OptiDwell: Intelligent Adjustment of Dwell Click Time

Aanand Nayyar
Dayalbagh Educational Institute
Agra
aanandnayyar@ieee.org

Utkarsh Dwivedi
IBM Research India
New Delhi
utkdwive@in.ibm.com

Karan Ahuja
Indian Institute of Information Technology
Guwahati
karan.ahuja@iiitg.ac.in

Nitendra Rajput
IBM Research India
New Delhi
rnitendra@in.ibm.com

Seema Nagar
IBM Research India
Bangalore
senagar3@in.ibm.com

Kuntal Dey
IBM Research India
New Delhi
kuntadey@in.ibm.com

ABSTRACT

Gaze based navigation with digital screens offer a hands-free and touchless interaction, which is often useful in providing a hygienic interaction experience in a public kiosk scenario. The goodness of such a navigation system depends not only on the accuracy of detecting the eye gaze but also on the ability to determine whether a user is interested in clicking a button or is just looking at the button. The time for which a user needs to gaze at a particular button before it is considered as a click action is called the dwell time. In this paper, we explore intelligent adjustment of dwell times, where mouse click events on the buttons of a given application are emulated with user gaze. A constant dwell-time for all buttons and for all users may not provide an efficient and intuitive interface. We thereby propose a model to dynamically adjust dwell-time values used to emulate user mouse click events, exploiting the user's experience with different portions of a given application. The adjustment happens at a per-user, per-button granularity, as a function of the user's (a) prior usage experience of the given button within the application and (b) Midas touch characteristics for the given button.

We propose *OptiDwell*, inspired by the action-value method based solutions to the Multi-Armed Bandits problem, for dwell click time adaptation. We experiment *OptiDwell* using an interactive TV channel browsing interface application, constituting of a mix of text and image buttons, over 10 computer-savvy users generating over 9000 click tasks. We observe significant improvement of user comfort level over the sessions, quantified by (a) improved (reduced) dwell times and (b) reduced number of Midas touches in spite of faster dwell-clicks, as high as 10-fold reduction in the best case. Our work is useful for creating an interface, with accurate, fast and comfortable dwell-clicks for each interface element (*e.g.*, buttons), and each user.

IUI 2017, March 13-16, 2017, Limassol, Cyprus.

DOI: http://dx.doi.org/10.1145/3025171.3025202

Author Keywords

Dwell click, Adaptive dwell time, Midas touch, Gaze tracking

ACM Classification Keywords

H.5.m. Information Interfaces and Presentation (e.g. HCI): Miscellaneous

INTRODUCTION

Eye gaze based interactions are often used to provide a user with an alternative to mouse-control navigation. They have been considered useful in several scenarios such as in zoomable information spaces [16], multimodal conversational interfaces [21] and for executing user interface commands [17]. They provide a touchless and more natural interaction with a visual interface. As opposed to speech based interaction, eye gaze can be used in public spaces, without being intrusive.

Research on human eye gaze estimation and tracking (EGET) has attained significant maturity over the past decade [1][32][34], and has entered its fourth era since the year 1879 [22]. Specialized gaze tracking hardware, such as EyeLink [1], Tobii [2] and Eye Tribe [3], have emerged. While the expensive devices such as Tobii and EyeLink provide high accuracy in estimating and tracking eye gaze, they often come at a high cost running into many thousands of dollars. On the other hand, affordable hardware such as Eye Tribe, cost an order of magnitude lesser, making these devices viable for practical use. However, they have lower frame sampling rates and image resolutions, leading to a somewhat reduced performance.

To make eye gaze based applications viable and practicable, researchers have also focused towards exploring software-driven options of EGET. Recent works, such as OpenFace [1] and OpenGazer [34], propose software approaches. Initial

[1] http://www.sr-research.com
[2] http://www.tobii.com
[3] https://theeyetribe.com

works, such as EyeTab [32], have indicated success towards using EGET on handheld portable devices, such as tablets and mobile phones, using the user-facing front camera. These early successes indicate that, the EGET technology is likely to stay and proliferate, find more practical applications.

Several research works are in the process of exploring different types of applications using eye gaze [5]. This encompasses various domains, such as neuroscience, psychology, industrial engineering and human factors, marketing and advertising and computer science. Subtasks span all the way over autism detection [6], text sentiment labeling [15], interactive application design [18] and advertising/marketing [31], to name a few amongst myriads.

Interactive application design has been a primary focus area, for researchers investigating applications of the EGET technology. Multiple user interaction applications have been proposed, that use EGET as a core constituent. A preliminary work is carried out in the Erica system [9], where the user gaze fixation point on the screen is estimated, and a menu option situated at the estimated screen location is executed. Such works, use a statically defined cumulative fixation (dwell) duration to trigger the click event, and ignore any user-specific or application-specific information. Note that, eye gaze fixation, also known as "dwell", corresponds to a relatively long stay of gaze on a visual object.

Majaranta *et al.* [12] provided an adjustable dwell (fixation) time based keyboard typing. They showed that, as the users gained gaze-typing experience over 10 sessions, their speed nearly three-folded, improving from 6.9 to 19.9 words per minute. They used adjustable dwell time, where the users could set the dwell-time for a key press (equivalent to a button click, in our setting) manually, at any time during their experiment session, to their preferred value. This work indicates the general need to adjust dwell times on actions, such as keypresses, on dwell based user interfaces, as well as indicates that dwell time reduces with usage (familiarity). Note that, while the dwell-time setting in application scenarios, such as this, is an interesting proposition, the state of the art fails to factor for a given user's exposure to (experience with) the different portions of a given application, which is likely to lead to different dwell times for different buttons of the application. The state of the art also fails to automatically adjust these dwell times, which could otherwise provide a closer-to-perfect dwell click experience for the end users.

We propose to address this problem, and, We hypothesize that users tend to be comfortable in having quicker and smoother (more error-free) interactions, with systems that they are more familiar with, and propose to apply our hypothesis to improve dwell times for application button mouse click emulation. We also aim to validate our hypothesis.

We design the *OptiDwell* algorithm, inspired by the Multi-Armed Bandits problem [27]. Initially, each given button is set with a given dwell time, for a given user. The aim is to dynamically update the dwell click time for each button, based upon the exposure (familiarity) that the user has gained with the button over her course of usage, making the interaction *fast*, *comfortable* and *accurate*, with optimized dwell times and no/minimal Midas touches [10], and validate user comfort over a post-experiment survey. As the user gains familiarity with the system over time, it dynamically *exploits* the user's growing familiarity with the application, as well as probabilistically *explores* the option of optimizing dwell-click times for the application buttons. Further, if the system reduces the dwell-time for click "too much" for the user and thereby the user generates inadvertent clicks by "accidentally" dwelling over unintended buttons, she explicitly indicates a Midas touch (by pressing spacebar, in our experiments). This, in turn, tends to increase the dwell-click time, aiming to avoid such inadvertent clicks.

We design a television (TV) application with a menu-based hierarchy, comprising of a set of text-based category icon buttons (*e.g.* sports, news, *etc.*) at the first level and image-based channel icon buttons (*e.g.* ESPN, ESPN HD, *etc.*) at the next (second) level of the hierarchy. The users browse through and select a TV channel, using gaze-based dwell clicks on the icon buttons. We demonstrate on a laptop with screen size $1,440$ px $\times$ 900 px, enabled with Eye Tribe for gaze estimation. We empirically set our lower baseline for dwell times and initial dwell time by experimenting over a combined 12 subjects. We set the lower baseline at 400 ms, *i.e.*, this is the fastest gaze dwell-time that our system explores during experiments. We set the initial dwell-time at $1,400$ ms for each button in the test application, keeping a 100 ms margin over the baseline of $1,300$ ms given in the literature [7].

We perform the experiments for adaptive dwell times on 10 subjects balanced over male and female and all within the 20-30 age group, and all different from those subjects who established the baselines. We conduct 5 sessions per subject, with 192 click instructions per session. We observe significant improvements in dwell-click time, with most users clicking most buttons as fast as the fastest manually set (lowest dwell click time) baseline of 400 ms, in the later sessions, as they gained exposure to (familiarity with) the interface over the sessions. Further, in-spite of the dwell-clicks as fast as the lowest (fastest) dwell click time manually set, we observed a best case 10.56 times reduction in the number of Midas touches over the fastest dwell click baseline, as well as a consistently high improvement in all other cases. These observations validate our hypothesis, as well establish a novel baseline for fine grained adaptation for using dwell-click.

The main contributions of our paper are the following.

- We propose a dwell-click interface, that dynamically adapts dwell time for emulating mouse click events for each user and each button of the application. Our work is the first of its kind.

- We design the *OptiDwell* algorithm for our purpose, inspired by the Multi-Armed Bandits problem. We tune the exploration dynamics of the algorithm, to model the exposure (usage experience) of the user for each given button.

- We empirically demonstrate the effectiveness of our system, by (i) jointly reducing dwell click times on application

buttons and Midas touches, and (ii) validating user comfort with a post user study survey.

RELATED WORK

The need for cognitive user interfaces has been acknowledged in the literature. For instance, Young [33] argues that future computer-based systems will need cognitive user interfaces to achieve sufficiently robust and intelligent human interaction. This work further argues that, these cognitive user interfaces will be characterized by the ability to support inference and reasoning, planning under uncertainty, short-term adaptation, and long-term learning from experience. Eye gaze movement is known to be one of the key indicators of the cognitive state of humans [14]. Eye gaze has also been used to predict the intent of the user [3]. Eye movement based computer-human interaction techniques have proliferated, both using hardware and software based solutions [13].

Mouse clicks on computers, and touch-based interaction on smartphones, are key components in computer-human interactions. Hansen *et al.* [7] are among the first ones to suggest the notion of adaptive dwell times for clicking buttons using eye gaze of users, and assigning individual button-level dwell times. This study also indicates the significance of familiarity with the input structure and mode, for accurate and productive interactions. Penkat *et al.* [20] explore dwell clicks using user eye gaze, and investigated the placement of content and button size aspects. Lutteroth *et al.* [11] create a novel scheme for dwell-based clicking of hyperlinks, and showed the efficacy of their system to closely match that of a traditional mouse-based interface. This demonstrates the effectiveness of emulating mouse-click events using user eye gaze-based dwell activities. Multiple other works exist in the literature, that focus on the accuracy, efficacy, user-comfort and other factors related to dwell-based mouse clicks.

Most modern-day dwell click systems are challenged by an absence of dynamic decisions - "when" (after how much user dwell time) should a dwell click event happen. In these systems, dwell-clicks happen after a statically defined dwell-time threshold condition is met. Early research has emanated, aiming to make the dwell click time, dynamically adjust/adapt to the user and application at hand. An early work, by Špakov and Miniotas [26], establishes a correlation between exit time (time interval between the moment a key was selected and the moment the gaze left the key) and dwell time. They map the user exit times to dwell times, and use this mapping to adapt the dwell time in their gaze keyboard setting. Majaranta and Bulling [12] also examine a gaze keyboard setting, aiming to provide a faster gaze typing experience to the end-user, by adapting the dwell times of the keys on the keyboard. Their work illustrate the significance of user familiarity and usage, as key factors to reduced dwell-click times. In these works, the dwell times are adapted *en mass* for the entire application, not as an independent adaptation separately for different keys (or, buttons, for our setting).

Analyzing the factors that determine dwell times for visual search, Becker [2] observes that the target item's visual similarity with the searched (source) item is a determinant of dwell-time to click the object. The higher the similarity, the higher will be the expected delay to click, leading to a higher dwell-time. Heo *et al.* [8] provide a home appliance infrastructure, where the menu options are displayed as well as clicked, based on user dwell behavior. In all these works, the dwell times are static threshold based, and non-adaptive in nature. Clearly, is is necessary to have independent adaptation of dwell click times for different buttons.

Dwell-click is an event implicitly inferred by an underlying system, not an explicit user action. Hence, the systems suffer from the risk of inadvertent clicks, where an interface incorrectly evaluates a given user gaze dwell as an intended interaction (click) command. This is known as a **Midas touch**. Jacob [10] conducts an early work, emphasizing on the importance of the Midas touch problem, in a dwell-click setting. Vrzakova and Bednarik [29] analyze the Midas touch phenomenon for gaze-driven events (such as clicks). They developed an annotation scheme that enabled them to create an error taxonomy and remedial strategies followed by users.

Dwell-based user interfaces have found support in the industry. MacOS Sierra, a desktop operating system released by Apple in September 2016, includes dwell control by default[4]. This indicates the significant relevance that our research bears. While Cardboard, Google's virtual reality device, is controlled by the user's head pose, the applications using this platform often employ Fuse buttons - dwell buttons in which a click can be fired during the dwell interval[5].

Thus, dwell-based interfaces are gaining prominence with advancement in research, and are becoming relevant in real life applications. Clearly, there is need and scope to improve the adaptive dwell-click paradigm, where the dwell-times associated with the click event of each independent button, can be independently adapted for a given user. At the same time, the model needs to safeguard from Midas touches, that would otherwise impair user experience by generating inadvertent clicks. Our methodology addresses this problem.

PROBLEM OVERVIEW

As evident from the exploration of literature, there is a clear need for improving the dwell-click paradigm. Investigation is needed to obtain a better understanding of the evolution of "natural" dwell-times for click, implicitly modeling user familiarity of the different sections of a given application, factoring for user agility and comfort with respect to dwell-click times, and modeling these factors to dynamically adapt dwell click times for different sections of given applications.

Objectives: The objectives of our work, are the following.

1. We propose to implement adaptive dwell-times for each button of a given application, for each user, such that the dwell-click actions are performed neither too fast, and nor too slowly. This can be attained by adapting the dwell-click time, implicitly leveraging the user's familiarity of the specific section of the application, leading to a near-optimal clicking experience in terms of speed and comfort.

[4] https://support.apple.com/kb/PH25153

[5] https://www.google.com/design/spec-vr/interactivepatterns/controls.html

2. Since, the user familiarity level will be different, for different buttons (since we experiment using button clicks in the given application), at different instants, the model needs to adapt the dwell time of each button independently, leading to a fine-grained improvement in user experience.

3. The model needs to provide adaptation such that, a Midas touch indicated by the user on a button, would tend to increase the dwell click time for that button. This will attempt to greatly reduce, if not completely avoid, Midas touches resulting from inadvertent and "too-fast" clicks.

4. We also require a qualitative validation of the model, from real human subjects, by allowing them to experience our system for sufficiently long, and further derive insights by quantifying the user adaptation to an application enabled with our system, as their familiarity with the different sections of the application grows with usage.

SOLUTION APPROACH

With the given objectives, we propose *OptiDwell*, wherein we adapt a version of the action-value based solution framework of the Multi-Armed Bandit problem [27], and introduce a novel time-decaying reward mechanism in order to model the exposure of a given user, to each given segment of a given application (in our experiment, we use buttons within the application). We subsequently perform user studies to obtain insights about the effectiveness of this model, in meeting the objectives.

While the dwell-times clearly need to be adapted, the challenges that arise are the following.

1. Should the variation of dwell time cover a continuous state space, or does exploring a discrete state space suffice? Note that, a state here corresponds to a distinct value of dwell click time - the dwell time on a button, after which the button will click because of the dwell.

2. How much to adapt (increase or decrease) the dwell click time, at a given stage, for a given button?

3. How to create a reward function, that rewards successful dwell gaze highly in the initial stages where the exploration rate should be higher in order to quickly discover an optimal dwell time for a given user on a given button, and reduces the exploration rate as the exploration process iteratively refines its understanding of a good value of dwell click time? Further, how to penalize Midas touches?

4. What is the appropriate dwell time range (boundaries) - the highest and lowest dwell times - that one needs to explore?

State Space Design

We note that, a state corresponds to the dwell time, that would result in a click action on a given button for the application usage of a given user. A state space constitutes of a set of states. In our notation, a button at a state s_t in the state space S, will have a dwell click time t; that is, if a user dwells on the button for a time duration t, a click action will be executed on that button.

First, we debate between the options of designing a discrete versus a continuous state space. In a continuous setting, a button can take all the possible dwell click time values between the upper and lower dwell click time limits. If the upper dwell time limit of a given button is t_u (*e.g.*, 1,800 ms) and the lower dwell time limit of the button is t_l (*e.g.*, 400 ms), then the state space S will comprise of all the possible infinite dwell click time values, lying between

$$|S| = |t_u - t_l + 1| \tag{1}$$

In our example, this will pick a dwell time in the continuous space of (*e.g.*, 1,800 - 400 + 1 = 1,401 milliseconds, if modeled in a continuous manner. In a degenerate approximation, this can be given, for instance, a 1 ms interval with 1,401 states, which is also a significantly high number of practically fine-grained states.

In a discrete setting, the button will take a few of the well-defined dwell click time values ("bins"), between the upper and lower dwell click time limits. If the number of bins is b, then, the number of states in the state space, defined by the number of distinct dwell click time values, will be given by

$$|S| = \left\lfloor |\frac{t_u - t_l}{b}| \right\rfloor + 1 \tag{2}$$

We note the following.

- The difference of dwell-time of a millisecond or two, for performing a dwell-click action, is unlikely to make a difference to human beings in real-life use. As observed by Davis [4], it needs 30-40 milliseconds for a visual stimulus activity even before it reaches the visual cortex. Further, as noted by Thrope *et al.* [28], the human brain takes at the best around 150 millisecond to process. Motivated by these studies, we argue that, having a huge number of states in the models, separated by a millisecond or two, will be redundant. Having discrete states, at an interval in the range of a hundred milliseconds, appears sufficient. This is also observed in the human eye saccade-and-fixation behavior, where it is noted that a typical fixation ("dwell") takes 200-600 ms, while, a typical transition (saccade) takes around 30-120 milliseconds, and further, the latency of eye movement from one object of interest to the next is around 100-200 milliseconds [23] [24].
- Further, having a continuous (or, fine-grained to the level of a millisecond) state space creates a massive state space, where even the smallest imperfection in human behavior as well as gaze tracking hardware limitations, will introduce noise in the model. Training such a large state space will require massive amounts of data, and yet, because of the inherent limitations in the behavior of the human eye as well as the hardware, the yield of such a model will be questionable, at the very least.

Based upon the above argument, we choose to opt for a discrete state space S, with each state $s_t \in S$ are separated out by a discrete time boundary. For experiments, based upon the observations made by [24], we argue that a 200 ms boundary will suffice. We, hence, choose to adapt with an interval of 200 ms across states. $Q_t(a)$ is assigned a default value as the

dwell time of its state. Each state has its own unique dwell click time, and has a one-to-one mapping with a unique bin.

The Multi-Armed Bandits Problem

In order to model the discrete state space, and deeply ingrain the reward-penalty paradigm associating with genuine and unintentional button clicks made by our system based on user dwell behavior, we draw inspiration from the Multi-Armed Bandits problem [27]. This belongs to the *estimator algorithms* class in the learning automata literature. The solution uses the notion of *action value* proposed by Watkins [30], and an ϵ-greedy method that the author uses.

Introducing the Multi-Armed Bandits Problem: We create a state for each bin identified between the dwell click times t_l and t_u (both inclusive). This gives us $n = |S|$ choices of states. In the n-armed bandits problem, the system is presented with a choice of n different options (namely, *actions*), and after each choice, the system receives a numerical reward having a value, that is chosen from a stationary probability distribution, depending upon the action. The aim is to maximize the total reward over time. Since, the reward value of each action are not known fully in advance with certainty (even though an estimate is known), optimally solving the problem remains non-trivial.

Our formulation of the solution, thus, becomes a task of minimizing dwell times and Midas touches, as a non-stationary Multi-Armed Bandit problem, with the bandits as the dwell-time bins. Our problem setting needs a non-stationary model, because the reward distribution of each dwell-time bin is not a constant. It changes for each button independently, with usage of the button.

Since the estimates of each action value at each state are known (note that, each action value uniquely corresponds to one state), hence, at each time step, it is trivial to determine at least one action, with the greatest estimated action value - simply by finding the maximum value. This is the *greedy* action. Selecting a greedy action, using the current knowledge of value of actions, is termed as *exploitation*. On the other hand, selecting a non-greedy action is *exploration*. Exploration is used to improve the estimate of the currently non-greedy action's reward values. While exploitation is locally greedy in that it produces the best immediate reward value, exploration, in spite of its lower immediate reward value, is necessary to produce a greater reward value in the long term. Note that, at a given step, one can only either *exploit* or *explore*, not both.

Let $Q_t(a)$ be the estimated value of an action a at the t-th time step. If before the t-th time step, the action a gets chosen K_a times, yielding rewards $R_1, R_2, ...R_{K_a}$, then the value of $Q_t(a)$ is estimated as

$$Q_t(a) = \frac{R_1 + R_2 + ... + R_{K_a}}{K_a} \tag{3}$$

At $K_a = 0$, a default value of $Q_t(a)$ is assigned. As $K_a \to \infty$, by law of large numbers, $Q_t(a)$ converges to $q_*(a)$, the true (actual) value of action a.

Modeling Exploration versus Exploitation: In order to ensure that while the current knowledge gets exploited to maximize the immediate reward, the system also continues to explore in order to improve the overall rewards in the long term, we opt for a semi-uniform strategy based modeling paradigm. In a semi-uniform paradigm, an adaptation algorithm is made to behave greedily, performing exploitation most of the turns; however, it explores uniformly among the rest of the state space during the remaining turns. This is implemented by introducing a parameter ϵ, and initializing with $0 < \epsilon < 0.5$, exploring if a uniform random number generated between 0 and 1 is smaller than epsilon, and exploiting otherwise. In experimental settings, we initialize our system with $\epsilon = 0.25$. In the cases of exploration, all the states that would be explored, are chosen with a uniform random probability.

Creating the Non-Stationary Model: As noted earlier, our problem setting is non-stationary. As a given user exposure to (familiarity with) the given application evolves with usage, for each button, the dwell click time adapts, attempting to better approximate the "natural" dwell clicking behavior of the given user. To model this phenomenon, we decay ϵ associated with the user for the button, essentially leading to an ϵ-decreasing strategy. The combination of (a) greedy choices of exploitation versus exploration, based upon the value of ϵ, and (b) the decay of ϵ for each button with usage by a given user, makes our approach a hybrid of ϵ-greedy and ϵ-decreasing strategies. We model the decay of epsilon as

$$\epsilon = 0.25 * e^{-\frac{ClickNo}{\tau}} \tag{4}$$

where $ClickNo$ denotes the count of the click made on the given button, and τ is a constant.

For our experiments, we use $\tau = 56$, based upon trial and error, and with a constraint that the value of ϵ does not fall under 0.01 (a small positive quantity) in the process, given that we start with the value $\epsilon = 0.25$. Also note that, ϵ is never let to fall to 0, so that, the exploration is never stopped, which makes the algorithm amenable to work for relatively rare, but possible, necessities to adapt in the long future.

Further, in a non-stationary reinforcement learning problem like ours, it is important to assign a higher weightage to recent rewards, compared to those longer back in the past. This is implemented using a constant step-size parameter, as observed in the literature [19]. For an action, if Q_k denotes the estimate of the k^{th} reward, *i.e.*, the average of the first $(k-1)$ rewards, and if R_k denotes the k^{th} reward, then the average of all k rewards is obtained by

$$Q_{k+1} = Q_k + \alpha\Big[R_k - Q_k\Big] \tag{5}$$

In our setting, the step-size parameter α is a constant, where $0 < \alpha \leq 1$. For our experiments, we empirically set α to 0.6. The quantity $(1 - \alpha)$ being less than 1, the given to the i^{th}-step reward R_i in computing R_k will decrease, as k increases. It has been shown in the literature that, the weight decay is exponential with respect to $(1 - \alpha)$ [19]. Thus, with our reward shaping function, we implement a non-stationary version of an ϵ-decaying Multi-Armed Bandits problem for adapting the dwell times of each button, for each user.

Modeling Midas Touch: Since, the familiarity of a user with an interface, will tend to optimize the dwell click times to the "natural" dwell click times of the user for that part of the application interface, hence, exploration is normally carried out only in those states within the state space, that have lower dwell-time values, compared to the current state. The exploration process does not explore the states with higher dwell times compared to the current state, it only explores the states with lower dwell times. The system is made to move to a state with higher dwell time, from its current state, by implementing an appropriate assignment of Midas touch reward values. The Midas touch, which is an indication given by the user, that the system has made an unintended dwell click, is intuitively an event of having "clicked too fast". This requires assigning a penalty on the state. The objective of this penalty will be to tend the system receive more reward by exploring the "upward" states (states with higher dwell click times), thereby tending to push the dwell click time to a value higher than the current value. Such an objective is intuitive, because, if a system clicks a button too fast for the user, then the dwell click time for that button needs to increase, in order to disable the accidental dwell clicks on the interface. By default, the system assumes each dwell click to be a genuine click. If, after an assumed genuine click, a Midas touch is indicated by the user, the last incorrectly assumed genuine click is completely undone. This is done by restoring $chosenBin$ and $exploitedBin$ from memory and back calculating for the Q values (i.e. $expectedRewards$) before the update. Q values can also be restored from memory, instead of back calculating. Note that if the user indicates a Midas touch, the Midas touch updates occur on the button that is last clicked by the user.

Assigning Reward Values

In our Multi-Armed Bandit based approach, we create a reward value model, that will reward genuine clicks, penalize Midas touches, and assist the system to tend each button to the state that a given user finds most comfortable. We observe that, the time spent in an end-to-end dwell click event of a button, including the user indicating inadvertent clicks, will have two components.

1. The time that the user dwells on a button for a click.
2. The time taken to indicate a Midas touch, if any, by indicating via an explicit reaction. In case there is no such indication by the user, then the system assumes the click to be genuine. In case there is an indication, then the indication is associated to the last click action observed before the indication was made.

We model the "interaction time" as a sum total of the time user dwells on a button for clicking, and the time to indicate a Midas touch, and use this value for reward shaping, as

$$Interaction\ time = Dwell\ click\ time\ on\ a\ button + Time\ taken\ to\ indicate\ Midas\ touch \quad (6)$$

In case no indication is given by the user of a Midas touch, the time taken to indicate Midas touch component is set to zero. Otherwise, this value is captured as the time elapsed between the dwell click, and the user taking the explicit action to indicate the Midas touch. In our experiments, we ask (and train) the user to press the keyboard spacebar, to indicate a Midas touch. We observe that, the average time taken to press spacebar is around $1,350$ ms, distinguishing the interaction time of a Midas touch significantly from a genuine click.

We observe that, the solution quality will not be impacted, if we model such that the genuine click interactions receive a high positive reward, and Midas touches receive a low (but positive) reward. Therefore, for ease of modeling, we define a large offset number L, and compute reward of an action as

$$R_a = L - Interaction\ time \quad (7)$$

This ensures, Midas touches receive a lower R_a value, as the user takes a non-zero time to indicate the Midas touch. For our experiments, we set L to 5, which effectively assigns the current reward value of a given state for a given button as $5,000$ ms (5 seconds), minus the dwell click time associated with the state (bin), and in case of a Midas touch event, also subtract the time taken by the user to indicate. Further note that, the interaction time will be smaller for the states with faster dwell, since the dwell click time on such buttons will be lesser, and hence, such buttons will receive higher rewards for successful dwell clicks. Thus, the reward model ensures that, the system will tend towards the fastest clicks possible, while minimizing the Midas touches.

The Overall Solution

Our overall solution design, thus, is built upon a non-stationary Multi-Armed bandits problem, a user exposure (familiarity)-specific balance between exploitation and exploration such that exploration rates fall off (but never zero) with usage by the user, and a reward model ensuring that the system moves towards the optimal click times, while minimizing Midas touches. Please note that, for experiments, we bootstrap with $\epsilon = 0.25$. Further, we assume the initial dwell click time for each button as $1,400$ ms (1.4 seconds). The overall methodology is presented in Algorithm 1.

EXPERIMENT

To examine the user experience for our proposed *OptiDwell* algorithm, we designed a generic user interface for a hierarchical menu system, that can be navigated via a dwell click paradigm. The purpose of the study was to observe, whether the adaptive facet of the algorithm was indeed useful and noticeable to the user, and, what dwell time value does such a system stabilize on after sufficient and repeated usage. The system can mostly stabilize on the last dwell click time values that the interface settles on (except for explorations at a low frequency), for each user and each button. We thus also examine the user reaction to the last-set dwell click times.

Scope of Our Experiments

We first designed a menu interface, representative of the use-cases discussed. Subsequently, we performed experiments to set the high and the low dwell time limits. We then performed a between-subjects study, with 2 control groups, and 1 test group that was exposed to the adaptive interface. The control

Algorithm 1 OPTIDWELL ALGORITHM

for initializing each button **do**
    $L \leftarrow$ high positive value (**REM** 5 for experiments)
    $\alpha \leftarrow$ CONSTANT (**REM** 0.6 for experiments)
    $\tau \leftarrow$ CONSTANT (**REM** 56 for experiments)
    cdct = 1.4 seconds (and the corresponding state)
    REM cdct $\leftarrow$ currentDwellClickTime
    if new user **then**
        $\epsilon \leftarrow 0.25$, clickNo $\leftarrow 0$
        maxBin = #states (**REM** 8 for experiments)
        for each state **do**
            if dwell click time of bin (state)$<$ cdct **then**
                state.expectedRewards = 0
            else
                state.expectedRewards = $L-$ dwell click time of state
            end if
        end for
    else
        load user's prior ϵ, cdct, clickNo, expectedRewards
    end if
    EpsilonGreedySelection()
    $S = \{s_i\} \leftarrow$ set of states (action values), one corresponding to each bin
    EpsilonGreedySelection()
end for

for each dwell click event **do**
    clickNo $\leftarrow$ clickNo + 1
    $\epsilon \leftarrow 0.25 * e^{-(clickNo/\tau)}$
    if $\epsilon < 0.01$ **then**
        $\epsilon \leftarrow 0.01$
    end if
    GenuineClick()
    EpsilonGreedySelection()
end for
if Midas touch indicated **then**
    restore expectedRewards, chosenBin, exploitedBin
    for all bin b (states) where dwell click time $>$ dwell click time of chosenBin **do**
        UpdateExpectation (bin, time to indicate Midas touch)
    end for
end if

EpsilonGreedySelection():
generate random number x, s.t. $0 < x \leq epsilon$
if $x \geq \epsilon$ **then**
    (exploit) chosenBin $\leftarrow$ bin (state) with maximum value in state.expectedRewards
    exploitedBin $\leftarrow$ chosenBin
else
    (explore) chosenBin $\leftarrow$ random bin between the first (lowest dwell click time) bin and exploitedBin
end if
update dwell time of button corresponding to chosenBin

GenuineClick():
for all bin b (states) where dwell click time $>$ dwell click time of chosenBin **do**
    UpdateExpectation (b, 0)
end for

UpdateExpectation (bin, time to indicate Midas touch):
expectation $\leftarrow$ state.expectedReward of chosen bin
dwlRwd $\leftarrow$ dwellTime corresponding to bin
reward $\leftarrow$ L - time to indicate Midas touch - dwlRwd
newExpectation $\leftarrow$ expectation + α.(reward - expectation)
update state.expectedReward at bin with newExpectation

groups were given fixed high and low dwell times to compare their comfort levels with the adaptive dwell time groups. We design 8 states, having dwell click times 400, 600, 800, 1,000, 1,200, 1,400, 1,600 and 1,800 ms respectively. In the following subsections, we describe the interface design, the task design for the user study, parameters for the control and treatment groups and finally summarize the results.

Interface Design

To create a generic menu interface, we surveyed current TV interface design guidelines to come up with a representative interface keeping our technological constraints in mind. We used the Eye Tribe gaze tracker with an operating range of 45 cm $\times$ 75 cm. The interface was shown on a LCD monitor with a viewable area of dimensions 40 cm $\times$ 25.5 cm, and a display resolution of 1,440 px $\times$ 900 px.

The menu was 2 levels deep, with 4 buttons in the first level, and 5 buttons in the second level with 1 Home button. The button sizes were derived from [7], fixed at 208 px $\times$ 244 px to fit the content, image and text and maintain uniform visibility. They were fixed by repeated testing with the eye tribe and monitor setup to account for eye gaze jitter. Each button gives a color based feedback to highlight the user, that it is being dwelled on. The color transitions as initial bright green $\rightarrow$ yellow $\rightarrow$ red with a fade out $\rightarrow$ fade in, with animation between each color pair. These colors are chosen due to their high sensitivities. The transition is equally spaced for the dwell click time of that button. For example, if the dwell click time is 900 ms, each of the 3 transition takes 300 ms, as shown in Figure 1. The border also thickens to further define the highlighting. A pop sound accompanies the click, providing audio feedback.

Task Design

The user has two inputs to the system: her gaze and her press of spacebar for recording unwanted clicks (Midas touches). Each user goes through a total of 192 tasks, 12 clicks per icon

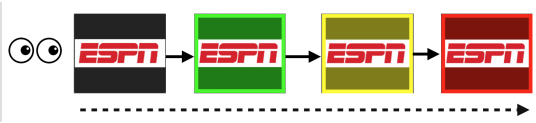


Figure 1. Gradual transition of button state, as dwell duration increases

for 16 icons. The order of these tasks are randomized, but all 3 groups go through the same tasks for every session.

A landing page is shown first, with the system usage instructions. To proceed, the user dwells on the "Begin" button. This button is initialized with a constant dwell time of $1,400$ ms, and is not accounted for in the study. The user then faces the first (topmost) level of the menu. This menu has text buttons, shown in Figure 2. The instructions for the current task are shown in the bottom right corner of the screen as plain text, and also relayed via audio. The user then selects the top menu, to go to the next level, for example from "News" to find "CNN", and then she has to dwell on the correct choice to finish the task. The inner menu has image buttons, shown in Figure 3. To ensure that, the button laid out at the same position as ("directly below") the previous button on the new screen, does not get dwell-clicked by accident, the user has to first gaze to the space outside the button (the "non-button space"), and then either gaze back on that button, or dwell on a different button, based on what she intends to click next.

If incorrect, then the audio instructions are "Incorrect, you have to find <current object name> for this example." A correct answer prompts the audio, "Correct! Now find <next object name> and the instruction text is updated." Both feedbacks are provided after a $1,000$ ms pause. This pause allows users to decide if the click was intended or not. They can record unintended clicks (Midas touches) using the spacebar via the keyboard attached. If the user records a Midas touch, the system acknowledges to the user via audio, "Error noted", and the screen gently flashes deep red color as visual feedback. The study setup is shown in the Figure 4.

User Study

We conducted a user study with 4 users in the control groups, that we exposed to the interface with fixed dwell times of 400 ms and $1,400$ ms. All users were from a lab environment and were regular computer users. The adaptive dwell time system was used by 10 users. All users were given a trial session of 4-5 minutes with a similar interface but different icons with very high dwell time to adjust to the gaze an input paradigm. Then the same interface would be used with a very low dwell time to make the user understand how to register unintended clicks. The users then filled a pre-questionnaire inquiring about their eye and general fatigue before every session. All users in all groups finished 5 sessions each, with an average gap of 18 hours (between 5 to 24 hours, chosen at random).

Selecting High and Low Dwell Times

High dwell times between $1,300$ ms [7] and $3,000$ ms [25] are recommended. We set the initial dwell-time at $1,400$ ms for each button in the test (adaptive) application, keeping an extra 100 ms margin over the baseline of $1,300$ ms observed in the literature [7]. We determined a high dwell time value of $1,400$ ms, based upon observations made by [7], factoring in an extra 100 ms margin for additional safety against potential dwell click errors, and validating this with a pilot run. The low dwell time was chosen after performing another pilot run on 2 users to the experimental interface at 200 ms, and 2 more users at a 300 ms constant dwell times. The 200 ms dwell click interfaces registered more than 50 Midas touches each, and the 300 ms interface registered more than 40 Midas touches each. Further, the users reported high fatigue even before the sessions proceeded halfway for both values. Both the 200 ms interface users abandoned the system midway, complaining discomfort of use. The 300 ms interface users completed the trial run, but complained of severe discomfort of usage also. Hence, we discarded 200 ms and 300 ms as too low to be useful dwell click times. 400 ms constant dwell times were reported to be much more usable (reported later in the results), and hence we used 400 ms as the lower baseline for fixed dwell times.

Data Collection

In the initial run, in spite of a pre-experiment hands-on training, most of the users made a few to many errors, to indicate Midas touched by pressing the spacebar key. After the first session, they were mostly able to indicate the Midas touches correctly, thereby reducing the number of unreported Midas touches. Sometimes the Eye Tribe would lose track of the users resulting in a system freeze, they would resume within a few seconds. If not, then the Eye Tribe was re-calibrated after pausing the session, and then resuming the session after a good (5 or 4 star) calibration. A similar proportion of users reported weariness for all the high, low and adaptive dwell click systems, so we logically attribute to variations in the starting time of the session, such as early morning or late evening, and observe that this is independent of the dwell times (high vs. low vs. adaptive), indicating inherent characteristics of dwell-click based systems.

RESULTS

Improvement of Dwell Time with Adaptation

As the users gain familiarity with the dwell based interface, we observe that the the *OptiDwell* system significantly reduces the average dwell time for clicks, across all the buttons. This is captured by faster dwell clicks, captured in Figure 5. This establishes the first part of our baseline premise, that, the users tend to be comfortable in having **quicker** interactions, with systems that they are more familiar with.

Reduction of Midas Touches with Familiarity

As the users gain familiarity with the dwell based interface, we observe a significant reduction of Midas touches made by the user. This establishes the second part of our baseline premise, that, the users tend to be comfortable in having **smoother** (more error-free) interactions, on systems that they are more familiar with. This further provides a justification for having adaptive dwell click time based interfaces. The trends are similar for adaptive as well as non-adaptive interfaces, and for non-adaptive interfaces, both for high as well as low dwell click times. The improvements of usage, in terms of reduction of Midas touches, is shown in Figure 6. In each

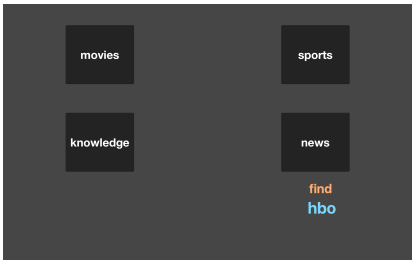


Figure 2. Home page screen, with text buttons only. Instruction at bottom right corner.

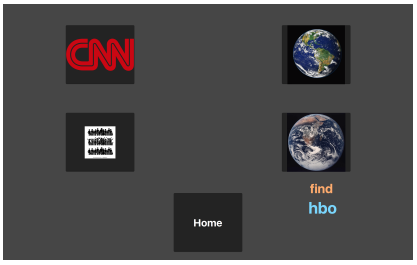


Figure 3. One of four second-level screens. Dwell-clicking "News" navigates here.

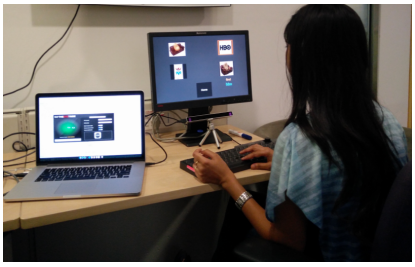


Figure 4. Our experimental setup with Eye Tribe and keyboard.

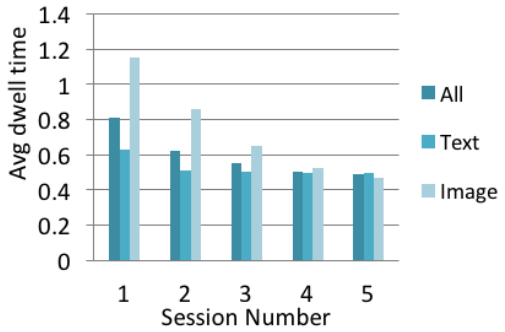


Figure 5. Trends of dwell click time, for different types of buttons, for the adaptive system, evolving over user sessions.

Ad. Type	Sess. #1	Sess. #2	Sess. #3	Sess. #4	Sess. #5	S1/S5 Rat.
All, Ad.	1.74	0.94	0.85	0.62	0.36	4.78
All, Lo.	5.65	4.65	3.74	3.15	2.40	2.35
All, Hi.	0.047	0.09	0	0	0.091	0.52
Lo./Ad.	3.24	4.92	4.41	5.05	6.59	-
Txt, Ad.	1.44	0.95	0.58	0.59	0.29	4.93
Txt, Lo.	4.09	2.41	1.36	0.82	0.82	4.96
Txt, Hi.	0	0	0	0	0	0/0
Lo./Ad.	2.85	2.54	2.36	1.39	2.83	-
Img, Ad.	2.31	0.95	1.33	0.69	0.51	4.57
Img, Lo.	8.47	8.34	7.59	7.10	5.34	1.59
Img, Hi.	0.13	0.25	0	0	0.26	0.50
Lo./Ad.	3.67	8.78	5.71	10.25	10.56	-

Table 1. Trend of Midas touches per session (normalized), with increasing user familiarity. Here, ad. ← adaptive, hi. $gets$ high, lo. ← low, sess ← session, txt ← text, img ← image, S1/S5 rat. ← Midas touch ratio between session 1 and session51, lo./ad. $gets$ Midas touch ratio of low dwell model and adaptive dwell model for a given session.

subfigure, the horizontal axis represents the session sequence of a given user. Note that, in our experiments, each user had undergone 5 sessions, hence 5 entries in the axis. The vertical axis represents the average number of Midas touches made by the users (normalized per 100 dwell clicks), in the given session sequence, *i.e.*, the average of the first session of the users, the average of the second session of the users, and so on.

The improvements attained by users over sessions, is captured in Table 1. The *S1/S5 ratio* denotes the ratio of the average number of Midas touches observed across users, in session 1 to session 5. A high value indicates a much-lower Midas touch count in session 5, compared to session 1, indicating significant improvement. Regression trend lines for each button for each user over the session show a decrease with steeper slopes for text buttons than image buttons ($p < 0.02$). The improvement is stark for the adaptive model, especially the images. Although the adaptive system settled often at a dwell click time of 400 ms for most buttons and most users, the number of Midas touches were significantly lower for the adaptive system, compared to the low dwell click time system also having a 400 ms click. The best users (lowest Midas touches) in the adaptive system had close to zero Midas touches in the later sessions (Figure 7), performing more accurately than the best users for both the other (high and low dwell time) systems. In general, a ratio of the number of low to adaptive Midas touches, for the average user, indicate stark improvements in terms of error-free interactions with the system, as noted in all the three *Lo./Ad.* rows in Table 1. For images, after the adaptation process in the final (5^{th}) session, this ratio is a high 10.56. This indicates the effectiveness of gradual adaptation of dwell time.

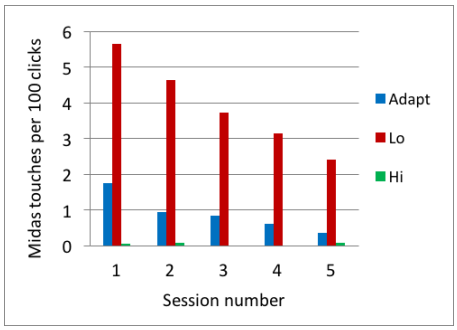
User Experience of The System

We performed pre and post evaluations, for each user for each session. We asked the users to rate their eye and general fatigue at the start and end of each session on a 7 point Likert scale, shown in Figures 8 and 9 respectively.

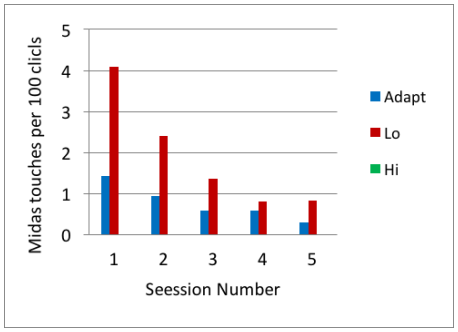
The post evaluation measured the effects of the adaptive and non-adaptive systems on 4 questions on a 7-point Likert scale. The users were blind to the system type (adaptive, high, low). The questionnaire mentioned that the dwell time may or may not have been modified. The 4 questions pertained to the user comfort ("very uncomfortable" to "very comfortable"), system responsiveness ("too slow" to "too fast"), the experience that the adaptation provided ("very obstructive" to "very helpful"), and their overall experience("very frustrating" to "very satisfying"). We aggregate the answers for all users per session, to compare the adaptive system with the high and low dwell time systems via a One way ANOVA, and found that the adaptation was both significantly more comfortable ($F(2, 12) = 15.44$, $p = 0.004$) and responsive than both the low and high constant dwell times ($F(2, 12) = 11.09$, $p = 0.02$). The adaptive system provided a significantly more satisfactory user experience ($F(2, 12) = 21.48$, $p = 0.0001$) both with respect to the adaptation and the overall experience ($F(2, 12) = 16.73$, $p = 0.0003$), compared to both the other systems ($p < 0.05$ for all comparisons).

DISCUSSION

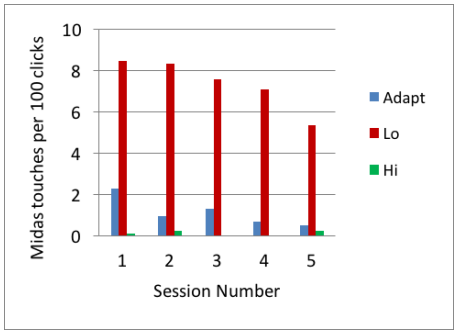
Our experiment indicate that, *OptiDwell* plays a balancing role, in optimizing per-user per-button dwell click time, while keeping Midas touches at a minimal. This establishes our



(a) Improvement for All Buttons



(b) Improvement for Text (Category) Buttons



(c) Improvement for Image (Channel) Buttons

Figure 6. Midas Touch improvements with increasing user familiarity. As the user familiarity increases, Midas touches significantly reduce in number.

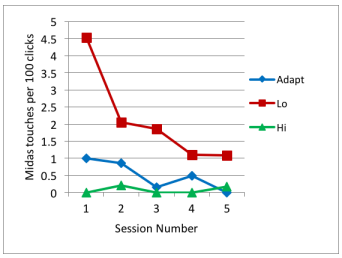


Figure 7. Midas touches of the most comfortable users on the three dwell time systems

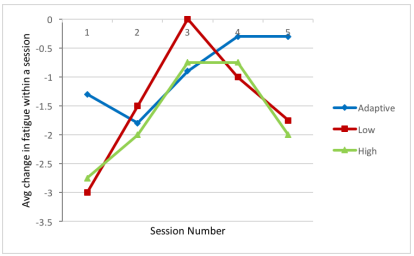


Figure 8. The absolute eye fatigue decreases more for the adaptive system

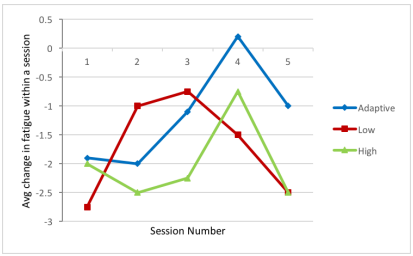


Figure 9. The absolute fatigue decreases more for the adaptive system

work as a robust first-of-its-kind platform for adaptive dwell clicks, and creates a baseline for future research.

User comfort gained by adaptation: An interesting observation is that, while the adaptive model finally adapted to 400 ms dwell time for almost all the subjects (Figure 5), which is the same dwell time as the lower baseline, the number of Midas touches even in the final (5^{th}) session of the fixed lower baseline were much higher compared to the adaptive models (Figure 6). The post-survey indicated higher user comfort for the adaptive dwell over the low dwell, even when the adaptive dwell system was clicking buttons mostly at 400 ms.

Eye fatigue: The proposed adaptive dwell click system was accepted well by the users. As shown in Figure 8 and Figure 9, the fatigue caused by each given session, measured as the difference of eye fatigue at the start of the session and that at the end of the session, is lower, on an average, with our adaptive dwell click time based system, compared to the high and low dwell click time based systems. In their post-survey, the users did not report discomfort, although the t-test did not conclusively establish the significance. Nevertheless, the indications are encouraging, and calls for a focused examination, across different adaptive dwell click based applications with different button types, and diverse individuals with different backgrounds and demographic attributes, in the future.

Impact of choosing α and ϵ values: In our setting, the initial rate of exploration needs to be high enough, while ensuring sufficient exploitation of the states. Further, over time, the rate of exploration needs to fall, as the system stabilizes to the appropriate dwell click values. This rate of fall needs to be fast enough that so the user does not remain in non-optimal dwell states for too long, but slow enough for the user to gain familiarity with the system in order to minimize Midas touches. Choosing a high initial value for ϵ is also likely to cause rapidly changing dwell click times over the entire permissible ranges of dwell click time values, caused by over-exploration, and leading to user discomfort. Our choice of the initial value of epsilon (ϵ) as 0.25, the rate of fall of ϵ, and the final stabilization of ϵ at 0.01, are all based upon this requirement. Further, since ϵ is never let to fall to 0 by design, the exploration process never completely aborts. This makes our approach effective for possible adaptation requirement in the long run.

Choosing alpha (α) low (nearly zero) will alter the reward structure to make the model highly stationary, and choosing α high will make it significantly non-stationary. This justifies the selection of α in the mid-range, and we chose $\alpha \leftarrow 0.6$ for our experiments, using an initial trial-and-error on a handful of pilot participant subjects. In future, it will be interesting to explore and create formal methodology to assign the values of α and ϵ, as well as, the decay rate of ϵ.

Improvements made by users: We observe that, as users gain familiarity over time, our algorithm converges to their optimal dwell time faster, as seen in Figure 5. The number of Midas touches also reduces, as seen in Figure 6. As indicated in Table 1, the rate of reduction of Midas touches is high for the users with our adaptive system, compared to the fixed (both high and low) dwell click time based systems. We further observe from Table 1 that, the rate of improvement made by the users on the text buttons are similar for the low and adaptive dwell settings, while the rate of improvement made by the users on the image buttons are significantly higher for in the adaptive dwell click settings compared to the others. An examination of the individual user data (not presented in the paper for space constraints), reveals this as a consistent trend. While there could be several possible explanations to

this phenomenon, we do not attempt to explore further, as this is not in scope of our problem.

Choosing a different model: Another area that needs more exploration in the future, is the model chosen for adaptation. Our work has established a first-of-its-kind baseline for using adaptive dwell click for user interface design, using the Multi-Armed Bandits problem model. However, it is important to perform subsequent explorations, to establish the viability of other adaptive dwell click models, by deploying those models, and comparing performance with our system. While our model does dramatically decrease Midas touch rates, it does rely on sufficient hand tuning in terms of choosing parameters such as α and ϵ and appropriately choosing the dwell bins. Additionally, the OptiDwell algorithm does not explicitly optimize user comfort, but aims to reduce midas touches (or more specifically, optimize $interactiontime$). We expect more user data and better user modeling to make way for more superior models (e.g. Q-learning).

Handling more complicated interactions: Click and drag interactions are more complicated and can be explored in future research. Performing them via this paradigm might require revisiting the core concept of dwell time, as they may require capturing the interaction context too. A simple increased dwell time will be an abrupt mapping to a traditional long click. So, we need to rethink the dwell time bins, or completely capture the intention of a long click for dragging to be a more acceptable interaction. Also dragging will have to made smooth if the user looses track of where the target is, to accommodate searching glances.

CONCLUSION

We proposed *OptiDwell* for intelligent adjustment of dwell click times. Using the hypothesis that users tend to be comfortable in having faster interactions with familiar system, we designed a framework that, over time, models implicit user familiarity with different parts of a given application interface, and thereby dynamically establishes per-user and per-button dwell times to emulate mouse click events. *OptiDwell* comprises of explorations and exploitations, and different reward values for genuine dwell-clicks and Midas touches, for dwell click time adaptation. We experimented with a television channel catalog application, comprising of a mix of text and image buttons. We demonstrated the effectiveness of our system, by performing a user study on the adaptive interface on 10 computer-savvy subjects, in the 20-30 year age-group, balanced between males and females. Our experiments indicated a combination of (a) optimization in dwell click times (the fastest dwell-clicks that an individual is comfortable with) and (b) reduction in Midas touches (as high as a 10-fold reduction in the best case), along with high user comfort, as each user's familiarity with the interface increased over the 5 sessions. Our model can be used to create dwell-click interface based applications, deeply ingraining the clicking behavior of each user for each button.

REFERENCES

1. Baltru, T., Robinson, P., Morency, L.-P., et al. Openface: an open source facial behavior analysis toolkit. In *2016 IEEE Winter Conference on Applications of Computer Vision (WACV)*, IEEE (2016), 1–10.
2. Becker, S. I. Determinants of dwell time in visual search: similarity or perceptual difficulty? *PLoS One 6*, 3 (2011), e17740.
3. Bednarik, R., Vrzakova, H., and Hradis, M. What do you want to do next: A novel approach for intent prediction in gaze-based interaction. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ETRA '12, ACM (New York, NY, USA, 2012), 83–90.
4. Davis, R. The human operator as a single channel information system. *Quarterly journal of experimental psychology 9*, 3 (1957), 119–129.
5. Duchowski, A. T. A breadth-first survey of eye-tracking applications. *Behavior Research Methods, Instruments, & Computers 34*, 4 (2002), 455–470.
6. Grice, S. J., Halit, H., Farroni, T., Baron-Cohen, S., Bolton, P., and Johnson, M. H. Neural correlates of eye-gaze detection in young children with autism. *Cortex 41*, 3 (2005), 342–353.
7. Hansen, J. P., Johansen, A. S., Hansen, D. W., Itoh, K., and Mashino, S. Command without a click: Dwell time typing by mouse and gaze selections. In *Proceedings of Human-Computer Interaction–INTERACT* (2003), 121–128.
8. Heo, H., Lee, J. M., Jung, D., Lee, J. W., and Park, K. R. Nonwearable gaze tracking system for controlling home appliance. *The Scientific World Journal 2014* (2014).
9. Hutchinson, T. E., White, K. P., Martin, W. N., Reichert, K. C., and Frey, L. A. Human-computer interaction using eye-gaze input. *IEEE Transactions on systems, man, and cybernetics 19*, 6 (1989), 1527–1534.
10. Jacob, R. J. What you look at is what you get: eye movement-based interaction techniques. In *Proceedings of the SIGCHI conference on Human factors in computing systems*, ACM (1990), 11–18.
11. Lutteroth, C., Penkar, M., and Weber, G. Gaze vs. mouse: A fast and accurate gaze-only click alternative. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, ACM (2015), 385–394.
12. Majaranta, P., Ahola, U.-K., and Špakov, O. Fast gaze typing with an adjustable dwell time. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ACM (2009), 357–360.
13. Majaranta, P., and Bulling, A. Eye tracking and eye-based human–computer interaction. In *Advances in physiological computing*. Springer, 2014, 39–65.
14. Marshall, S. P. Identifying cognitive state from eye metrics. *Aviation, space, and environmental medicine 78*, 5 (2007), B165–B175.
15. Mishra, A., Kanojia, D., Nagar, S., Dey, K., and Bhattacharyya, P. Leveraging cognitive features for sentiment analysis. *CoNLL 2016* (2016), 156.

16. Mollenbach, E., Stefansson, T., and Hansen, J. P. All eyes on the monitor: gaze based interaction in zoomable, multi-scaled information-spaces. In *Proceedings of the 13th international conference on Intelligent user interfaces*, ACM (2008), 373–376.

17. Morency, L.-P., and Darrell, T. Head gesture recognition in intelligent interfaces: the role of context in improving recognition. In *Proceedings of the 11th international conference on Intelligent user interfaces*, ACM (2006), 32–38.

18. Morimoto, C. H., and Mimica, M. R. Eye gaze tracking techniques for interactive applications. *Computer Vision and Image Understanding 98*, 1 (2005), 4–24.

19. Oksanen, J., Lundén, J., and Koivunen, V. Reinforcement learning based sensing policy optimization for energy efficient cognitive radio networks. *Neurocomputing 80* (2012), 102–110.

20. Penkar, A. M., Lutteroth, C., and Weber, G. Designing for the eye: design parameters for dwell in gaze interaction. In *Proceedings of the 24th Australian Computer-Human Interaction Conference*, ACM (2012), 479–488.

21. Prasov, Z., and Chai, J. Y. What's in a gaze?: the role of eye-gaze in reference resolution in multimodal conversational interfaces. In *Proceedings of the 13th international conference on Intelligent user interfaces*, ACM (2008), 20–29.

22. Rayner, K. Eye movements in reading and information processing: 20 years of research. *Psychological bulletin 124*, 3 (1998), 372.

23. Ross, J., Morrone, M. C., Goldberg, M. E., and Burr, D. C. Changes in visual perception at the time of saccades. *Trends in neurosciences 24*, 2 (2001), 113–121.

24. Sibert, L. E., and Jacob, R. J. Evaluation of eye gaze interaction. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems*, ACM (2000), 281–288.

25. Špakov, O. Comparison of gaze-to-objects mapping algorithms. In *Proceedings of the 1st Conference on Novel Gaze-Controlled Applications*, ACM (2011), 6.

26. Špakov, O., and Miniotas, D. On-line adjustment of dwell time for target selection by gaze. In *Proceedings of the third Nordic conference on Human-computer interaction*, ACM (2004), 203–206.

27. Thathachar, M., and Sastry, P. A class of rapidly converging algorithms for learning automata.

28. Thorpe, S., Fize, D., Marlot, C., et al. Speed of processing in the human visual system. *nature 381*, 6582 (1996), 520–522.

29. Vrzakova, H., and Bednarik, R. That's not norma (n/l): a detailed analysis of midas touch in gaze-based problem-solving. In *CHI'13 Extended Abstracts on Human Factors in Computing Systems*, ACM (2013), 85–90.

30. Watkins, C. J. C. H. *Learning from delayed rewards*. PhD thesis, University of Cambridge England, 1989.

31. Wedel, M., and Pieters, R. A review of eye-tracking research in marketing. *Review of marketing research 4*, 2008 (2008), 123–147.

32. Wood, E., and Bulling, A. Eyetab: Model-based gaze estimation on unmodified tablet computers. In *Proceedings of the Symposium on Eye Tracking Research and Applications*, ACM (2014), 207–210.

33. Young, S. Cognitive user interfaces. *IEEE Signal Processing Magazine 27*, 3 (2010), 128–140.

34. Zielinski, P. Opengazer: open-source gaze tracker for ordinary webcams. *Samsung and The Gatsby Charitable Foundation, http://www. inference. phy. cam. ac. uk/opengazer* (2007).