Gaze as an Indicator of Input Recognition Errors

CANDACE E. PEACOCK, Reality Labs Research, Redmond, USA
BEN LAFRENIERE, Reality Labs Research, Toronto, Canada
TING ZHANG, Reality Labs Research, Redmond, Canada
STEPHANIE SANTOSA, Reality Labs Research, Toronto, Canada
HRVOJE BENKO, Reality Labs Research, Redmond, USA
TANYA R. JONKER, Reality Labs Research, Redmond, USA

Input recognition errors are common in gesture- and touch-based recognition systems, and negatively affect user experience and performance. When errors occur, systems are unaware of them, but the user's gaze following an error may provide valuable cues for error detection. A study was conducted using a manual serial selection task to investigate whether gaze could be used to discriminate user-initiated selections from injected false positive selection errors. Logistic regression models of gaze dynamics could successfully identify injected selection errors as early as 50 milliseconds following a selection, with performance peaking at 550 milliseconds. A two-phase gaze pattern was observed in which users exhibited high gaze motion immediately following errors, and then decreased gaze motion as the error was noticed. Together, these results provide the first demonstration that gaze dynamics can be used to detect input recognition errors, and open new possibilities for systems that can assist with error recovery.

CCS Concepts: **•Human-centered computing~Human computer interaction (HCI)~HCI theory, concepts and models**

Additional Key Words and Phrases: error mediation, eye tracking, virtual reality

ACM Reference format:

Candace E. Peacock, Ben Lafreniere, Ting Zhang, Stephanie Santosa, Hrvoje Benko, Tanya R. Jonker. 2022. Gaze as an Indicator of Input Recognition Errors. *Proc. ACM Hum.-Comput. Interact*, 6, ETRA, Article 142 (May 2022), 18 pages. https://doi.org/10.1145/3530883.

1 INTRODUCTION

The proliferation of wearable augmented and virtual reality (VR) technologies has resulted in a need for natural user input, such as through bare-hand midair gestures. To provide a seamless, positive input experience, gesture recognizers must be able to reliably distinguish intentional user input (e.g., a user performing a midair selection gesture) from all other user behavior (e.g., a user gesturing during conversation). As such, in recent years, the field of human-computer interaction has seen increasing interest in developing highly accurate and usable gesture recognizers and

Authors' addresses: C. Peacock, peacock.candace@gmail.com; B. Lafreniere, benlafreniere@fb.com; T. Zhang, tingzhang@fb.com; S. Santosa, ssantosa@fb.com; H. Benko, benko@fb.com; T. Jonker, tanya.jonker@fb.com.

2573-0142/2022/05 – Article#142... $15.00
https://doi.org/3530883

interaction techniques. A substantial body of research and development has made considerable gains to this end but, unfortunately, input recognition errors are difficult to eliminate completely, and when they do occur, they can significantly degrade the usability and experience of a system.

When an input recognizer fails to accurately discriminate intentional input, two types of system errors can occur: false positives, where the system recognizes "input" that the user did not intentionally perform; and false negatives, where the system fails to recognize input that was intentionally performed by the user. Both types of errors have been shown to degrade user experience, but false positive errors are particularly costly [24,28]; users are willing to tolerate more time spent correcting false negative errors to avoid false positive errors [24], and false positive errors require greater attentional resources to detect than false negative errors [24].

Given that it is difficult – if not impossible – to create an errorless gesture recognizer and given that false positive errors are particularly costly to the user experience, the present work investigates whether a system might detect false positive input recognition errors from user eye gaze in the moments after they occur, which could then be used to improve the input experience. For example, if a recognition system were able to detect when it had made an input recognition error (i.e., a false positive), it could use this information to refine and/or personalize its recognition model to reduce errors in the future. Even more compelling, if a system could detect errors soon enough after they occur, the system could directly assist the user with error recovery by communicating what had changed as a result of the unintended input and providing a convenient way to reverse the selection [25]. If the error detection was accurate enough, the system could even automatically undo the erroneous action. This capability has the potential to increase a user's trust in the input system and create richer human-computer interactions, more akin to how humans communicate with one another, where small mistakes or misunderstandings are seamlessly resolved in the flow of a conversation.

Gaze is a compelling modality for the detection of false positive input recognition errors because it has been shown to provide cues about fast, real-time changes in users' cognitive states [15,17,20,36], it is tightly coupled to manual selections [5,14,19], and it is sensitive to expectancy violations [22,26,31,43]. Gaze behavior has also been shown to vary following false positive errors in a variety of interaction tasks, including human robot interaction [2] and automation errors in the context of air traffic control [10]. Given the consistent relationship that exists between gaze and manual actions during interaction (i.e., gaze motion is correlated to hand motion) [5,14,19], it is reasonable that gaze might serve as a proxy for a user's intention to interact with a system [5,14,36] and therefore could be used to discriminate intentional input actions from false positive errors.

To provide a proof-of-concept that gaze dynamics are sensitive to false positive input recognition errors, a VR experiment was developed that mimicked common serial selection tasks (e.g., photo selection, movie selection). In the task, users searched through an array of tiles to find targets, which they were to select using a VR controller (i.e., user-initiated clicks). As they searched, the system would occasionally inject false 'clicks' to select a non-target item (i.e., injected clicks) thereby simulating input recognition errors. By examining gaze dynamics following *user-initiated clicks* versus *injected clicks*, the results revealed:

- Users exhibited distinct gaze patterns following user-initiated clicks versus injected clicks.
- Gaze dynamics exhibited a two-phase pattern, with (1) increased gaze motion immediately following injected clicks, which reflected attentional orienting to the next tile, and (2) decreased gaze motion later in time, which reflected noticing the error.

- Machine learned models discriminated user-initiated clicks from injected clicks almost immediately (at 50 milliseconds, .63 AUC-ROC) and detection accuracy increased with time (at 550 milliseconds, .81 AUC-ROC).

Together, these findings demonstrate the utility of gaze for false positive error detection and suggest that systems might be able to use gaze dynamics to rapidly detect errors. Importantly, these results focus on gaze *dynamics*, such as velocity and angular changes, which increases their generalizability to novel tasks and user interface layouts because they are not dependent on gaze-in-world coordinates. These findings have implications for the design of systems that can detect and correct input recognition errors or provide adaptive interfaces to help users recover from these errors. This work is future looking and does not pertain to current commercial products.

2 RELATED WORK

2.1 Reducing Input Recognition Errors

To reduce input recognition errors, prior work has explored *multimodal interaction techniques*, which enable users to leverage multiple modalities to complete their task. For example, Yu et al. explored coordinating gaze and hand motion to support object manipulation in VR [46], Pfeuffer et al. used gaze to select objects for pinching interactions in VR [38], and Kumar and Winograd used gaze in conjunction with key presses to control scrolling [27].

While the above-mentioned works incorporate gaze and an additional modality to explicitly control the interface, a similar but distinct vein of research has shown that gaze can be fused with other modalities to contextualize input. For example, combining gaze and gestural inputs reduces location errors when pointing at objects in VR [45], and differences between gaze position and touch position on touchscreens can be used to reduce system uncertainty by expanding a cursor's radius for pointing actions with reduced gaze guidance [42]. Beyond multimodal fusion sensing, gaze has also been used to anticipate explicit input, such as key presses on a touchscreen or in virtual reality. For example, gaze has been shown to produce consistent patterns leading up to and during manual selection [5,14,19], which has enabled user's interaction intention to be identified prior to the onset of input [14].

Other research has explored how the system can adapt its decision making based on recent user behavior as an alternative approach to reduce the costs of input recognition errors. For example, bi-level thresholding employs a restrictive threshold for all initial attempts at input to reduce false positives; once an initial attempt is made, if it does not surpass the conservative threshold, a more relaxed threshold is applied for a short time, allowing the user to generate a second attempt at input [24,34]. Although bi-level thresholding might reduce false positives, it does so by requiring users to repeat their input gestures, which is an unacceptable requirement in most situations.

These examples provide a snapshot of a large body of work, but they demonstrate approaches taken to sensing and thresholding of input recognition models. These methods are essential to improve accuracy, but they do not address the errors that are bound to occur even for reliable input recognizers. To mediate these negative effects, a system must sense errors, and yet to our knowledge there is an impoverished literature on the use of gaze to detect false positive errors.

2.2 Gaze and User Errors

Gaze is promising for the detection of input recognition errors because it is known to reflect cognitive states [15,17,20,36], it is tightly linked with manual selection [5,14,19], and it is sensitive to expectancy violations [22,26,31,43]. However, to date, much of the research linking gaze to errors explores *user-generated task errors*, such as clicking on the wrong button or failing to perceive changes in the visual field. This type of error is distinct from errors due to system failures, such as input recognition errors. For example, Ratwani et al. demonstrated that user-generated task errors are more likely to occur when users fixate less frequently and when they fail to fixate on task-relevant information [39]. Furthermore, Bovo et al. found that gaze was predictive of user errors during a block assembly task in which users manually assembled blocks as their gaze was tracked [8]. Finally, Xu et al. found that when users interacted with a tabletop touchscreen, the distance between gaze position and touch position was lower for intentional touch events compared to unintentional touch events, suggesting that these errors can be caused by a lack of attention to the user interface [44]. Although such studies demonstrated a link between eye movements and errors, they focused on user-generated errors, which are likely due to inattention, cognitive load, and/or interruptions [3,11,12,30,40]. The cognitive states and gaze behaviors that *produce* user-generated errors are likely different from the gaze behaviors that *arise* from input recognition errors, such as increased attentional resources to detect and correct errors [28].

2.3 Expectancy Violation and Gaze

The two prior sections review research on sensing and gaze prior to and during input events. Little work has focused on the distinct behavioral signatures *succeeding* input events. However, literature from cognitive science demonstrates that distinct gaze signatures occur following surprising, unexpected events. For example, when an incongruent object is placed in an environment (e.g., an octopus in a farmyard), people tend to produce more and longer fixations on that object compared to congruent objects placed in the environment (e.g., a tractor in a farmyard) [22,26,31,43]. Furthermore, after people learn motion sequences of disks, their gaze anticipates the next direction of rotation for the disk (expected change), as reflected by increases in gaze velocity [33,35]. When a novel motion direction is injected (unexpected change), gaze velocity decreases as subjects identify the unexpected change to orient their gaze in the actual direction of the disk's motion. Together, these findings suggest that there are markers of gaze that are sensitive to environmental inconsistencies and unexpected changes. Although this work does not explore gaze following input events, input recognition errors are indeed unexpected and surprising, and as such this literature suggests that we might expect gaze behaviors to be highly sensitive to unexpected input recognition errors during interaction.

2.4 Detecting Input Recognition Errors from Gaze

To date, there have been some early explorations into the use of EEG-based error potential and gaze to detect input recognition error [13,23,29]. Kalaganis et al. used electroencephalography (EEG) and gaze to detect errors in an eye-typing interface [23] and found that EEG produced a reliable error potential. However, users rarely moved their eyes following an error due to the design of the task, and as such the detection of input recognition errors was largely driven by EEG. Similarly, Bednarik et al. focused on the use of gaze dynamics to detect user input [5]. However, they used fixation behavior preceding and following the input event to detect the input event itself, making it difficult to determine if the model could detect errors based on gaze behavior following the input error alone. It is important to explore the utility of gaze following input recognition errors as systems might not always capture and process gaze due to computational and power constraints. A high efficiency system might only use gaze in the moments following an input event to determine whether that event was erroneous.

A handful of additional studies have provided more compelling evidence for the use of gaze for the detection of input recognition errors. For example, Aronson and Admoni (2018) demonstrated that gaze is useful for error recognition in human-robot shared manipulation [2]. Specifically, they found that when the robot misinterpreted the user's intent, users tended to look in atypical locations, such as the robot's elbow rather than its end effector. Similarly, Bruder et al. (2016) demonstrated that gaze is sensitive to automation errors in the context of air traffic control [10]. Here, fixations were more likely to be directed to relevant areas of interest when errors were detected. While these works demonstrate the utility of gaze to detect errors, the use of interest areas might not be feasible in a real-world system, as any framework that uses fixation-environment interactions as input to a model requires accurate calibration to track the focus of gaze along with computer vision models to determine task-relevancy of the fixated areas. Furthermore, an approach that relies on the content of areas-of-interest is less likely to generalize beyond a very specific interaction task and it neglects the rich information available in the temporal dynamics of gaze.

Although the literature suggests that gaze differs following errors (broadly defined) compared to accurate events, to our knowledge, it has not yet been demonstrated that a model could detect input recognition errors based on gaze dynamics alone.

3 STUDY SYSTEM

To explore whether gaze could be used detect false positive input recognition errors, a task was designed that (1) mimicked common serial selection tasks (e.g., photo selection, movie selection), (2) used a click gesture on the VR touchpad that produced plausible sensing errors, and (3) allowed for experimental control over extraneous factors surrounding user-initiated versus erroneously injected clicks, such as the error rate and timing of errors. Here, participants flipped over tiles and selected targets in VR while their eye movements were recorded. As participants flipped tiles, the system would occasionally select a non-target (i.e., an injected error). As such, the study system resulted in (1) *user-initiated clicks* that were produced by the user and (2) *injected clicks* that were introduced by the study system. The data were then analyzed to understand gaze patterns following injected and user-initiated clicks, and gaze-based models were trained to discriminate these events. This paper explores three hypotheses related to gaze and input recognition errors:

H1: Gaze features differ following user-initiated clicks and injected clicks.

H2: An individual model trained on gaze can discriminate between user-initiated and injected clicks for one user.

H3: A group model trained on gaze can discriminate between user-initiated and injected clicks for new users.

3.1 Methods

3.1.1 Participants, Apparatus, and Experimental Design

Thirty-two participants (M = 35 years, 13 females, 19 males) provided consent under a protocol approved by the authors' Institutional Review Board. Participants were screened to have normal or corrected-to-normal vision with contact lenses. One participant was excluded because they did not pass eye-tracker calibration (see below) and two participants were excluded because they received no injected clicks due to a bug in the code, resulting in a sample of 29 participants.

Eye and head movement data were collected using an HTC VIVE Pro Eye virtual reality headset and a MSI GS66 gaming laptop. The study was conducted remotely due to the COVID-19 pandemic; headsets and laptops were shipped to each participant's address. To ensure understanding of the task and activities, participants worked with research assistants via video conferencing to complete the hardware setup. The study interface consisted of a 3×3 grid of tiles (each tile was 0.35x0.35 m). Tiles were arranged such that the centers of neighboring tiles were 0.6 m part with 0.25 m of empty space between each tile. Participants were seated 3 m from the grid of tiles. However, the perceived tile size varied slightly as head motion was not controlled and people could lean backwards/forwards or adjust their chair.

Gaze data was logged at 60 Hz. Before the study, each participant completed a 9-point calibration. To ensure successful calibration within the task, participants were asked to fixate on a central tile for 60 seconds. If participants fixated on the central tile for at least 75% of the time and their gaze velocity was below 30° per second, then they were permitted to complete the study. If these criteria were not met, the calibration procedures were repeated. If after several attempts a participant's gaze could not be tracked, they were provided compensation and released from the study. As noted above, one participant could not be successfully calibrated.

The study employed a methodology similar to that used by Lafreniere et al. [28], but it was adapted to VR and tracked gaze. The study required participants to uncover and select target items using a ray-cast pointer, as demonstrated in Fig. 1 and the video. The pointer was enabled when participants rested their thumb on the controller touchpad. On each "page", six randomly selected tiles, located in a 3×3 grid, were enabled (i.e., indicated in yellow); the rest were grayed out and were not interactable. Participants searched through these six tiles for a specified number of target items (e.g., "Select [2] × [green circles]"). To reveal the contents of a tile, participants dwelled on the tile for 1.25 seconds while a circular progress indicator filled. Once the dwell time elapsed, the tile flipped over to reveal one of six icons: green circle, red heart, orange triangle, yellow star, blue moon, or purple plus sign. If the icon matched the target (e.g., a green circle), the participant was asked to select the tile by briefly breaking and then re-engaging contact between their thumb and the controller's touchpad (Fig. 1b, top). This "lift and recontact" gesture was chosen to make input errors more plausible, compared to pressing a button where errors would seem very implausible if they occurred when the user's thumb was not touching the button. If the tile was not selected within 1 second, the tile closed automatically. If selected, the tile would close 0.5 seconds after the click and click feedback would occur (as described below). Once the specified number of target items was selected, the next page was displayed.

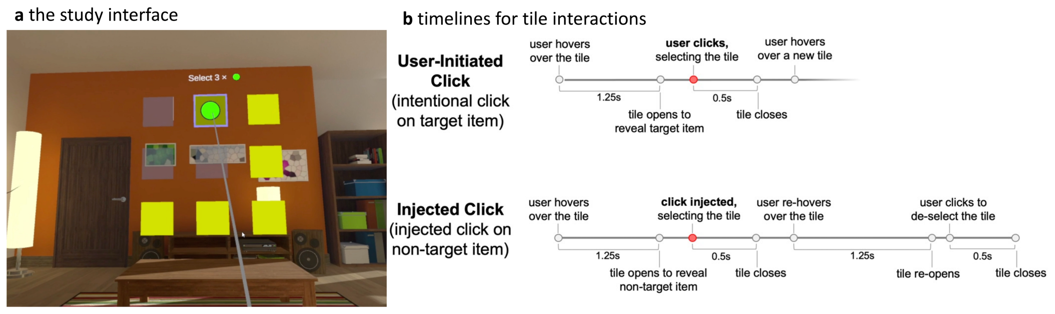
a the study interface
Select 3 ×
b timelines for tile interactions
User-Initiated Click
(intentional click on target item)
user hovers over the tile
user clicks, selecting the tile
user hovers over a new tile
1.25s
0.5s
tile opens to reveal target item
tile closes
Injected Click
(injected click on non-target item)
user hovers over the tile
click injected, selecting the tile
user re-hovers over the tile
user clicks to de-select the tile
1.25s
0.5s
1.25s
0.5s
tile opens to reveal non-target item
tile closes
tile re-opens
tile closes

Fig 1. A: The study interface. Participants were shown tiles and were to locate and select two or three hidden targets by dwelling to flip the tiles. The number of remaining targets was indicated at the top of the study interface. B: The timelines for tile interactions with user-initiated clicks (i.e., intentional, correct clicks on targets) and injected clicks (i.e., injected, erroneous clicks on non-targets).

When the participant uncovered a non-target icon, the system occasionally injected a click at a randomly selected time between 0.2 seconds and 0.5 seconds after the tile was opened or at the moment when the participant's ray-cast pointer left the tile, whichever occurred first (Fig. 1b, bottom). When an error was injected, the non-target item would appear to be selected and the click feedback would occur (as described below). To deselect an erroneously selected item, the participant needed to first re-open the tile and then click to deselect it. To ensure that the error was corrected, the participant was prevented from opening any other tiles until the non-target item was deselected.

When both user-initiated and injected clicks occurred, the border of a selected tile would turn blue, the ray-cast pointer would change to yellow and increase in thickness for 100ms, and a click sound would occur. To prevent rapid clicking, a 1.0 second lockout was imposed following each click. During this time, the ray-cast pointer would temporarily change to grey to communicate the lockout state. Visual feedback following user-initiated clicks and injected clicks was identical to ensure that there were no visual differences between the two conditions.

Each participant completed 12 "blocks" of the task described. A block consisted of 60 tile openings over several pages. Across all tile openings in a block, ~50% of the tile openings revealed target items; the rest revealed a randomly selected non-target item. A total of 9 injected clicks were injected for each block (9/60 tile openings; 15% of the time). Before each block, the target icon was communicated to the participant via text description (e.g., "The target item for this block is the green circle"). The order of the target items was counterbalanced across participants using a balanced Latin square. There were two practice blocks preceding the study. Participants practiced selected target icons in the first block and practiced deselecting icons when errors were injected in the second block.

3.1.2 Pre-Processing and Feature Extraction.

The 3D gaze vectors were transformed from the eye-in-head to eye-in-world direction using head orientation [10]. Next, the angular displacement between gaze samples, represented as normalized vectors u and v, $\theta = 2 \cdot atan2(\| u - v \|, \| u + v \|)$, were computed. Gaze velocity was computed as θ divided by the change in time between gaze samples.

The gaze data was then filtered to remove noise and unwanted segments before event detection and feature extraction. Data from the practice trials was discarded prior to analysis, and all gaze samples where the gaze velocity exceeded 800° per second were removed because they represented unfeasibly fast eye movements [16]. Any missing values were filled in using linear interpolation. Finally, a median filter with a window size of seven samples was applied to the gaze velocity signal to smooth the signal and account for noise prior to event detection [37].

Saccade detection was performed using I-VT on the filtered gaze velocities by identifying consecutive samples that exceeded 70° per second [41]. Minimum and maximum saccade durations of 17 to 200 milliseconds were enforced. Fixation detection was performed using I-DT by computing dispersion over time windows as the largest angular displacement from the centroid of gaze samples [41]. Time windows where dispersion did not exceed 1° were marked as fixations. Minimum and maximum fixation durations of 50 to 1500 milliseconds were enforced.

Ten features were chosen based on their predominance in past research. These features included fixation features (fixation duration, the angular displacement between fixation centroids), saccade features (the angular displacement between the current and previous saccade centroids, the angular displacement between the current and previous saccade landing points, saccade amplitude, saccade duration), and continuous features (fixation probability, saccade probability, gaze velocity, dispersion). The fixation and saccade features were used, as they are affected by incongruent scene information [22,26,31,43]. The continuous features were used, as they are sensitive to the onset of user input [14] and thus might be sensitive to the offset of input. The dispersion algorithm used the gaze samples corresponding to a lagging 1000 milliseconds to compute the dispersion of samples [36]. To represent these features as a continuous time-series, empty values between each event were linearly interpolated. Fixation probability and saccade probability were not z-scored because they were categorical; the remaining features were z-scored within participant.

3.2 Model: Gaze-based Input Recognition Error Detection

To determine whether gaze could be used to classify user-initiated versus injected clicks, logistic regression models were trained. To explore how quickly a system might detect an injected click, models were trained with varying time durations following a selection event, which is referred to as the *lens approach.* Here, gaze data from 50 to 600 milliseconds following each user-initiated or injected click was grouped into twelve 50 millisecond bins. Six hundred milliseconds was chosen as the maximum duration as it was the average amount of time that it took to select a new tile following a user-initiated click. Selection events that occurred at the end of a page were excluded because they were followed by additional graphical visualizations (tile shuffling) which may have elicited different gaze behaviors. Finally, user errors were quite rare (see Supplement); any user errors were excluded from the models.

Each time sample became a beta parameter in the model. For the 50 milliseconds lens size, there were three beta parameters for each feature since there were three time samples that occurred in the 50 milliseconds following error injection. The class weights were set to be inversely proportional to the number of samples for each class.

The first set of models were trained and tested for individual participants, which allowed the models to represent individual differences in gaze [1,21]. Individual models were trained on a random 80% of the data and tested on the remaining 20% of the data. A random 80% of the data was used because each selection was an independent event.

A second set of models (group models) were used to determine whether gaze behaviors that differentiate user-initiated clicks from injected clicks would be generalizable to new participants. Group models were trained using leave-one-participant-out cross-validation, i.e., models were trained on N-1 datasets and tested on the left-out dataset.

Performance was evaluated using the area under the receiver operator characteristic curve (AUC-ROC), which ranges from 0 to 1 with an AUROC of .5 indicates baseline classification by guessing [9]. The AUC-ROC values at a given lens size were compared to chance (.5) using a one-sample t-test. Comparisons of two AUC-ROC values for a given lens size were conducted using paired-samples t-tests. The false discovery rate (FDR) correction was used to control for multiple comparisons across the lens sizes for each feature [6].

4 RESULTS

4.1 Gaze Signatures After User-initiated versus Injected Clicks

The first analysis examined whether gaze differed following user-initiated and injected clicks using a statistical analysis. Here, the average value of each feature at each time point for each participant was computed for user-initiated versus injected clicks. These were then compared using a paired t-test to determine which time points were statistically different. This resulted in 36 paired t-tests (one for each time point between 17 to 600 milliseconds); the aforementioned

FDR correction was applied to control for multiple comparisons for each feature [6]. Significant differences were found for all features of the time-series data following user-initiated and injected clicks (Fig. 2).

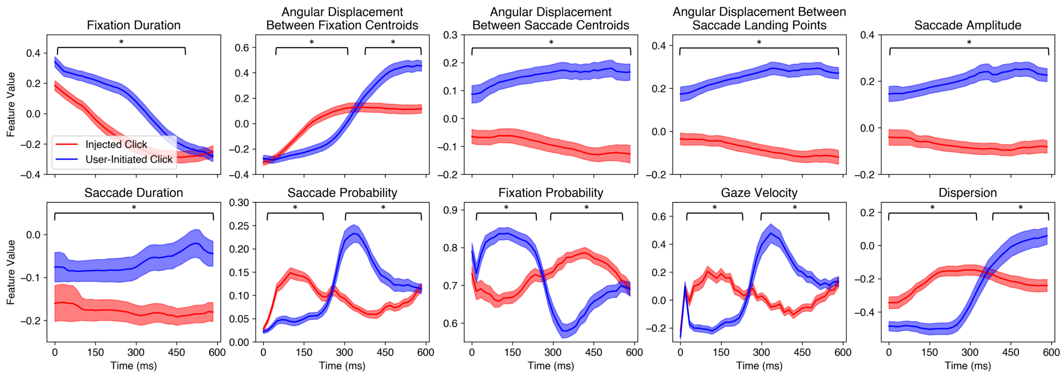

Fig 2. Time-series of gaze following the user-initiated clicks (blue) and injected clicks (red). Brackets and asterisks correspond to time points where the two conditions differed significantly. Error bands correspond to one standard error of the mean.

Visualization of the data suggests that there were two processes occurring, each at a different period of time. First, following injected clicks, participants' gaze was more likely to be in motion between 0 and 250 milliseconds, as evidenced by increased gaze velocities, dispersions, saccade probability, and angular displacement between fixation centroids, as well as decreased fixation probability. Given that these effects occur at the onset of the injected clicks, they suggest that these oculomotor behaviors were in progress at the moment of selection and that the participant was not attending to any particular object. A second pattern can also be observed between 300 and 450 milliseconds, where the pattern reverses. This second phase likely captured gaze behaviors related to noticing the error and recognizing the need to reorient attention to the target to correct the error. Together, these findings support the hypothesis that there are patterns of gaze that differ following user-initiated versus injected clicks (H1). For an analysis of user errors, see the Supplement, which depicts gaze following user-initiated clicks, injected clicks, and user errors (specifically, user-initiated non-target clicks).

4.2 Discriminating Between User-Initiated Clicks and Injected Clicks using an Individual Model

To test H2, models were trained and tested on the same participant to understand how well a personalized model might perform. One set of models used all ten features (Fig. 3a; Table 1) and another set used the ten features individually (Fig. 3b; Table 1). One sample t-tests indicated that individual models trained on all the features could reliably discriminate user-initiated from injected clicks for all lens sizes ($p < .05$; Fig. 3a). Models trained on individual features could discriminate user-initiated clicks from injected clicks well above chance for all lens sizes for each feature, with three exceptions: saccade amplitudes at 600 milliseconds and saccade durations at 150 and 600 milliseconds (Fig. 3; $p > .05$). Despite these exceptions, the individual feature results suggest that all gaze features were sensitive to error injections. Together, the results support the hypothesis that individual models can discriminate between user-initiated and injected clicks within milliseconds of an event (H2).

Table 1. The mean AUC-ROC for the individual and group models are depicted at 50ms and the lens size in which the AUC-ROCs were best.

	Individual Models		Group Models	
Model Name	AUC-ROC, 50ms	Best AUC-ROC, Lens Size	AUC-ROC, 50ms	Best AUC-ROC, Lens Size
All Features	.63	.81, 550ms	.63	.79, 450ms
Fixation Duration	.55	.66, 450ms	.54	.63, 500ms
Angular Displacement Between Fixation Centroids	.58	.72, 500ms	.55	.68, 400ms
Angular Displacement Between Saccade Centroids	.56	.64, 550ms	.57	.61, 450ms
Angular Displacement Between Saccade Landing Points	.58	.68, 550ms	.58	.64, 450ms
Saccade Amplitude	.54	.65, 550ms	.57	.61, 450ms
Saccade Duration	.56	.60, 500ms	.56	.59, 550ms
Saccade Probability	.53	.73, 350ms	.52	.74, 550ms
Fixation Probability	.54	.72, 400ms	.56	.72, 450ms
Gaze Velocity	.54	.76, 400ms	.56	.76, 550ms
Dispersion	.58	.77, 500ms	.57	.74, 600ms

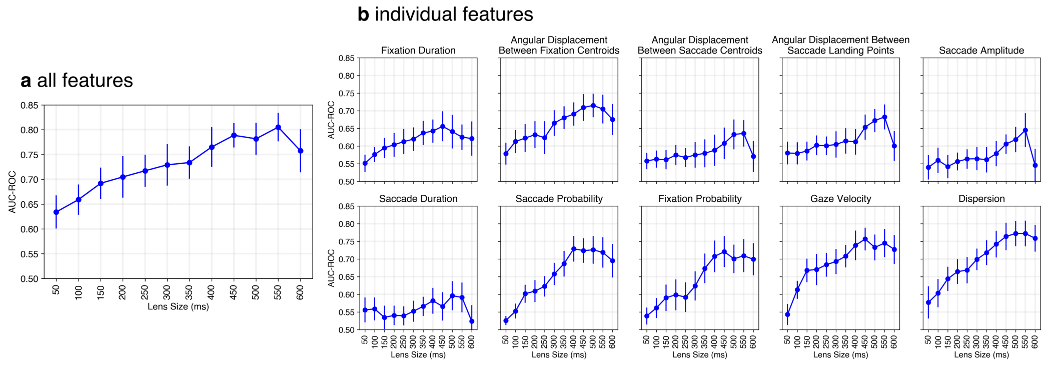

Fig 3. AUC-ROC scores from the individual models. Panel a depicts the mean AUC-ROC for a model trained on all ten features. Panel b depicts the mean AUC-ROC for models trained on each individual feature. Error bars depict 95% confidence intervals.

4.3 Discriminating Between User-Initiated Clicks and Injected Clicks using a Group Model

The individual models provided a compelling demonstration that gaze can be used to detect errors. However, if these individual models were used in practice, a system would require substantial ground truth data from each user before the model could be used. Alternatively, systems might deploy a group model if it functions well for new users, which would eliminate the need for users to undergo training before it could be used. To explore whether a group model could detect errors for new users, models were trained on N-1 participants and then tested on the held-out participant (Fig. 4; Table 1). The models that used all the features and each feature individually performed above chance for all lens sizes ($p < .05$). Together, these results suggest that a group model can detect errors for new users (H3).

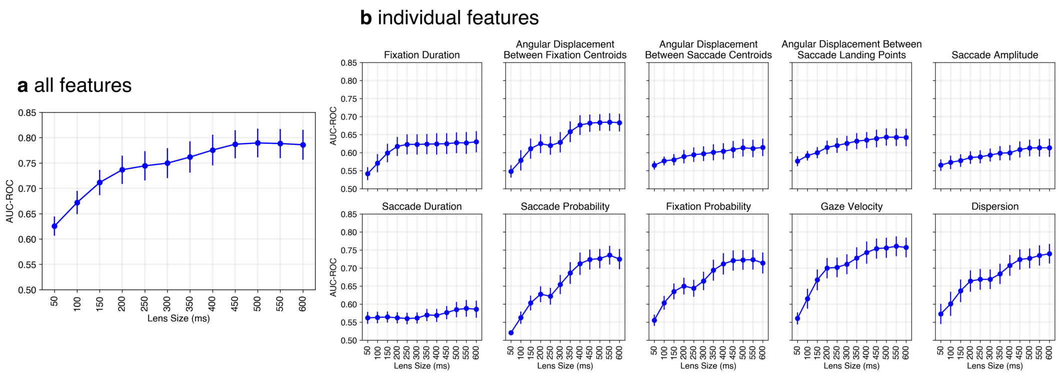

Fig 4. AUC-ROC scores from the group model. Panel A depicts the mean AUC-ROC for a model trained on all features. Panel B depicts the mean AUC-ROC for models trained on each individual feature. Error bars depict 95% confidence intervals.

4.3.1 Model Resiliency to Other Tasks and Interfaces.

The results provide a compelling demonstration of the use of gaze to detect input recognition errors. However, this was constrained to a single task, which raises questions about generalizability. To address this potential limitation, it was possible to use the existing study system to explore the extensibility of the present findings. Specifically, the model was trained only on instances that were followed by click feedback; model training excluded user-initiated clicks for the last target on a page because the post-selection user interface behaved differently — rather than providing click feedback, the tiles immediately reshuffled to generate the layout for the next page. In other words, the visuals of the user interface substantially differed for these terminal clicks; furthermore, when participants completed the final target selection, they were likely to move their eyes differently compared to a case where they were continuing their serial search for targets on the page.

As such, to test the generalizability of the model to new user interfaces, the trained group model was tested on these held-out terminal user-initiated clicks to determine how well the model performed even if the face of novel interface behavior. Overall, the model's performance when discriminating terminal user-initiated clicks from injected clicks was like that of the serial user-initiated clicks and injected clicks; there was no difference in the paired t-tests for any of the lens sizes ($p > .05$). This suggests that the model could discriminate between injected and user-initiated clicks regardless of whether the interface or task changed.

4.3.2 Influence of Time-based Versus Adaptive Error Injection.

After injected clicks, participants exhibited more eye movement compared to user-initiated clicks. However, it is possible that this was an artifact of our study design, which randomly injected clicks within 200 and 500 milliseconds of tile opening ("time-based injection") or whenever the participant's cursor left the bounds of the tile ("adaptive injection"). Given that gaze couples with hand movements [3,9,13,21], it is possible that this latter criterion of adaptive injection could have introduced a confound because injected clicks were more likely to co-occur when the cursor left the tile.

To address this concern, a follow-up study was run where clicks were injected based upon time alone (200 to 500 ms after a tile opened) with a sample of 10 participants from the first study (M = 35 years, 5 females, 5 males, 10 right-handed). By using a subset of the original participants, the follow-up could directly test whether gaze changed as a function of how errors were injected by comparing session performance for each individual.

Two group models were trained on all the features, one using the matched study participants from the first study and a second using the participants from the follow-up study using the same cross-validation procedure. Overall, paired t-tests showed no significant differences for the between the first and second study AUC-ROC values at all lens sizes ($p > 0.05$). This suggests that our results were not simply due to the mechanism by which errors were injected.

4.3.3 Model Performance of Individual Versus Group Models.

We next compared the group to the individual models to determine whether a group model could be used in a system that had not yet been personalized. Overall, there were no significant differences in the all-feature model performance for the group and individual models ($p > .05$) (Fig. 5a). This is somewhat of a surprising outcome, given individual differences in eye behavior [1,21], and could be explained by differences in the amount of training data. This was investigated further by computing learning curves for the group (Fig. 5b) and individual models (Fig. 5c). Inspection of the learning curves suggested that the individual models would likely perform better with more training data.

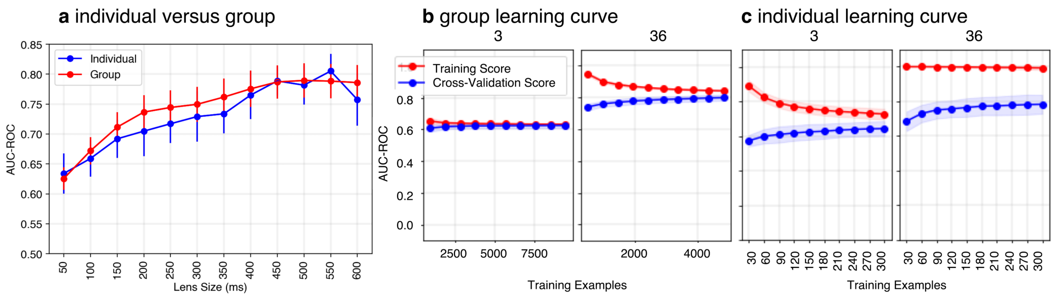

Fig. 5. Group and individual model comparisons. Panel a: The red and blue lines correspond to the group and individual model AUC-ROC at each lens size, respectively. Error bars correspond to 95% confidence intervals. The group (b) and individual (c) model learning curves for the smallest (3) and largest lens size (36). The red and blue lines correspond to the training and cross-validation scores, respectively. Error bands for (b) and (c) reflect one standard deviation.

5 DISCUSSION

Prior work has explored how gaze can be used to control or contextualize user input (see Related Work). However, to date, none have explored whether gaze following input events could be used to distinguish input recognition errors from user-initiated selections. If an input recognition system were able to rapidly detect false positive input recognition errors, it could then employ mediation techniques to clarify the user's intentions and assist with error recovery or use this signal to improve and personalize the input recognizer to avoid future errors. To enable future systems with this capability, the present work explored whether gaze dynamics following click events could be used to identify input recognition errors.

The results demonstrated that gaze dynamics can be used to detect false positive input recognition errors. A logistic regression model was able to discriminate injected clicks from user-initiated clicks as early as 50 milliseconds after their occurrence (.63 AUC-ROC) and achieved peak performance at 550 milliseconds following the input event (.81 AUC-ROC), demonstrating how quickly a system might be able to detect these input recognition errors with reasonable confidence. Importantly, models trained on numerous features performed best, but models trained on gaze velocity alone performed nearly as well. Gaze velocity is a simple feature that represents changes in the gaze vector from the preceding time point to the current one, and as such it does not require substantial feature engineering to be used in real-time systems. Furthermore, given that gaze velocity is independent of the spatial coordinates of gaze, it is expected to be useful even when user interfaces vary in their layout and design. In other words, it is a promising feature for the generalizability of the proposed model, which is discussed in more depth below.

An important finding for the applicability of our work to real systems is that a group model achieved comparable performance to individual models. The high performance of the group models demonstrates that personalization to an individual user is not required to provide this capability "out of the box". Although it is likely that models would improve with personalization, the results suggest that a group model can identify errors well above chance.

We also discovered a two-phase gaze pattern following the click event (Fig. 2). Following injected clicks, there was increased eye motion between 0 and 250 milliseconds. At the moment of an injected click, participants are likely already orienting their attention to the next tile because they do not intend to select the non-target tile, which manifests in high gaze velocity and a greater likelihood of saccades. This finding is consistent with prior work, which has found that eye movements are used for motor planning [5,14,19]. Second, between 300 and 450 milliseconds, there was a decrease in gaze motion following an injected click. This is the timeframe in which participants have likely detected the click feedback. In other words, during this time, they have abandoned their current plan and begun to reorient their gaze back to the erroneously selected object to correct it. From the present study, it is unclear whether this second phase occurs only when users correct input errors in a user interface. Although it would require future studies to determine whether the second phase occurs irrespective of the correctability of the error, prior research on expectancy violation suggests that it might [22,26,31,43]. For example, after people learn motion sequences, their gaze velocity increases as they anticipate what direction a disk will move next; however, when a novel, unexpected motion direction is injected, gaze velocity decreases as subjects identify the change to orient their gaze in the actual direction of the disk's motion [26,28]. The two-phase pattern is likely something that models could learn to detect to improve user experience. For instance, future modeling work might develop models that could represent each of these two phases separately (e.g., as states), which would allow systems to understand whether users have noticed errors once selection feedback is given.

Together, these results demonstrate that input systems could deploy models based upon gaze dynamics to detect input recognition errors and adapt the input detection system to improve user experience. This model expands prior work that uses interactions between gaze and areas of interest to predict errors [2,10] by instead using gaze dynamics, which are more likely to generalize to new tasks and contexts. More work is needed to explore future directions, but the results provide a compelling demonstration for future hardening of gaze-based input recognition error detectors.

5.1 Design Implications for Input Recognizer Systems

The ability to detect recognizer errors rapidly opens several new possibilities for the design of input systems, including the development of improved input recognizers and the design of error mediation techniques.

Detection of input recognition errors can provide valuable information for refining and personalizing an input recognition model, which would improve its accuracy in the future. For example, gaze-based detection of false positive errors could be combined with detection of false negative errors using a heuristic method similar to that used in the bi-level thresholding work [24,34] to give the recognizer system a comprehensive "error awareness". When an error is detected, the system might capture the raw data around the error event (e.g., recorded video or other inputs used by the recognizer) to improve and/or personalize its input recognition model. These data can be computationally expensive to capture and store on an ongoing basis, and the labeling of ground truth can be complex. The model framework introduced here could be used to capture false positive events and label them, decreasing the computational and storage load on the system.

Second, the ability to detect false positive errors opens the possibility of interactive error mediation techniques that could be deployed to assist the user with error recovery, or otherwise reduce the impact of these errors on user experience. While there is precedent for mediation techniques that interactively disambiguate a user's intentions before the system finalizes an action (e.g., through prompts or n-best lists) [18,32], here, the detection of errors occurs after the input event. In other words, the model proposed here is driven by the system detecting its error from the user's gaze, which unlocks a distinct and novel space for design. Drawing on prior literature, there are several ways in which error mediation techniques could assist the user. For example, the display might highlight the change to application state that has resulted from a potential false positive error, a method similar to Phosphor [4]. Critical to successful error communication is to draw the user's attention to the spatial location where change has occurred, perhaps by displaying a summary of the change(s), as in work on mnemonic rendering [7]. In addition to helping users notice the error, the system could assist by providing a convenient way to 'undo' the changes. An important consideration when designing these techniques is that the error detection model may itself produce errors, which could lead the system to assist with "recovering" from intentional selections. To this end, it will be critical for future work to test whether the recognition accuracy of the present models is acceptable to users in real interaction systems and how model accuracy interacts with various types of error mediation (e.g., auto-undo versus confirmation). Investigating the design space for potential error mediation techniques and studying the effects of such techniques on user trust, experience, and other factors are all interesting areas for future research that are opened by the findings presented here.

5.2 Generalizability

The intent of this work was to provide a proof-of-concept that gaze could be used to detect input recognition errors, but future work must explore how well these outcomes generalize to other interfaces, tasks, and input recognizers. The tile task used in the present work was designed to represent a broad set of search-and-select interactions. In the tile task, a user must sequentially inspect several items (i.e., performing a serial targeting task) and then provide distinct input to select desired targets; when errors occurred, they had to correct them. Given the design of the task and the use of target-agnostic gaze features (e.g., gaze velocity), the results observed here would likely be similar in many other serial targeting tasks. In other words, the fundamental demands of this task are representative of other point-and-select tasks in human-computer interaction.

Second, the present work opted to use gaze dynamics as input features rather than spatial coordinates or gaze heatmaps. Dynamic features, such as fixation duration, gaze velocity, and angular displacements capture relative changes in gaze over time and are thus independent of the specific spatial coordinates of gaze. As such, these features are less dependent on the spatial layout of the user interface and any directional components of scanning. Future work might deploy both, exploring both spatial features of the interface (e.g., target size, target density) and dynamic gaze features, particularly when a task-specific model is desirable.

Finally, there is the question of whether the results will generalize across different recognition-based input techniques. In the present study, clicks were injected rather than using an input recognizer because this allowed for control over extraneous factors surrounding user-initiated clicks versus erroneously injected clicks. However, it is likely that any given recognizer would have certain contexts where errors would be more likely to occur (e.g., an IMU-based recognizer for midair gestures would be more susceptible to errors when the user is moving their hands quickly). This is relevant to our error detection paradigm because gaze signatures that discriminate true selections from input recognition errors are likely to vary based on when in an action the error occurred. Furthermore, the results could change depending on the type of interaction technique (e.g., whether one uses a thumb pad versus point-and-click interactions). Overall, our analyses demonstrated that there was discriminating signal at each time point, from the click onset, which suggests that this modeling framework could be successfully applied to recognizers with errors at different timings. Ultimately, the modeling demonstrated here was a proof-of-concept that can be applied to novel interfaces, tasks, and recognizers. The present approach will function best when it is tuned to a specific interface and input recognizer.

5.3 User Errors Versus Input Recognition Errors

As reviewed in the Related Work section, user errors and input recognition errors occur for different reasons. Two interesting questions, then, are (1) whether the gaze dynamics following user-initiated non-target clicks are different than injected clicks, and (2) how a system might respond to these two types of errors. In the Supplement, we demonstrate a clear distinction in gaze behavior between user-initiated non-target clicks and injected clicks. Although this exploration was not the intention of our study, it suggests that it is possible for a system to discriminate between these two error types, which introduces a novel research space that could explore how a system might respond to these errors. We leave this research space open for future work on gaze and input error mediation.

5.4 Limitations

There are several limitations that should be addressed in future work. First, the injected click errors used might have affected users' reaction to those errors. For example, because errors were only injected to non-target items, the injected clicks may have been more predictable than they would be in real interaction contexts. In fact, if errors were more predictable, the observed effects are likely weaker than would be expected in real interaction contexts. As reviewed in the Related Works, people produce distinct gaze patterns to unexpected events. As such, if input recognition errors are less predictable in real interaction contexts, then one might expect a stronger distinct gaze reaction to these errors. This is of course speculation; it is necessary for future work to explicitly evaluate this.

A second limitation of the present work is a focus on only one type of error: false positive input recognition errors. There are many other types of errors, including finger slippage in which a target item is selected before an intentional act, which could surprise a user and cause atypical gaze behavior even though the user might not in fact want to correct the "error". Furthermore, future work may wish to focus on false negative input recognition errors, as it is unknown whether they will produce the same gaze dynamics as false positive input recognition errors.

Finally, this research used a very simple logistic regression model. This enabled easy inspection and interpretation of the gaze features, but it is possible that there are important patterns in the time-series of gaze that it was not able to represent. For example, perhaps the transition from high velocity to a sharp directional change in gaze is a unique signature of noticing an error. The current architecture is limited in its ability to represent relative changes in gaze behavior. Future work on gaze for error detection should examine various time-series and latent state models (e.g., LSTM) and the performance gains that can be made with these models.

6 CONCLUSION

The present research demonstrates that input recognition errors can be detected moments after they occur using gaze dynamics alone. This work introduces an entirely new approach to solving the challenge of imperfect input recognizers. That is, rather than improving the sensing model, the present work provides a promising proof-of-concept that systems could leverage gaze dynamics immediately following an input event to detect errors and provide effortless error mediation techniques. This approach has the potential to transform interaction through more intelligent recognition systems that can operate before, during, and following input events to provide the best possible input experience.

ACKNOWLEDGMENTS

We would like to thank Dan Clarke and Vivien Francis for the development of the data collection environment and for their assistance with software support. We would also like to thank Taylor Bunge, Chip Connor, Nour Shoora, and Joseph Montgomery for implementing the remote data collection infrastructure. Finally, we thank Michelle Annett and Kash Todi for their helpful comments.

REFERENCES

[1] Nicola C. Anderson, Fraser Anderson, Alan Kingstone, and Walter F. Bischof. 2015. A comparison of scanpath comparison methods. Behav Res 47, 4 (December 2015), 1377–1392. DOI:https://doi.org/10.3758/s13428-014-0550-3.

[2] Admoni Aronson and Henny Admoni. 2018. Gaze for error detection during human-robot shared manipulation. In RSS Workshop: Towards a Framework for Joint Action.

[3] Brian P. Bailey and Joseph A. Konstan. 2006. On the need for attention-aware systems: Measuring effects of interruption on task performance, error rate, and affective state. Computers in Human Behavior 22, 4 (July 2006), 685–708. DOI:https://doi.org/10.1016/j.chb.2005.12.009

[4] Patrick Baudisch, Desney Tan, Maxime Collomb, Dan Robbins, Ken Hinckley, Maneesh Agrawala, Shengdong Zhao, and Gonzalo Ramos. 2006. Phosphor: explaining transitions in the user interface using afterglow effects. In Proceedings of the 19th annual ACM symposium on User interface software and technology (UIST '06), Association for Computing Machinery, New York, NY, USA, 169–178. DOI:https://doi.org/10.1145/1166253.1166280

[5] Roman Bednarik, Hana Vrzakova, and Michal Hradis. 2012. 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 Press, Santa Barbara, California, 83. DOI:https://doi.org/10.1145/2168556.2168569

[6] Yoav Benjamini and Yosef Hochberg. 1995. Controlling the False Discovery Rate: A Practical and Powerful Approach to Multiple Testing. Journal of the Royal Statistical Society 57, 1 (1995), 289–300. DOI:https://doi.org/10.2307/2346101

[7] Anastasia Bezerianos, Pierre Dragicevic, and Ravin Balakrishnan. 2006. Mnemonic rendering: an image-based approach for exposing hidden changes in dynamic displays. In Proceedings of the 19th annual ACM symposium on User interface software and technology (UIST '06), Association for Computing Machinery, New York, NY, USA, 159–168. DOI:https://doi.org/10.1145/1166253.1166279

[8] Riccardo Bovo, Nicola Binetti, Duncan P. Brumby, and Simon Julier. 2020. Detecting errors in pick and place procedures: detecting errors in multi-stage and sequence-constrained manual retrieve-assembly procedures. In Proceedings of the 25th International Conference on Intelligent User Interfaces (IUI '20), Association for Computing Machinery, New York, NY, USA, 536–545. DOI:https://doi.org/10.1145/3377325.3377497

[9] Andrew P. Bradley. 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition 30, 7 (July 1997), 1145–1159. DOI:https://doi.org/10.1016/S0031-3203(96)00142-2

[10] Carmen Bruder, Paul Weber, and Catrin Hasse. 2016. To look and (not) see: predicting the detection of automation failures based on the eye movements of human operators. In Proceedings of the International Conference on Human-Computer Interaction in Aerospace, ACM, Paris France, 1–7. DOI:https://doi.org/10.1145/2950112.2964579

[11] Michael D. Byrne and Susan Bovair. 1997. A Working Memory Model of a Common Procedural Error. Cognitive Science 21, 1 (1997), 31–61. DOI:https://doi.org/10.1207/s15516709cog2101_2

[12] Michael D. Byrne and Elizabeth M. Davis. 2006. Task Structure and Postcompletion Error in the Execution of a Routine Procedure. Hum Factors 48, 4 (December 2006), 627–638. DOI:https://doi.org/10.1518/001872006779166398

[13] Ricardo Chavarriaga, Pierre W. Ferrez, and José del R. Millán. 2008. To Err is Human: Learning from Error Potentials in Brain-Computer Interfaces. In Advances in Cognitive Neurodynamics ICCN 2007, Springer Netherlands, Dordrecht, 777–782. DOI:https://doi.org/10.1007/978-1-4020-8387-7_134

[14] Brendan David-John, Candace E Peacock, Ting Zhang, T Scott Murdison, Hrvoje Benko, and Tanya R Jonker. 2021. Towards gaze-based prediction of the intent to interact in virtual reality. In ACM Symposium on Eye Tracking Research and Applications, Virtual Event, Germany, 7. DOI:https://doi.org/10.1145/3448018.3458008

[15] Gabriel Diaz, Joseph Cooper, Dmitry Kit, and Mary Hayhoe. 2013. Real-time recording and classification of eye movements in an immersive virtual environment. Journal of Vision 13, 12 (October 2013), 5–5. DOI:https://doi.org/10.1167/13.12.5

[16] Stefan Dowiasch, Svenja Marx, Wolfgang Einhäuser, and Frank Bremmer. 2015. Effects of aging on eye movements in the real world. Front. Hum. Neurosci. 9, (2015). DOI:https://doi.org/10.3389/fnhum.2015.00046

[17] Anjith George and Aurobinda Routray. 2016. Real-time eye gaze direction classification using convolutional neural network. In 2016 International Conference on Signal Processing and Communications (SPCOM), 1–5. DOI:https://doi.org/10.1109/SPCOM.2016.7746701

[18] Tovi Grossman and Ravin Balakrishnan. 2006. The design and evaluation of selection techniques for 3D volumetric displays. In Proceedings of the 19th annual ACM symposium on User interface software and technology (UIST '06), Association for Computing Machinery, New York, NY, USA, 3–12. DOI:https://doi.org/10.1145/1166253.1166257

[19] Mary M Hayhoe, Anurag Shrivastava, Ryan Mruczek, and Jeff B Pelz. 2003. Visual memory and motor planning in a natural task. Journal of Vision 3, 6 (2003), 49–63. DOI:https://doi.org/10.1167/3.1.6

[20] Hao He, Yingying She, Jianbing Xiahou, Junfeng Yao, Jun Li, Qingqi Hong, and Yingxuan Ji. 2018. Real-Time Eye-Gaze Based Interaction for Human Intention Prediction and Emotion Analysis. In Proceedings of Computer Graphics International 2018 on - CGI 2018, ACM Press, Bintan, Island, Indonesia, 185–194. DOI:https://doi.org/10.1145/3208159.3208180

[21] John M. Henderson and Steven G. Luke. 2014. Stable individual differences in saccadic eye movements during reading, pseudoreading, scene viewing, and scene search. Journal of Experimental Psychology: Human Perception and Performance 40, 4 (2014), 1390–1400. DOI:https://doi.org/10.1037/a0036330

[22] John M. Henderson, Phillip A. Weeks, and Andrew Hollingworth. 1999. The effects of semantic consistency on eye movements during complex scene viewing. Journal of Experimental Psychology: Human Perception and Performance 25, 1 (1999), 210–228. DOI:https://doi.org/10.1037/0096-1523.25.1.210

[23] Fotis P. Kalaganis, Elisavet Chatzilari, Spiros Nikolopoulos, Ioannis Kompatsiaris, and Nikos A. Laskaris. 2018. An error-aware gaze-based keyboard by means of a hybrid BCI system. Sci Rep 8, 1 (September 2018), 13176. DOI:https://doi.org/10.1038/s41598-018-31425-2

[24] Keiko Katsuragawa, Ankit Kamal, Qi Feng Liu, Matei Negulescu, and Edward Lank. 2019. Bi-Level Thresholding: Analyzing the Effect of Repeated Errors in Gesture Input. ACM Trans. Interact. Intell. Syst. 9, 2–3 (April 2019), 1–30. DOI:https://doi.org/10.1145/3181672

[25] Mohamed Khamis, Ludwig Trotter, Markus Tessmann, Christina Dannhart, Andreas Bulling, and Florian Alt. 2016. EyeVote in the wild: do users bother correcting system errors on public displays? In Proceedings of the 15th International Conference on Mobile and Ubiquitous Multimedia (MUM '16), Association for Computing Machinery, New York, NY, USA, 57–62. DOI:https://doi.org/10.1145/3012709.3012743

[26] Kathryn Koehler and Miguel P Eckstein. 2016. Scene Inversion Slows the Rejection of False Positives through Saccade Exploration During Search. In Proceedings of the 37th Annual Meeting of the Cognitive Science Society, 6.

[27] Manu Kumar and Terry Winograd. 2007. Gaze-enhanced Scrolling Techniques. In ACM Symposium on User Interface Software and Technology, Newport, Rhode Island, 4. DOI:https://doi.org/10.1145/1240866.1241036

[28] Ben Lafreniere, Tanya R Jonker, Stephanie Santosa, Mark Parent, Michael Glueck, Tovi Grossman, Hrvoje Benko, and Daniel Wigdor. 2021. False Positives vs. False Negatives: The effects of recovery time and cognitive costs on input error preference. In 34th ACM Symposium on User Interface Software and Technology. DOI:https://doi.org/10.1145/3472749.3474735

[29] Moritz Lehne, Klas Ihme, Anne-Marie Brouwer, Jan B.F. van Erp, and Thorsten O. Zander. 2009. Error-related EEG patterns during tactile human-machine interaction. In 2009 3rd International Conference on Affective Computing and Intelligent Interaction and Workshops, 1–9. DOI:https://doi.org/10.1109/ACII.2009.5349480

[30] Simon Y W Li, Anna L Cox, Ann Blandford, Paul Cairns, Richard M Young, and Aliza Abeles. 2006. Further Investigations into Post-completion Error: The Effects of Interruption Position and Duration. In Proceedings of the Annual Meeting of the Cognitive Science Society, 7.

[31] G R. Loftus and Normah H Mackworth. 1978. Cognitive determinants of fixation location during picture viewing. Journal of Experimental Psychology: Human Perception and Performance 4, 4 (1978), 565–565. DOI:https://doi.org/10.1037/0096-1523.4.4.565

[32] Jennifer Mankoff, Scott E. Hudson, and Gregory D. Abowd. 2007. Interaction techniques for ambiguity resolution in recognition-based interfaces. In ACM SIGGRAPH 2007 courses (SIGGRAPH '07), Association for Computing Machinery, New York, NY, USA, 11-es. DOI:https://doi.org/10.1145/1281500.1281522

[33] Jessica Maryott, Abigail Noyce, and Robert Sekuler. 2011. Eye movements and imitation learning: Intentional disruption of expectation. Journal of Vision 11, 1 (January 2011), 7–7. DOI:https://doi.org/10.1167/11.1.7

[34] Matei Negulescu, Jaime Ruiz, and Edward Lank. 2012. A recognition safety net: bi-level threshold recognition for mobile motion gestures. In Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services (MobileHCI '12), Association for Computing Machinery, New York, NY, USA, 147–150. DOI:https://doi.org/10.1145/2371574.2371598

[35] Abigail Noyce, Jessica Maryott, and Robert Sekuler. 2010. Unexpected events, predictive eye movements, and imitation learning. Journal of Vision 10, 7 (August 2010), 755–755. DOI:https://doi.org/10.1167/10.7.755

[36] Candace E Peacock, Brendan David-John, Ting Zhang, T Scott Murdison, Hrvoje Benko, and Tanya R Jonker. 2021. Gaze Signatures Decode the Onset of Working Memory Encoding. In EMICS '21: ACM CHI '21 Workshop on Eye Movements as an Interface to Cognitive State, Yokohama, Japan, 6.

[37] Jami Pekkanen and Otto Lappi. 2017. A new and general approach to signal denoising and eye movement classification based on segmented linear regression. Scientific Reports 7, 1 (December 2017), 17726. DOI:https://doi.org/10.1038/s41598-017-17983-x

[38] Ken Pfeuffer, Benedikt Mayer, Diako Mardanbegi, and Hans Gellersen. 2017. Gaze + pinch interaction in virtual reality. In Proceedings of the 5th Symposium on Spatial User Interaction (SUI '17), Association for Computing Machinery, New York, NY, USA, 99–108. DOI:https://doi.org/10.1145/3131277.3132180

[39] Raj M. Ratwani, J. Malcolm McCurry, and J. Gregory Trafton. 2008. Predicting postcompletion errors using eye movements. In Proceeding of the twenty-sixth annual CHI conference on Human factors in computing systems - CHI '08, ACM Press, Florence, Italy, 539. DOI:https://doi.org/10.1145/1357054.1357141

[40] Raj M. Ratwani, J. Gregory Trafton, and Christopher Myers. 2006. Helpful or Harmful? Examining the Effects of Interruptions on Task Performance. Proceedings of the Human Factors and Ergonomics Society Annual Meeting 50, 3 (October 2006), 372–375. DOI:https://doi.org/10.1177/154193120605000334

[41] Dario D Salvucci and Joseph H Goldberg. 2000. Identifying Fixations and Saccades in Eye-Tracking Protocols. In Proceedings of the 2000 Symposium on Eye tracking research and applications, 71–78. DOI:https://doi.org/10.1145/355017.355028

[42] Baris Serim and Giulio Jacucci. 2016. Pointing while Looking Elsewhere: Designing for Varying Degrees of Visual Guidance during Manual Input. In Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI '16), Association for Computing Machinery, New York, NY, USA, 5789–5800. DOI:https://doi.org/10.1145/2858036.2858480

[43] Melissa L.-H. Võ and John M. Henderson. 2009. Does gravity matter? Effects of semantic and syntactic inconsistencies on the allocation of attention during scene perception. Journal of Vision2 9, 3 (2009), 1–15. DOI:https://doi.org/10.1167/9.3.24

[44] Xuhai Xu, Chun Yu, Yuntao Wang, and Yuanchun Shi. 2020. Recognizing Unintentional Touch on Interactive Tabletop. Proc. ACM Interact. Mob. Wearable Ubiquitous Technol. 4, 1 (March 2020), 33:1-33:24. DOI:https://doi.org/10.1145/3381011

[45] Suparat Yeamkuan and Kosin Chamnongthai. 2021. 3D Point-of-Intention Determination Using a Multimodal Fusion of Hand Pointing and Eye Gaze for a 3D Display. Sensors 21, 4 (January 2021), 1155. DOI:https://doi.org/10.3390/s21041155

[46] Difeng Yu, Xueshi Lu, Rongkai Shi, Hai-Ning Liang, Tilman Dingler, Eduardo Velloso, and Jorge Goncalves. 2021. Gaze-Supported 3D Object Manipulation in Virtual Reality. In Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems, ACM, Yokohama Japan, 1–13. DOI:https://doi.org/10.1145/3411764.3445343

Received November 2021; revised January 2022; accepted April 2022.