addition,

[4]Cohen's ω effect size [9, 37].

there was a positive linear correlation between participants' mean time on stage 1 trials and their selected timeout value for System C ($r = 0.41, p < 10^{-5}$).These results all support H_1.

3.6.4. Effect of gaming frequency and age

As many computer games expose users to a demanding pace of interaction, we wondered whether the frequency of gaming use might influence preferences for timeout duration. We therefore conducted a Spearman rank correlation between participants' self-reported frequency of gaming use and their setting for the System C timeout, which showed at most a weak relationship ($\rho = 0.13, p = .058$), suggesting that gaming frequency was not a strong influence on our results. Finally, analysis of the relationship between participants' age and their setting for System C timeout also showed no clear relationship (Pearson $r = .08, p = .24$).

3.6.5. Participant comments

Participants' comments amplified the quantitative observations above. Many of the most forceful comments were from the fastest users who expressed strong dislike for the 750 ms timeout in System A/B. For example, the participant with the fastest mean time on stage 1 trials (participant 1389, mean 656 ms) commented *"I hated the length of the delay in system A."* This participant set the System C timeout to 180 ms. The participant with the second fastest mean time on stage 1 trials (participant 1100, mean 808 ms) set the System C timeout to the minimum value available at 1 ms, but subsequently commented that this was too short: *"When I did the initial trials I thought that setting it to the quickest speed would be the most efficient and it was for many of the trials. But, if I got off track at all, it was harder to get back to the right section to make the drop. The overall speed was a positive but it wasn't perfect. Doing a few more trials now makes me think that something around 75 ms would work best for me. It's just enough of a delay for me to easily get in the right box, but not so much of a delay as to be annoyingly slow."*

At the other end of the spectrum of user pace, several participants in the slow Q_3 quantile made comments referring to the frustration of the timeout being too short. For example, participant 1193 (mean 3254 ms) commented that she had *"a difficult time controlling the mouse to move the lines down to the correct line"* and that her selected timeout value of 857 ms was selected *"because I thought that maybe I could keep up with it better"*, which suggests a desire for an improved match between her pace and that of the system. Similarly, participant 1335 (mean 3760 ms) commented *"It was aggravating when the numbers would drop down before I wanted them to, the precision was very difficult"*, and he set a high timeout value of 897 ms, further commenting after completing the System C trials that even this was too fast *"If I could, I would have gone back and chosen more time before the drop-down occurred."*

3.7. Discussion of Study 1

The results and participant comments from Study 1 indicate that user preferences are positively influenced when the system's pace better matches the user's pace. Participant preferences for slow and fast system pace aligned with our categorisation of participants as having slow, medium, or fast user pace, and the timeout values that users explicitly set for their upcoming trials correlated with their mean time on an earlier set of trials that did not involve a timeout. In short, as hypothesised, fast users preferred a faster interface, and slow users preferred a slower one.

Although *some* interfaces provide facilities that allow the user to configure timeouts many others do not, leaving users with a 'one size fits all' system pace. And even when customisation facilities are provided they often go ignored, and they necessarily increase the complexity of the interface presented to the user. Our results suggest that instead of invariant pace or customisation facilities, interfaces could be designed to observe the user's pace of interaction, and automatically adapt interface properties to converge towards the user's pace. In this way, the interface might become more responsive (e.g., a shorter timeout) when used by a fast user or when the user is inferred to be rushing, and the

timeout might increase when the user is slow or interacting more leisurely.

However, the conditions tested in Study 1 did not involve the system actually converging to the user's pace. Instead, the method measured each users' mean time in completing an artificially imposed set of drag-and-drop trials (the time from picking up the box to dropping it on the target); this mean was then used to categorise participants as fast, medium, or slow, based on $\frac{1}{3}$ quantiles of the distribution. And all participants chose their preferred between a fast (250 ms) and slow (750 ms) interface.

This procedure is deficient as evidence for the viability of automatic interface pace adaptation in three ways:

1. *Data source for determining user pace.* The method required users to complete a set of artificially imposed 'stage one' drag-and-drop trials in which every trial had a clear starting and terminating action (initiating the drag at a specific location, and dropping the item on a predetermined target). However, real interaction normally lacks such clear start and end points. For example, it can be very difficult to determine where and when a particular pointing action begins, and whether the target is successfully hit or missed [5]. While the use of a set of specific trials served our experimental purpose in characterising each user's pace, this method would not be acceptable in a real interface, where users want to get their own work done and would likely resist completing an artificial set of system-imposed tasks.

2. *Posthoc assignment to quantiles.* The assignment of each participant to the fast, medium, and slow quantiles was conducted in hindsight – *after* all of the participants had completed the experiment. Instead, in a real deployment of adaptive pace, the system would need to measure and adapt to each user's pace in-situ.

3. *Fast versus slow choice.* Participants chose their preferred of two interfaces – one that was intentionally fast and one that was intentionally slow. However, in a real deployment designers would choose a sensible 'one

size fits all' baseline value that is intended to be neither too fast nor too slow for the majority of users. Therefore a more ecologically valid experiment would involve a less sensitive choice between a baseline pace and an automatically adapted pace.

The following study further examines the dataset from Study 1 to characterise aspects of the users' pace so that these three limitations can be overcome in Studies 3-4.

4. Study Two: Characteristics of User Pace

Study 2 consisted of a series of posthoc analyses of the Study 1 dataset, with a focus on two aspects of user pace: first, what characteristics of the user's pace might be available to a system as a basis for automated system pace adaptation?; second, do users adapt their pace towards that of the system (as suggested by Shneiderman, see Section 2.2.2).

As mentioned above, traditional measures of user pace (such as Fitts's Law analyses) are problematic for automatic pace determination due to the difficulty of inferring when the user's pointing action begins and when it accurately ends. However, we wondered whether other aspects of the user's interaction might permit reliable categorisation of the user's pace. In particular, we wondered whether fast-paced users might move the cursor more quickly than slow-paced users (*mean cursor velocity*), whether they would have a higher peak velocity per trial than slow paced users (*mean maximum cursor velocity*), or whether fast users might exhibit a shorter delay between entering a target and releasing the mouse button to select it (*confirmation time*).

4.1. Cursor velocity

First, we determined *mean cursor velocity* and *mean maximum cursor velocity* for each participant in their stage 1 trials. To calculate these values, we first measured cursor velocity at every Enter event in stage 1 (i.e., the velocity when the cursor entered each of the 16 target items). *Mean cursor velocity*

was then calculated for each participant by determining the mean of the Enter velocities in each trial, and then calculating a grand mean across all trials. *Mean maximum cursor velocity* for each participant was calculated by finding the maximum Enter velocity for each trial, and then calculating a mean of these values across all of the stage 1 trials for that participant.

The effectiveness of these two values for predicting users' preference for system pace was tested in two ways: first, using linear correlation of the variable of interest (*mean cursor velocity* and *mean maximum cursor velocity*) with the participant's selected System C timeout; and second, using binomial logistic regression of the variable of interest with the participants' preference choice.

For *mean cursor velocity*, there was a significant negative linear correlation with user selected System C timeout ($r = -0.3, p = 8.5 \times 10^{-6}$), providing some indication that users who moved the cursor faster on average preferred a lower value for System C (i.e., faster response). However, binomial logistic regression showed no significant relationship between *mean cursor velocity* and the participants' preference choice for the fast or slow interface ($p = 0.18$). Results were weaker for *mean maximum cursor velocity*, showing only a marginal negative linear correlation with System C timeout ($r = -0.14, p = .046$), and no binomial logistic regression fit with the participants' preference choice ($p = .66$).

These results indicate that neither *mean cursor velocity* nor *mean maximum cursor velocity* provide a good basis for adapting system pace to user pace. We suspect that these measures are too noisy – for example, a user with high cursor velocities might tend to overshoot the target more, causing slow overall performance.

4.2. Confirmation time

Next we examined *confirmation time*, which, for each participant, was the mean time between the cursor entering the final target in stage 1 trials (which does not expand) and the mouse button being released to drop the dragged object onto the target. To some extent this time represents the user's reaction time to observing that the cursor has entered and become stable within the

target. This time can be automatically measured by an interface for any drag-and-drop action, so it is a viable candidate for automatic pace determination. Our analysis determined *confirmation time* from the mean of the final three stage 1 trials for each participant – that is, we only analyse the trials that were used to classify participants' pace in Study 1.

Across all participants, the mean confirmation time was 917 ms (s.d. 459). In examining *confirmation time* across three quantiles (representing classifications as Q_1 fast, Q_2 medium, and Q_3 slow), the fastest quantile of participants had a mean confirmation time of 555 ms (s.d. 84 ms), and the slowest quantile had a mean of 1380 ms (s.d. 508 ms). There was also a significant positive correlation between *confirmation time* and user selected System C timeout ($r = 0.37, p = 4.7 \times 10^{-8}$). Binomial logistic regression also showed a significant relationship between *confirmation time* and preference choice ($p = .008$).

Finally, when participants were assigned to three quantiles according to *confirmation time*, the proportion of preferences for the fast interface (250 ms) across quantiles was similar to that for Study 1 (see Figure 4a) at 56% for Q_1 (fast), 59% for Q_2 (medium), and 41% for Q_3 (slow) participants.

Confirmation time therefore appears to be a promising metric for a system to automatically determine the user's pace – it is readily measured by the system, it correlates reasonably well with users' preferences and settings for preferred system pace, and the preference proportions across quantiles are similar to those produced when the user's pace is inferred from an analysis of Fitts's Law pointing performance.

4.3. Pace Entrainment

Although our research focus is on improving user preferences through automatic adaptation of the *system's* pace, a related issue concerns adaptation in the opposite direction – do users adapt their pace to that of the system? As reviewed in Section 2.2.2, previous HCI studies have indicated that user speech rates adapt towards that of animated agents [41], so perhaps other aspects of users' interaction pace, such as *confirmation time*, will also entrain towards the

system's pace. If users do entrain to the system's pace then this might influence the design of adaptive systems as well as the design of experiments used to evaluate them (both discussed later).

To examine whether and how the participants' changed their pace in response to the system's pace, we conducted a series of analyses.

First, we analysed *confirmation time* across the fast (250 ms) and slow (750 ms) conditions for systems A and B. Note that there is no requirement for users to alter *confirmation time* with the fast and slow interfaces – the 250/750 ms timeout only influences the hover time necessary to expand parent items, and it does not have any implemented effect on the final target item that is used to determine confirmation time.

The mean *confirmation time* when using the *fast* condition was 739 ms (s.d., 335), which was approximately 9% less than that with the *slow* condition (mean 814 ms, s.d., 384): $F_{1,208} = 24.5, p < 10^{-5}, \eta_g^2 = .105$. This difference indicates that users do indeed entrain to interface pace – subjects reacted more quickly to the cursor-over state when using the fast interface than when using the slow interface, even though there was no need for them to do so.

Second, we analysed how *confirmation time* changed across the series of twelve trials using the fast and slow interfaces. With the *slow* interface, there was a positive correlation between trial count (1-12) and *confirmation time* ($r = 0.53$), with mean values increasing from 768 ms in the first trial to 858 ms in the twelfth. Conversely, with the *fast* interface, the correlation was weakly negative ($r = -.043$), with no clear trend across trials: mean 745 ms in the first trial and 728 ms in the twelfth. The positive correlation for the *slow* interface might imply that users gradually shifted towards a slower pace of interaction when the pace requirements imposed by the interface are slower (i.e., the relatively sedate 750 ms timeout).

Third, we examined whether patterns of entrainment differed across different categories of user pace using a 2×3 ANOVA of the dependent variable *confirmation time* for independent variables *interface pace* ∈ {fast-ui (250 ms), slow-ui (750 ms)} and *user pace* ∈ {Q_1 fast, Q_2 medium, Q_3 slow}. Re-

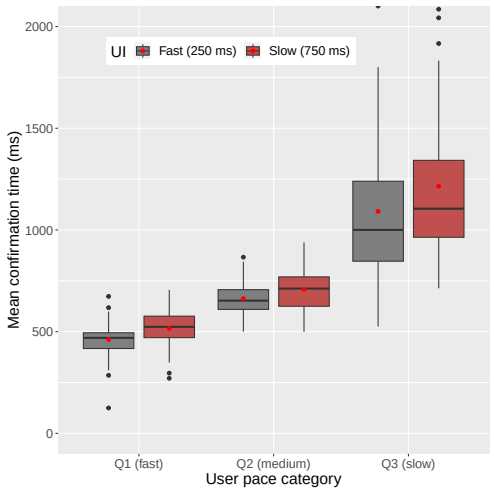


Figure 5: Entrainment analysis: confirmation time when using the fast (250 ms) and slow (750 ms) interface across user pace categories. In all box-whisker plots, red dots show the mean, black dots show outliers, the centre line shows the median, the hinges show first and third quartiles, and whiskers extend to max/min value within 1.5× the inter-quartile range from the hinge.

sults are summarised in Figure 5. As should be expected from the results reported above, this analysis showed a significant main effect of *interface pace* ($F_{1,206} = 24.8, p < 10^{-5}, \eta_g^2 = .027$) as well as a significant main effect of *user pace* ($F_{2,206} = 204.8, p < 10^{-5}, \eta_g^2 = .61$). However, there was no significant *interface pace*×*user pace* interaction ($F_{2,206} = 2.6, p = 0.07$), providing no reliable evidence that users in different categories of pace exhibit different patterns of entrainment.

4.4. Summary of pace characterisation

There are two main findings of Study 2's pace characterisation. First, *confirmation time* – the time between the cursor entering a target and a terminating release action on that object – appears to provide a good indication of user pace, correlating fairly well with subjects' preference choices for fast/slow interface conditions and with their preferred setting for system pace. Importantly, *confirmation time* is a value that any user interface can automatically measure

from basic user interface events. The upcoming studies reported in this paper use *confirmation time* as a basis for automatic system pace adaptation.

Second, analysis of *confirmation time* indicates that users entrain towards the system's pace. *Confirmation times* were shorter when using the fast user interface than when using the slow interface, even though the interface imposed no need for confirmation time to vary in these conditions. This observation supports the notion that users may "pick up the pace of a rapid interaction sequence" postulated by Shneiderman almost 40 years ago [48]. We return to the implications of this in the Discussion.

5. Study 3: Adapted Timeout versus a Baseline

Results from Study 1 indicate that preferences for system pace covary with measures of the user's pace, and results from Study 2 indicate that automatically measurable aspects of the user's pace (specifically, *confirmation time*) could be used as a parameter for automatically adapting the system's pace towards the user's pace.

Study 3 therefore examined whether users would prefer a system that automatically adapted its pace to a system that used a fixed baseline timeout. The adaptive timeout was set by continually measuring the participant's user pace (based on the mean *confirmation time* as described in Section 4.2) across the user's last three trials). The study was conducted twice, once with crowdsourced participants on mechanical turk (Study3_{mTurk}) and once with local students (Study3_{local}).

The hypothesis, H_2, for Study 3 was that the majority of fast and slow participants would prefer the *adapted interface* to the *baseline interface*.

5.1. Method

The experiment used a similar method to that of Study 1. Participants completed a set of *stage 1* trials, with the *confirmation time* data from the final three trials used to establish the initial timeout value for the *adaptive interface*. In *stage 2*, participants completed a series of trials using the baseline and

adaptive interfaces, which were presented as 'System A' and 'System B' (order counterbalanced), selecting their preferred interface after completing trials with System B (see Section 3.3.2). Finally, they completed *stage 3*, setting their preferred static timeout value for 'System C' (see Section 3.3.3).

For Study 3_{mTurk}, 83 participants were recruited on Amazon Mechanical Turk: 28 female, 55 male; mean age 40.3 years (s.d. 11.7); 76 reported using a mouse for input, 7 reported using a trackpad. For Study 3_{local}, 121 participants were recruited from an undergraduate computer science course, and carried out the study in a supervised computer lab. Participants' mean age was 21.8 years (s.d. 2.8); 23 female, 96 male, 2 other; 96 used a mouse for input, 25 used a trackpad.

After all participants had completed the study, our analysis classified each person into one of three quantiles (Q_1 fast, Q_2 medium, and Q_3 slow, separately for Study 3_{mTurk} and Study 3_{local}), and then assessed the proportion of participants in each quantile who preferred the adaptive or baseline interfaces. Participants were classified into quantiles based on their *mean adapted timeout*, which is the mean of the timeout value used by the adaptive interface across all of the *stage 2* trials for each user. Mean adapted timeout provides a more complete characterisation of the user's pace than the measures used in Study 1 because it covers all of each user's trials throughout the main part of the experiment (recall that in Study 1 participants were statically assigned to a fast or slow interface based on their pace using only stage 1 trials).

5.2. Study 3 results

Figure 6a summarises the main results for Study 3_{mTurk} (left) and Study 3_{local} (right), showing the proportion of participants who selected the *adaptive* and *baseline* interface as preferred across three user-pace quantiles, from Q_1 (fast) to Q_3 (slow). In Study 3_{mTurk}, the proportion preferring the adaptive interface was 43% in Q_1 (fast), 32% in Q_2 (medium) and 58% in Q_3 (slow); all showing no significant difference (Q_1 $\chi^2 = 0.3, p = .57$; Q_2 $\chi^2 = 2.4, p = .12$; Q_3 $\chi^2 = 0.3, p = .57$). The proportions preferring the adaptive interface were

higher in Study3_{local}, at 60% in Q_1, 63% in Q_2 and 41% in Q_3; again all showing no significant difference (Q_1 $\chi^2 = 1.2, p = .27$; Q_2 $\chi^2 = 2.0, p = .15$; Q_3 $\chi^2 = 1.6, p = .21$).

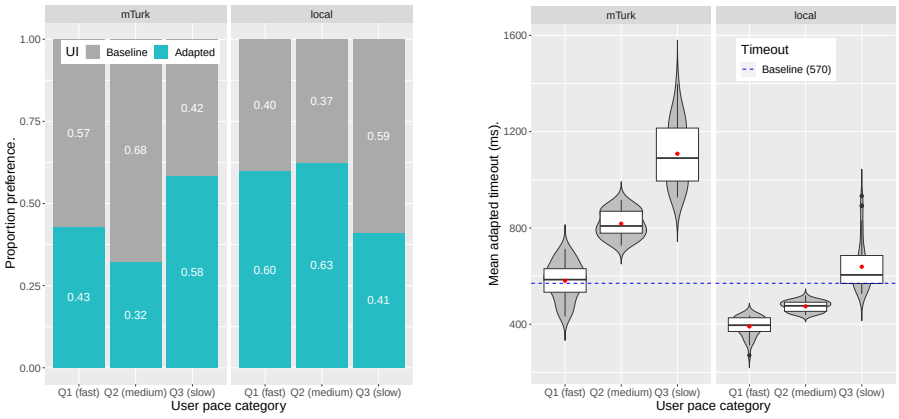
While these preference choice results do not provide support for H_2, the difference between the baseline timeout and the adapted timeout (which we term the 'delta' value) is an important factor in how much benefit is potentially provided by the adaptive interface. If the difference between the baseline and the adapted timeout is small, then there is little value provided by the adapted interface (and therefore little reason for users to prefer it). Figure 6b shows the distributions of *mean adapted timeout* values across quantiles in Study 3, and it also shows the baseline timeout. The figure highlights several important points, as follows.

First, the crowdsourced participants in Study3_{mTurk} had much longer *mean adapted timeout* values (mean 836 ms, s.d. 237) than the local participants in Study3_{1ocal} (mean 502 ms, s.d. 123 ms). In other words, the crowdsourced participants interacted much more slowly than the local participants – recall that *mean adapted timeout* values are derived from how quickly users release the mouse button to drop a dragged object after entering the target.

Second, as a consequence of their slow pace, the fastest quantile (Q_1) of crowdsourced participants received adaptive timeout values that were similar to the baseline (a mean difference of only 11 ms), and they therefore had no compelling reason to prefer adaptation. Participants in Q_3, however, received adaptive timeouts that were substantially different than the baseline (a mean difference of 538 ms), and they therefore had a much stronger rationale for preferring the adaptive interface, with a 58% majority doing so.

Third, the Q_1 (fast) and Q_2 (medium) local participants in Study 3_{local} had *mean adapted timeout* values that were notably faster than the 570 ms baseline value, providing a reason for them to prefer the adapted system (and 60% and 63% did so).

Fourth, the delta value does not adequately explain why a minority of Q_2 (medium) participants in Study 3_{mTurk} and Q_3 (slow) participants in Study 3_{local}



(a) Preference choice.

(b) Timeouts with the adaptive interface.

Figure 6: Results from Study 3 across three quantiles of user pace (Q_1 (fast) to Q_3 (slow)): interface preference choice (left) and distribution of adapted timeouts (right). In each plot, data from Study 3_{mTurk} (n=83) is left and Study 3_{local} (n=121) is right.

selected a preference for the adapted interface. In both of these conditions, the *mean adapted timeout* value was higher than the baseline timeout (providing a reason to prefer the slower pace of the adaptive interface), yet only 32% (Q_2 in Study 3_{mTurk}) and 41% (Q_3 in Study 3_{local}) did so. In these conditions, we believe that the negative experience of *inconsistency of the adapted timeout* was a stronger influence on the user's experience than any positive effects associated with the slower pace provided by the adaptive interface. These issues are further examined in Study 4.

Finally, regarding the much slower interaction by the crowdsourced participants in Study 3_{mTurk} than the local participants in Study 3_{local}, we see three possible explanations. First, the local participants were substantially younger (mean age 21.8 years) than the Turkers (40.3 years). Second, the local participants were likely more focused on the task than the Turkers (who may have been completing more than one task concurrently). These two issues are further discussed in Section 7.3. Third, entrainment effects may have also contributed to the difference: for example, if a Turker was slow in one trial (perhaps due to paying attention elsewhere), then the adaptive system would result in longer

timeout values for subsequent trials, and if the timeout was particularly long, that may have encouraged them to split their attention during trials, potentially leading to even longer timeout values. In contrast, an attentive participant who interacted quickly might be encouraged to 'pick up the pace of a rapid interaction sequence' [48], leading to a shorter adaptive timeout, encouraging yet faster interaction pace. Consequently, we were concerned that the results of Study 3 may have been influenced by the participants' susceptibility to entrain to the system's pace, rather than due to their preferences for a system that matched their own more 'natural' pace. We therefore adjusted the experimental method in Study 4 to reduce the likely influence of entrainment effects.

6. Study 4 – Adapted Timeout Versus a Baseline, with Mixed Tasks

In Studies 1 and 3, all of the *stage 2* tasks involved dragging an item through three hierarchical levels to reach a drop target (e.g., target item 11.5.5 as shown in Figure 2b). All trials in *stage 2* therefore exposed the user to the system pace, and therefore every trial had the potential to reinforce entrainment effects, with the users adapting their pace towards that of the system.

To mitigate this effect, Study 4 altered *stage 2* to contain additional drag and drop trials that did not involve exposure to the system pace. The studies used 36 drag-and-drop trials (rather than the 12 used previously), consisting of 12 repetitions of a three-trial pattern with two trials that did not use a hierarchy (e.g., 'drop in 16' as shown in Figure 2a) and one with hierarchy (e.g., 'drop in 11.5.5' as shown in Figure 2b). The non-hierarchical trials involved dropping the target onto a non-expanding item, and therefore they did not involve any system-imposed timeout. The additional trials allowed users to interact at their own pace for two trials before encountering the element of system-imposed pace in a third trial, with the goal of reducing artificial entrainment effects. Like Study 3, Study 4 was completed by two participant cohorts – crowdworkers in Study 4_{mTurk} and local students in Study 4_{local}. To account for the faster interaction by local participants than crowdworkers in Study 3, the baseline

timeout value used with local students was reduced to 500 ms. Other than these methodological variations, Study 4 was identical to Study 3.

Study 4_{mTurk} ran on Amazon Mechanical Turk, with 42 participants whose mean age was 40.7 years (s.d., 10.3); 18 female, 24 male; 32 mouse users, 10 trackpad. In Study 4_{local}, 116 participants were recruited from undergraduate classes at a local university (none of whom participated in the earlier experiment) – mean age 21.7 years (sd 3.2); 85 male, 25 female, 6 declined to answer; 83 reported using a mouse for input, 33 a trackpad.

6.1. Study 4 results

The main results, concerning the proportion of preference for the *adaptive* and *baseline* interfaces, are summarised in Figure 7a. In Study 4_{mTurk}, a 78% majority of Q_1 (fast) participants preferred the *adaptive interface* ($\chi^2 = 3.5, p = .06, h = .62$[5]), aligning with hypothesis H_2. However, in both Q_2 (medium) and Q_3 (slow), a 57% majority of participants preferred the *baseline interface* ($\chi^2 = .07, p = .79$ for both). And in Study 4_{local} a 69% majority of Q_1 (fast) participants again preferred the *adaptive interface* ($\chi^2 = 5.0, p = .025, h = 0.39$), but with only 55% preference among Q_2 participants ($\chi^2 = 0.24, p = .63$), and 36% preference among Q_3 participants ($\chi^2 = 2.6, p = .11$).

As in Study 3, the preference results are best understood with reference to the *mean adapted timeout* values and their difference from the baseline timeout values, as summarised in Figure 7b. The key observations are as follows.

First, as in Study 3, the crowdsourced participants were much slower than the local participants, with respective mean *mean adapted timeout* values of 585 ms (s.d. 227 ms) and 394 ms (s.d. 74 ms).

Second, the methodological change between Studies 3 and 4 (adding trials that did not involve the hierarchical timeout in Study 4) resulted in much faster interaction in Study 4 than observed in Study 3. This is evident in the values of *mean adapted timeout*, which reduced across studies from 836 ms to 585 ms for

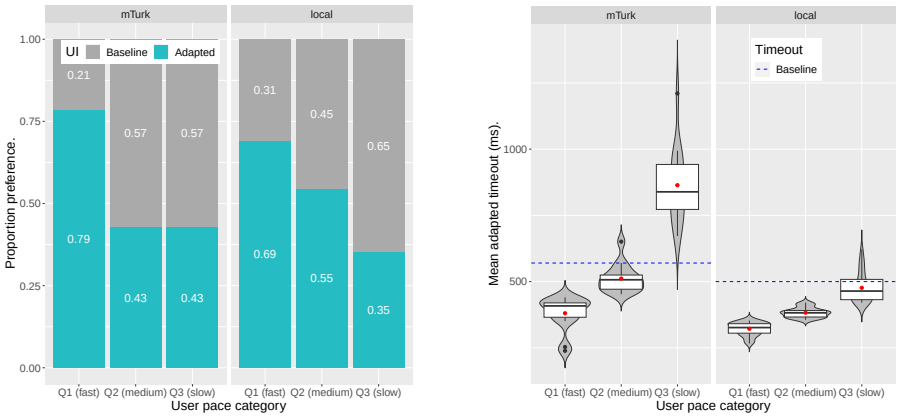
[5]Cohen's h effect size for proportions [9].

the crowdworkers, and from 502 ms to 394 ms for the local participants. These relatively large value changes suggest that entrainment effects can have a substantial influence on interaction users' pace, as further discussed in Section 7.4.

Third, the difference between the adapted timeout and the baseline, and the consistency of the timeout values, can again provide insight into the results. In both Study 4_{mTurk} and Study 4_{local}, the Q_1 (fast) participants' *mean adapted timeout* values were substantially lower than then baseline timeout value, providing a rational explanation for the 79% and 69% preference for the adapted interface – the Q_1 mean of *mean adapted timeout* in Study 4_{mTurk} was 380 ms, 190 ms lower than the baseline of 570 ms; in Study 4_{local} the mean was 322 ms, 178 ms lower than the baseline of 500 ms. While the preference choices of Q_1 (fast) participants align with H_2, the choices of Q_3 (slow) participants do not, with a 57% and 65% majority choosing the baseline interface as preferred. In Study 4_{local} this preference for the baseline interface can be explained by two factors – the mean *mean adapted timeout* value (at 477 ms) was very similar to the 500 ms baseline, providing at most a minimal reason for preferring the adaptive interface; and as shown in Figure 8, the adapted timeout values for Q_3 participants were much more widely distributed than those for Q_1 participants, indicating that these Q_3 received relatively inconsistent adaptive timeouts.

7. Discussion

To summarise the findings, Study 1 showed that fast-paced users preferred a fast system pace to a slow one, and that slow-paced users preferred slow to fast. Study 2 further examined the Study 1 data, with two main findings: users adapt their pace of interaction towards that of the system, even when there is no need to do so; and measures of *confirmation time* (the time between the cursor entering a target item and releasing the mouse button to select it) correlate with preferences for system pace. Studies 3-4 then examined whether users would prefer a system that *dynamically* adapted its pace towards that of the user, to a system that used a baseline pace. In general, the findings of Studies 3-4



(a) Preference choice.

(b) Timeouts with the adaptive interface.

Figure 7: Results from Study 4 across three quantiles of user pace (Q_1 (fast) to Q_3 (slow)): interface preference choice (left) and distribution of adapted timeouts (right). In each plot, data from Study 4_{mTurk} is left and Study 3_{local} is right.

showed that the majority of fast users preferred the faster interaction enabled by adaptive pace; however, results were less clear for slow-paced users. The studies also identified two underlying issues that can help to explain the results: the size of the difference between user pace and adapted pace (with larger differences providing more reason to prefer adaptation), and the variability in the sequence of adapted timeouts (with higher variance causing an inconsistent interactive experience, reducing preference for the adaptation). Table 1 provides a summary of numerical values associated with Studies 3 and 4.

		Baseline	**Prefer Adapted**			**Adapted T**		**SystemC T**	
Study	**n**	T	Q_1	Q_2	Q_3	$\bar{x}$	σ	$\bar{x}$	σ
3_{mTurk}	83	570 ms	43%	32%	58%	836 ms	237	679 ms	385
3_{local}	121	570 ms	60%	63%	41%	502 ms	123	372 ms	203
4_{mTurk}	42	570 ms	78%	43%	43%	585 ms	227	563 ms	367
4_{local}	116	500 ms	69%	55%	45%	394 ms	74	372 ms	178

Table 1: Summary of values across Studies 3 and 4. T for 'timeout'. SystemC T values are the user-selected preferred values for sytem timeout.

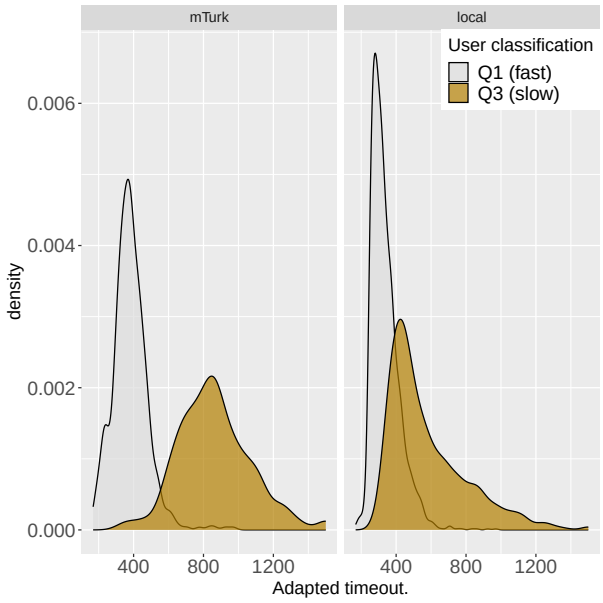


Figure 8: Distribution of adapted timeout values that Q_1 (fast) and Q_3 (slow) participants received in Study 4_{mTurk} (left) and Study 4_{local} (right). Timeout values were more variable for Q_3 (slow) participants than for Q_1 (fast).

The following subsections provide a pooled data analysis of the results from Studies 3-4, and discuss the full set of results, including possible explanations for the findings, the implications for design, and study limitations and opportunities for further work.

7.1. Pooled data analysis of Studies 3-4

We followed up on the idea of difference between user pace and adapted pace by carrying out a pooled data analysis across Studies 3-4. Hypothetically, user preferences for an ideal adaptive pace system should follow a U-shape across the spectrum of user pace, as depicted in Figure 9a. If the baseline value is well chosen, then around the median user pace, preferences should be roughly evenly split between the adapted interface and the baseline (we show a slight preference for the baseline interface at the mid-point because it is reasonable to suspect that the pace inconsistency caused by adaptation will be less desirable

to these users than the consistency of an unchanging baseline.) The further the users' pace lies from the mid-point, however, the greater the proportion of users who should prefer adaptation – very fast or very slow users should receive an adapted pace that is substantially different from the baseline, and better suited to their needs.

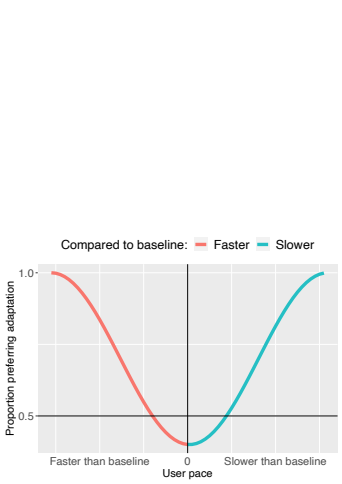
To examine conformance with this hypothetical trend, we pooled data from Studies 3-4, determining a *delta* value for each participant that is the difference between their *mean adapted timeout* and the *baseline timeout*. We then analysed the Spearman rank correlation between the *delta* values and the binary preference choice for the adapted interface, conducting this analysis separately for participants with negative delta (i.e., faster than the baseline) and positive delta (slower than the baseline).

Figure 9b summarises the results. As hypothesised, for fast participants (negative *delta*), there was a significant negative correlation between *delta* and preference for the adapted interface (Spearman $\rho = -.18, p = .005$) – preference for the adapted interface increased with the magnitude of *delta*, with the linear model indicating ≈80% preference for the adaptive interface for the very fastest participants (*delta*=-300) down to ≈40% preference at *delta*=0.

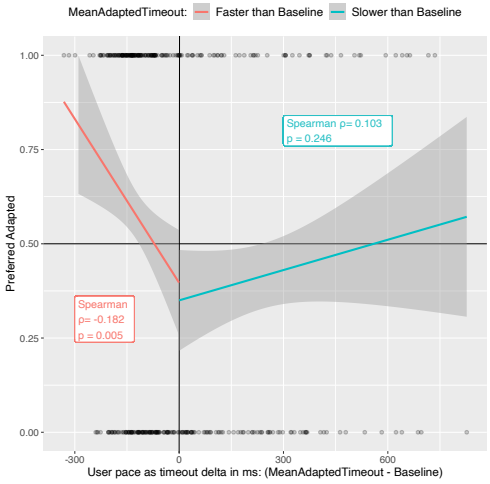
For slow participants (positive *delta*) there was a non-significant positive correlation (Spearman $\rho = 0.1, p = .25$), with low levels of acceptance for the adapted interface, ranging from ≈35% at *delta*=0 to ≈56% for the very slowest users.

These results support the general findings reported above – while automatic adaptation can improve the preferences of fast users, there is a lack of evidence that it can succeed for slower users.

This pooled data analysis gives a wide spectrum of values of *mean adapted timeout*. This analysis was not possible for the individual studies because the proportion of participants with *mean adapted timeout* values above/below the baseline timeout varied substantially across studies (see Figures 6b and 7b) – for example, in Study 3_{mTurk} few participants had a *mean adapted timeout* value below that of the baseline, and in Study 4_{local} few participants had a *mean*



(a) Hypothetical curve showing the proportion of users preferring an interface that adapts to their pace over a well chosen static baseline.



(b) Pooled data analysis of Studies 3-4 including linear correlations for participants faster than the baseline (left) and slower (right), including 95% confidence interval.

Figure 9: The proportion of participants who prefer the adapted interface to a static baseline interface across user pace (fast to slow). Hypothetical U-shaped curve (left), and empirical results (right).

adapted timeout above that of the baseline.

7.2. What caused the observed effects?

7.2.1. Why did fast users prefer the adaptive interface?

In Studies 1, 3_{local}, 4_{mTurk}, and 4_{local}, the majority of fast-paced participants preferred the faster pace provided by the adaptive interface: 58% in Study 1, 60% in Study 3_{local}, 79% in Study 4_{mTurk}, and 69% in Study 4_{local}. In these studies, at least three factors may have contributed to the 'fast' participants selecting the adaptive interface as preferred to the baseline interface. First, as hypothesised, users who interact quickly may appreciate the faster interaction enabled by the system. Although we have not highlighted participant comments for Studies 3-4, many 'fast' participants made comments similar to those reported for Study 1 (see Section 3.6.5), emphasising their satisfaction with the adaptive system's quick interaction. While further work is needed to

better understand exactly why the fast participants preferred fast interaction, we suspect that the ability to complete tasks at a fast rate (i.e., high task throughput with the interface) was a key factor.

Second, fast participants may have been dissatisfied with the slower interaction provided by the baseline. This possibility raises experimental concerns related to the choice of the baseline value. In Studies 3_{local}, 4_{mTurk}, the baseline was set at 570 ms, based on the mean value for System C selected as preferred by participants in Study 1. In Study 4_{local}, we reduced the baseline value to 500 ms because the local participants in Study 3_{local} interacted much more quickly than the crowd-workers in Studies 1 and 3_{mTurk} (the choice of 500 ms was based on the mean value of *mean adapted timeout* in Study 3_{local}). As discussed in Section 3.2, the choice of timeout value (for the baseline interface in this case) is an important experimental consideration – if it is too high then a flooring effect is likely, with most participants preferring a faster interface regardless of their pace, and the inverse if it is too slow. While it is likely that a higher baseline value would result in a higher proportion of 'fast' participants choosing the adaptive interface as preferred, it is also likely that it would reduce the number of 'slow' participants preferring adaptation. Further experimental work is needed to examine these effects, but the baseline values used in our studies are not dissimilar to the default values used in contemporary software (for example, the 600 ms value used in the Mac OS Finder's column view; see Section 2.2.1). Furthermore, the pooled data analysis reported in Section 7.1 reduces some of the potential impact of the baseline value by examining results with respect to the timeout *delta* from the baseline. The difficulty we had in setting a baseline value that worked across multiple studies is also an indication of how difficult it will be for designers to choose a reasonable one-size-fits-all value that will work well for all users – providing additional rationale for the adaptation approach.

Third, it is possible that fast participants may have appreciated the system's adaptation towards their pace, liking the fact that the system 'cottoned on' to their desire for faster interaction. While this explanation remains a possibility,

we did not see participant comments directly referring to it, and we suspect it was not a major influence in their preference choice. There are interesting opportunities for further study into the subjective experience of observing that the system is trying to adapt to better fit the user's needs. Further research is needed to better understand the role of each of these three potential explanations on user preferences.

In Study 3_{mTurk}, however, only 43% of participants classified as 'fast' preferred the fast interface. We believe that this too can be attributed to the users' pace with respect to the baseline value – as summarised in Figure 6b, the mean timeout value provided to the crowdworkers was 581 ms and therefore very similar to the baseline value (at 570 ms), providing little reason to select a preference for the adaptive interface.

7.2.2. Why did adaptation fail for slow users?

While a 61% majority of slow-paced participants preferred the slow interface to the fast interface in Study 1, in all of the adapted-pace studies (except Study 3_{mTurk}) the majority of slow-paced participants preferred the baseline interface to the adaptive interface: 59% in Study 3_{local}, 57% in Study 4_{mTurk} and 65% in Study 4_{local}. In contrast, in Study 3_{mTurk}, 58% of slow participants preferred the adaptive interface.

Comparing data distributions between Q_3 (slow) and Q_1 (fast) participants provides insights into why slow participants may not appreciate the adaptive interfaces. In particular, data from Q_3 participants is much more variable than that for Q_1 participants (see the distributions in Figure 8). This variability in *mean adapted timeout* for Q_3 participants will have caused inconsistent timeout values in the adaptive interface, potentially explaining a preference for the more consistent baseline interface. Furthermore, users who interact more slowly may have difficulty building confidence with a system, and adaptations (such as changing timeouts) may exacerbate these difficulties. These observations support the findings of Yu [55], who noted that a predictable interaction rhythm is beneficial during interaction with intelligent systems. Our results suggest that

this finding may be especially true for users who interact more slowly.

Various methods could be used to reduce the instability of the adaptive timeouts when users have higher variance in their interaction. We implemented two controls in Study 3 and 4 (a limit on change and an overall cap), but more could be done to improve the experience for slow users. For example, the one-Euro filter developed by Casiez and colleagues has been shown to provide a reasonable balance between responsiveness and stability for noisy human input streams [3]. Further work is necessary, however, to compare different hysteresis functions and validate their effectiveness. At present, however, our studies indicate that there is a lack of evidence that automatic pace adaptation can succeed for slow users.

7.3. Why did the crowd-workers differ from the local students?

Studies 3 and 4 both used two participant cohorts – crowd-workers and local students, with the crowd-workers interacting much slower than the local students, and they therefore received much higher mean values for *mean adapted timeout*: in Study 3 , 836 ms for the crowd-workers versus 502 ms for the local students; and in Study 4 585 ms for the crowd-workers versus 394 ms for the local students.

We suspect that the main reason for the crowd-workers' slower performance and preferences was due to their age. The mean age of the crowd-worker participants was nearly two decades higher than than that of the local students (40.5 years versus 21.8 years). Prior studies have demonstrated that age-related cognitive-motor decline in reaction times begins from around 24 years of age [49], with decline values estimated at 2.8 ms/year [54] for cognitive reaction time alone. And a study of pointing performance [23] found no difference between 'young' (aged 12-14) and 'adult' participants (aged 25-33) , while 'elderly' participants (aged 61-69) were significantly slower (means of 1587 ms and 2175 ms for adults and elderly respectively).

However, we also suspect that a high proportion of crowd-workers had low engagement with the study, possibly due to their trying to complete multi-

ple studies concurrently to maximise their income on the platform [43]. One indication supporting this inference is that a surprisingly large number of crowd-worker participants (79 of 548, 14%) gave a one-word response of 'good,' 'nothing,' 'none' or 'nice' in the free-form comments at the end of the experiment; in contrast, none of the 237 local students provided a one-word comment. Problems with the veracity of data derived from crowd-sourced platforms, particularly on Amazon Mechanical Turk, are discussed in Peer et al. [43].

7.4. The influence of entrainment

The main experimental tasks of *stage 2* in Study 3 *all* involved dragging an object through a series of three hierarchical levels. In Study 4 we modified the method to introduce two non-hierarchical drag-and-drop tasks before each hierarchical task. We did so to ease concerns that users may have been excessively entraining to the pace of interaction imposed by the expansion timeout. By introducing the non-hierarchical drag-and-drop tasks, users would be free to interact at their preferred rate (for the non-hierarchical tasks) without needing to wait for the timeout to expire.

We were surprised by how much this methodological adjustment influenced the participants' performance. For the crowd-workers in Study 3_{mTurk}, the mean value of *mean adapted timeout* was 836 ms, which reduced to 585 ms in Study 4_{mTurk} (a 32% reduction); and for the local students, the value reduced from 502 ms to 394 ms (a 22% reduction).

There are three high-level implications of this observation. First, users appear to be strongly influenced by system pace, and they are likely to quickly adapt their pace towards that of the system. Second, reflecting the findings of Yu [55], consistent system pace creates a predictable pace that the user can entrain towards; but the high variability of performance by slow users can impair pace consistency in adaptive systems. Third, researchers conducting experiments relating to system pace should think carefully about the nuances of their method, including the desirability of amplifying or suppressing entrainment effects.

7.5. Study limitations, design implications, and further work

Many forms of interaction involve elements of system pace, including speech interaction, animations, control-display gain functions, as well as the use of timeouts to disambiguate user intentions (e.g., the use of long-press and very-long-press on mobile touchscreen devices). All of our studies were conducted within one limited interaction – the duration of the hover timeout used to determine when to expand a hierarchical item during drag-and-drop. While we hope that our findings generalise to interactions beyond hierarchical drag-and-drop, further studies are necessary. In particular, studies are needed to to replicate our findings, to broaden timeout-based findings to other types of timeouts (such as those used to discriminate gestures), and to generalise to other types of pace-based interactions.

Within our specific domain of hierarchical drag-and-drop, the results have important implications for a basic interaction that is used in an extremely wide range of applications, including file browsers (such as Finder and Windows Explorer), email clients, code navigators, web-browser bookmarks, mobile phone homescreens, photo browsers, and many more. While some applications or operating systems provide explicit customisation controls to configure the expansion timeout, research suggests that such customisation features are seldom used [33, 27, 34]. Furthermore, the default values used for the expansion timeout varies substantially between applications (see Section 2.2.1), suggesting that some users will be poorly served by the defaults – too fast for some, too slow for others. Another avenue for further work within drag-and-drop interactions concerns iterations and improvements of the adaptive algorithm: while our study used *confirmation time* and a small amount of hysteresis to smooth adaptive changes, other measures of user pace and other adaptive algorithms may further improve user experience. Longitudinal studies could also examine how users' pace changes over time, how preferences change over time, and how systems can best adapt to long-term aspects of user pace.

Our results indicate that systems can automatically identify fast users who are likely to benefit from a timeout value that is shorter than the default value

(by inspecting *confirmation time*), and that by reducing the timeout value for these participants their subjective experience can be improved. The results also indicate that for the remaining medium- and slow-paced users, the system should leave the baseline timeout value unaltered. However, further work is needed to better understand how the best possible 'one size fits all' baseline value can be established for use in any experiment examining automatic pace adaptation. For example, our studies showed that the crowd-workers interacted much more slowly than our lab participants, possibly due to their older age.

8. Conclusion

We conducted a series of four studies to examine whether users would benefit from a system pace that better matched their own pace of interaction. All of the studies were based on drag-and-drop interactions, where a timeout is used to determine when to expand a hierarchical item. The first study showed that fast users prefer a fast timeout to a slow one, and the inverse for slow users. The second study showed that the time between the cursor entering a target and releasing the mouse button for selection provides a reliable measure of the users' pace, and that users quickly adapt their pace towards the system's pace even when there is no need for them to do so. The third and fourth studies examined whether users would prefer a system that adapts its pace toward that of the user over a system that uses a 'one size fits all' timeout, with results showing that adaptation can improve preferences for fast users, but not for medium- or slow-paced users. This finding suggests that automatic timeout adaptation could improve the preferences of a large and important group of users – fast, high-performance users.

Acknowledgement

This research was supported by Royal Society of New Zealand Marsden grant M1218.

References

[1] James J. Bradac, Anthony Mulac, and Ann House. 1988. Lexical diversity and magnitude of convergent versus divergent style shifting-: Perceptual and evaluative consequences. *Language and Communication* 8, 3 (1988), 213–228.

[2] Stuart K. Card, Thomas P. Moran, and Allen Newell. 1983. *The psychology of human-computer interaction*. L. Erlbaum Associates, Hillsdale, N.J.

[3] Géry Casiez, Nicolas Roussel, and Daniel Vogel. 2012. 1€ filter: a simple speed-based low-pass filter for noisy input in interactive systems. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2527–2530.

[4] Géry Casiez, Daniel Vogel, Ravin Balakrishnan, and Andy Cockburn. 2008. The Impact of Control-Display Gain on User Performance in Pointing Tasks. *Human-Computer Interaction* 23, 3 (2008), 215–250.

[5] Olivier Chapuis, Renaud Blanch, and Michel Beaudouin-Lafon. 2007. *Fitts' Law in the Wild: A Field Study of Aimed Movements*. Technical Report. `https://hal.archives-ouvertes.fr/hal-00612026` LRI Technical Report Number 1480, Univ. Paris-Sud, 11 pages.

[6] Tanya L. Chartrand and John A. Bargh. 1999. The Chameleon Effect: The Perception-Behavior Link and Social Interaction. *Journal of Personality and Social Psychology* 76, 6 (1999), 893–910.

[7] Sin-Horng Chen, Chiao-Hua Hsieh, Chen-Yu Chiang, Hsi-Chun Hsiao, Yih-Ru Wang, Yuan-Fu Liao, and Hsiu-Min Yu. 2014. Modeling of Speaking Rate Influences on Mandarin Speech Prosody and Its Application to Speaking Rate-Controlled TTS. *IEEE/ACM Trans. Audio, Speech and Lang. Proc.* 22, 7 (July 2014), 1158–1171. `https://doi.org/10.1109/TASLP.2014.2321482`

[8] Fanny Chevalier, Nathalie Henry Riche, Catherine Plaisant, Amira Chalbi, and Christophe Hurter. 2016. Animations 25 Years Later: New Roles and Opportunities. In *Proceedings of the International Working Conference on Advanced Visual Interfaces* (Bari, Italy) *(AVI '16)*. Association for Computing Machinery, New York, NY, USA, 280–287. https://doi.org/10.1145/2909132.2909255

[9] Jacob Cohen. 1988. *Statistical power analysis for the behavioral sciences* (2nd ed.). L. Erlbaum Associates, Hillsdale, N.J.

[10] Thomas J Conrad and Yin Yin Wong. 2000. Computer system with graphical user interface including spring-loaded enclosures. https://patents.google.com/patent/US6061061 Patent No. US6061061, Filed Jul. 8, 1997, Issued May 9, 2000.

[11] Oscar de Bruijn and Robert Spence. 2000. Rapid Serial Visual Presentation: A Space-Time Trade-off in Information Presentation. In *Proceedings of the Working Conference on Advanced Visual Interfaces* (Palermo, Italy) *(AVI '00)*. Association for Computing Machinery, New York, NY, USA, 189–192. https://doi.org/10.1145/345513.345309

[12] Alena Denisova and Paul Cairns. 2015. Adaptation in Digital Games: The Effect of Challenge Adjustment on Player Performance and Experience. In *Proceedings of the 2015 Annual Symposium on Computer-Human Interaction in Play* (London, United Kingdom) *(CHI PLAY '15)*. Association for Computing Machinery, New York, NY, USA, 97–101. https://doi.org/10.1145/2793107.2793141

[13] AJ Dix. 1992. Pace and interaction. In *Proceedings of HCI '92: People and computers VII*. 171–190.

[14] Kohji Dohsaka, Atsushi Kanemoto, Ryuichiro Higashinaka, Yasuhiro Minami, and Eisaku Maeda. 2010. User-Adaptive Coordination of Agent Communicative Behavior in Spoken Dialogue. In *Proceedings of the 11th Annual*

Meeting of the Special Interest Group on Discourse and Dialogue (Tokyo, Japan) *(SIGDIAL '10)*. Association for Computational Linguistics, USA, 314–321.

[15] Greg T. Elliott and Bill Tomlinson. 2006. PersonalSoundtrack: Context-Aware Playlists That Adapt to User Pace. In *CHI '06 Extended Abstracts on Human Factors in Computing Systems* (Montréal, Québec, Canada) *(CHI EA '06)*. Association for Computing Machinery, New York, NY, USA, 736–741. https://doi.org/10.1145/1125451.1125599

[16] PM Fitts. 1954. The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement. 47 (1954), 381–391.

[17] T. Francis. 1996. *Mac OS 8 Revealed*. Addison-Wesley. https://books.google.co.nz/books?id=xELQOQAACAAJ

[18] Louis David Giacolone Jr. 1998. Dynamic rate control method and apparatus for electronically played games and gaming machines. https://patents.google.com/patent/US5758875 Patent No. US5758875A, Filed Jan. 11th, 1996, Issued Jun. 2nd., 1998.

[19] Howard Giles, Anthony Mulac, James J. Bradac, and Patricia Johnson. 1987. Speech Accommodation Theory: The First Decade and Beyond. *Annals of the International Communication Association* 10, 1 (1987), 13–48. https://doi.org/10.1080/23808985.1987.11678638 arXiv:https://doi.org/10.1080/23808985.1987.11678638

[20] Alix Goguey, Carl Gutwin, Zhe Chen, Pang Suwanaposee, and Andy Cockburn. 2021. Interaction Pace and User Preferences. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 195, 14 pages. https://doi.org/10.1145/3411764.3445772

[21] Chris Harrison, Brian Amento, Stacey Kuznetsov, and Robert Bell. 2007. Rethinking the Progress Bar. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (Newport, Rhode Island, USA) *(UIST '07)*. ACM, New York, NY, USA, 115–118. `https://doi.org/10.1145/1294211.1294231`

[22] Andrew Heathcote, Stephen J. Popiel, and D. J. K. Mewhort. 1991. Analysis of Response Time Distributions: An Example Using the Stroop Task. *Psychological bulletin* 109, 2 (1991), 340–347.

[23] Morten Hertzum and Kasper Hornbæk. 2010. How Age Affects Pointing With Mouse and Touchpad: A Comparison of Young, Adult, and Elderly Users. *International journal of human-computer interaction* 26, 7 (2010), 703–734.

[24] William E. Hockley. 1984. Analysis of response time distributions in the study of cognitive processes. *Journal of experimental psychology. Learning, memory, and cognition* 10, 4 (1984), 598–615.

[25] Scott E. Hudson and John T. Stasko. 1993. Animation Support in a User Interface Toolkit: Flexible, Robust, and Reusable Abstractions. In *Proceedings of the 6th Annual ACM Symposium on User Interface Software and Technology* (Atlanta, Georgia, USA) *(UIST '93)*. Association for Computing Machinery, New York, NY, USA, 57–67. `https://doi.org/10.1145/168642.168648`

[26] Matt Jones, Preeti Jain, George Buchanan, and Gary Marsden. 2003. Using a mobile device to vary the pace of search. *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)* 2795 (2003), 390–394.

[27] Wiard Jorritsma, Fokie Cnossen, and Peter M. A. van Ooijen. 2015. Adaptive support for user interface customization: a study in radiology. *International Journal of Human - Computer Studies* 77 (2015), 1–9.

[28] Melissa Jungers, Caroline Palmer, and Shari Speer. 2002. Time after time: The coordinating influence of tempo in music and speech. *Cognitive Processing* 1 (01 2002), 21–35.

[29] Rivk Levitan, Štefan Beňuš, Agustín Gravano, and Juli Hirschberg. 2015. Entrainment and turn-taking in human-human dialogue, Vol. SS-15-07. 44–51.

[30] Rivka Levitan and Julia Hirschberg. 2011. Measuring acoustic-prosodic entrainment with respect to multiple levels and dimensions. In *Proceedings of the Annual Conference of the International Speech Communication Association, INTERSPEECH 2011*. 3081–3084. `http://www.cs.columbia.edu/~rlevitan/papers/IS11full_paper.pdf`

[31] Natalie Lewandowski and Matthias Jilka. 2019. Phonetic Convergence, Language Talent, Personality and Attention. *Frontiers in Communication* 4 (2019), 18. `https://doi.org/10.3389/fcomm.2019.00018`

[32] Nichola Lubold. 2017. Building Rapport through Dynamic Models of Acoustic-Prosodic Entrainment. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems* (Denver, Colorado, USA) *(CHI EA '17)*. Association for Computing Machinery, New York, NY, USA, 297–300. `https://doi.org/10.1145/3027063.3027132`

[33] Wendy E. Mackay. 1991. Triggers and Barriers to Customizing Software. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (New Orleans, Louisiana, USA) *(CHI '91)*. Association for Computing Machinery, New York, NY, USA, 153–160. `https://doi.org/10.1145/108844.108867`

[34] Sylvain Malacria, Alix Goguey, Gilles Bailly, and Géry Casiez. 2016. Multi-Touch Trackpads in the Wild. In *Actes de La 28ième Conference Francophone Sur l'Interaction Homme-Machine* (Fribourg, Switzerland) *(IHM '16)*. Association for Computing Machinery, New York, NY, USA, 19–24. `https://doi.org/10.1145/3004107.3004113`

[35] Joseph H. Manson, Gregory A. Bryant, Matthew M. Gervais, and Michelle A. Kline. 2013. Convergence of speech rate in conversation predicts cooperation. *Evolution and Human Behavior* 34, 6 (2013), 419–426.

[36] Mudit Misra, Elena árquez Segura, and Ahmed Sabbir Arif. 2019. Exploring the Pace of an Endless Runner Game in Stationary and Mobile Settings. In *Extended Abstracts of the Annual Symposium on Computer-Human Interaction in Play Companion Extended Abstracts* (Barcelona, Spain) *(CHI PLAY '19 Extended Abstracts)*. Association for Computing Machinery, New York, NY, USA, 543–550. https://doi.org/10.1145/3341215.3356256

[37] Kevin R. Murphy and Brett Myors. 2004. *Statistical power analysis: a simple and general model for traditional and modern hypothesis tests* (2nd ed.). L. Erlbaum Associates, Publishers, Mahwah, N.J.

[38] Jakob Nielsen. 1993. *Usability engineering*. Academic Press, Boston.

[39] J Nielsen. 1993. *Usability Engineering*. London: Academic Press.

[40] William Odom, Richard Banks, Abigail Durrant, David Kirk, and James Pierce. 2012. Slow Technology: Critical Reflection and Future Directions. In *Proceedings of the Designing Interactive Systems Conference* (Newcastle Upon Tyne, United Kingdom) *(DIS '12)*. Association for Computing Machinery, New York, NY, USA, 816–817. https://doi.org/10.1145/2317956.2318088

[41] Sharon Oviatt, Courtney Darves, and Rachel Coulston. 2004. Toward Adaptive Conversational Interfaces: Modeling Speech Convergence with Animated Personas. *ACM Trans. Comput.-Hum. Interact.* 11, 3 (Sept. 2004), 300–328. https://doi.org/10.1145/1017494.1017498

[42] Jennifer S. Pardo, Jennifer S. Pardo, Adelya Urmanche, Adelya Urmanche, Sherilyn Wilman, Sherilyn Wilman, Jaclyn Wiener, and Jaclyn Wiener. 2017. Phonetic convergence across multiple measures and model talkers. *Attention, Perception, & Psychophysics* 79, 2 (2017), 637–659.

[43] Eyal Peer, Laura Brandimarte, Sonam Samat, and Alessandro Acquisti. 2017. Beyond the Turk: Alternative platforms for crowdsourcing behavioral research. *Journal of experimental social psychology* 70 (2017), 153–163.

[44] Martin J. Pickering and Simon Garrod. 2004. Toward a mechanistic psychology of dialogue. *The Behavioral and brain sciences* 27, 2 (2004), 169–190.

[45] Philip Quinn, Sylvain Malacria, and Andy Cockburn. 2013. Touch Scrolling Transfer Functions. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology* (St. Andrews, Scotland, United Kingdom) *(UIST '13)*. Association for Computing Machinery, New York, NY, USA, 61–70. https://doi.org/10.1145/2501988.2501995

[46] Jeff Sauro and James R. Lewis. 2010. Average Task Times in Usability Tests: What to Report?. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Atlanta, Georgia, USA) *(CHI '10)*. Association for Computing Machinery, New York, NY, USA, 2347–2350. https://doi.org/10.1145/1753326.1753679

[47] Benjamin G. Schultz, Irena O'Brien, Natalie Phillips, David H. McFarland, Debra Titone, and Caroline Palmer. 2016. Speech rates converge in scripted turn-taking conversations. *Applied Psycholinguistics* 37, 5 (2016), 1201–1220.

[48] Ben Shneiderman. 1984. Response time and display rate in human performance with computers. *ACM Computing Surveys (CSUR)* 16, 3 (1984), 265–285.

[49] Joseph J. Thompson, Mark R. Blair, and Andrew J. Henrey. 2014. Over the hill at 24: persistent age-related cognitive-motor decline in reaction times in an ecologically valid video game task begins in early adulthood. *PloS one* 9, 4 (2014), e94215–e94215.

[50] Rick B. van Baaren, Rob W. Holland, Bregje Steenaert, and Ad van Knippenberg. 2003. Mimicry for money: Behavioral consequences of imitation. *Journal of experimental social psychology* 39, 4 (2003), 393–398.

[51] Rodrigo Vicencio-Moreira, Regan L. Mandryk, and Carl Gutwin. 2015. Now You Can Compete With Anyone: Balancing Players of Different Skill Levels in a First-Person Shooter Game. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 2255–2264. https://doi.org/10.1145/2702123.2702242

[52] Kent Wittenburg, Clifton Forlines, Tom Lanning, Alan Esenther, Shigeo Harada, and Taizo Miyachi. 2003. Rapid Serial Visual Presentation Techniques for Consumer Digital Video Devices. In *Proceedings of the 16th Annual ACM Symposium on User Interface Software and Technology* (Vancouver, Canada) *(UIST '03)*. Association for Computing Machinery, New York, NY, USA, 115–124. https://doi.org/10.1145/964696.964709

[53] Jacob O. Wobbrock, Edward Cutrell, Susumu Harada, and I. Scott MacKenzie. 2008. An error model for pointing based on Fitts' law. In *CHI '08: Proceeding of the twenty-sixth annual SIGCHI conference on Human factors in computing systems* (Florence, Italy). ACM, New York, NY, USA, 1613–1622. https://doi.org/10.1145/1357054.1357306

[54] David L. Woods, John M. Wyma, E. William Yund, Timothy J. Herron, and Bruce Reed. 2015. Age-related slowing of response selection and production in a visual choice reaction time task. *Frontiers in Human Neuroscience* 9 (2015). https://doi.org/10.3389/fnhum.2015.00193

[55] Christine Guo Yu. 2019. *Effects of timing on users' perceived control when interacting with intelligent systems*. Technical Report UCAM-CL-TR-939. University of Cambridge, Computer Laboratory. https://doi.org/10.48456/tr-939