Detecting Input Recognition Errors and User Errors using Gaze Dynamics in Virtual Reality

Naveen Sendhilnathan
naveensn@fb.com
Reality Labs Research, Meta
Redmond, USA

Ting Zhang
tingzhang@fb.com
Reality Labs Research, Meta
Redmond, USA

Ben Lafreniere
benlafreniere@fb.com
Reality Labs Research, Meta
Toronto, Canada

Tovi Grossman
tovi@dgp.toronto.edu
Department of Computer Science,
University of Toronto
Toronto, Canada

Tanya Jonker
tanya.jonker@fb.com
Reality Labs Research, Meta
Redmond, USA

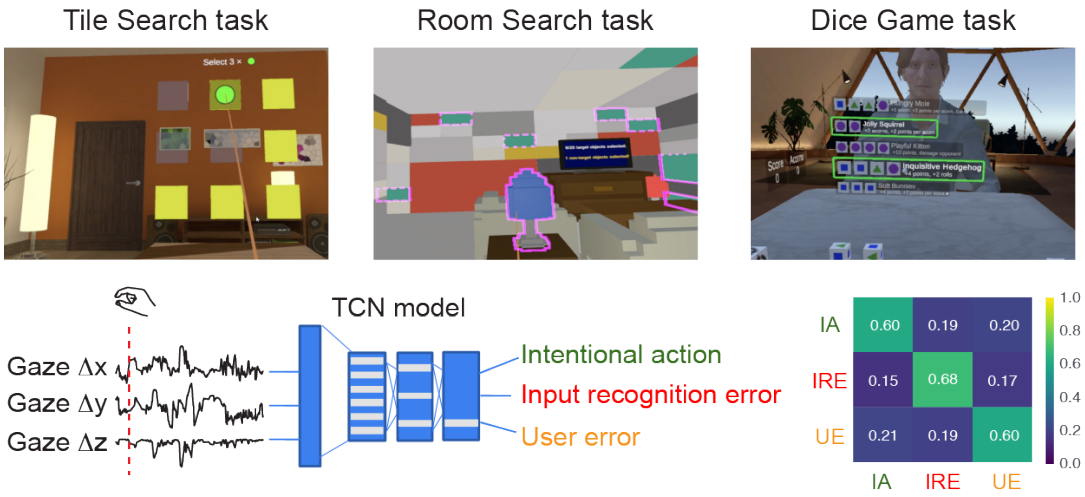

Figure 1: An overview of the error detection model. A deep learning model was trained on three distinct VR tasks (top) using users' natural gaze dynamics (bottom left) to classify input events into three classes in the moments after they occur: intentional actions, input recognition errors, and user errors. A confusion matrix for the model trained and tested on data from all three tasks showed above-chance (0.33) performance for the three classes (bottom right). Such error type information could be used by future interaction systems to refine their model for better personalization or assist the user with error recovery, resulting in low friction interactions.

ABSTRACT

Gesture-based recognition systems are susceptible to input recognition errors and user errors, both of which negatively affect user experiences and can be frustrating to correct. Prior work has suggested that user gaze patterns following an input event could be used to detect input recognition errors and subsequently improve interaction. However, to be useful, error detection systems would need to detect various types of high-cost errors. Furthermore, to build a reliable detection model for errors, gaze behaviour following these errors must be manifested consistently across different tasks. Using data analysis and machine learning models, this research examined gaze dynamics following input events in virtual reality (VR). Across three distinct point-and-select tasks, we found differences in user gaze patterns following three input events: correctly recognized input actions, input recognition errors, and user errors. These differences were consistent across tasks, selection versus deselection actions, and naturally occurring versus experimentally injected input recognition errors. A multi-class deep neural network successfully discriminated between these three input events using only gaze dynamics, achieving an AUC-ROC-OVR score of 0.78. Together, these results demonstrate the utility of gaze in detecting interaction errors and have implications for the design of intelligent systems that can assist with adaptive error recovery.

UIST '22, October 29-November 2, 2022, Bend, OR, USA

ACM ISBN 978-1-4503-9320-1/22/10…$15.00
https://doi.org/10.1145/3526113.3545628

CCS CONCEPTS

• **Human-centered computing → Gestural input**; **Empirical studies in HCI**; **Mixed / augmented reality**.

KEYWORDS

Recognizer error, input recognition errors, gaze behavior, gaze dynamics, eye tracking, adaptive user interfaces

ACM Reference Format:
Naveen Sendhilnathan, Ting Zhang, Ben Lafreniere, Tovi Grossman, and Tanya Jonker. 2022. Detecting Input Recognition Errors and User Errors using Gaze Dynamics in Virtual Reality. In *The 35th Annual ACM Symposium on User Interface Software and Technology (UIST '22), October 29-November 2, 2022, Bend, OR, USA.* ACM, New York, NY, USA, 19 pages. https://doi.org/10.1145/3526113.3545628

1 INTRODUCTION

With the growing adoption of head-mounted augmented reality (AR) and virtual reality (VR) technologies, there is an increasing interest in AR and VR input technologies that eliminate the need for handheld input controllers and enable more natural interactions with digital content, such as mid-air, hand-based gesture input. To achieve this, however, a system must deploy input recognizers that distinguish intentional user input to a system (e.g., a mid-air selection gesture) from other user behavior that is directed at the real world and not intended for interaction with a system (e.g., pointing someone in the direction of the local coffee shop). Prior approaches to solve this problem have deployed mode switching [35, 64, 71], but these techniques require time, memory, and effort from users. To enable low friction, intuitive interaction with AR/VR systems, gesture recognizers need to be highly accurate and activate without any required effort from users, and they must discriminate intentional system-directed gestures from other world-directed actions. However, even with the rapid development of accurate and usable gesture recognizers and recognition techniques in recent years, input recognition errors are still difficult to completely eliminate and continue to significantly degrade user experiences [41].

Input recognizer errors are not the only errors that degrade user experiences. Users themselves can also produce errors. A user error occurs (1) when a user makes a selection that an input recognizer correctly detects, but the selected object was task-irrelevant, leading to an unexpected or a negative outcome, or (2) when a user rapidly changes their mind about their selection [42]. These errors are frustrating and costly to users [42] – especially if they cannot be reversed – but they cannot be eliminated by improving an input recognizer because the system correctly recognized the gesture produced by the user. Common approaches to reducing user errors include interaction techniques that prevent errors [31, 68] and techniques that assist users in recovering from errors [2, 5, 44, 56]. Although these methods substantially improve one's interaction experience, they are not a universal solution and must be designed anew for each application and user interface. Given the prevalence [8, 9, 24, 66] and costly [41] nature of input recognizer and user errors, and the lack of a universal solution for handling errors, it would be valuable for systems to be able to *detect* whether a recent input event was an input recognition error or user error and use this information to adapt its models and/or support interaction in an error-type specific manner. For example, if a system could detect erroneously recognized input events, it could use those instances to refine and personalize its recognizer models. Similarly, if a system could detect input recognition errors and user errors, it could provide adaptive error recovery techniques to assist users with error correction. Although the detection of probable errors for model improvement and mediation is compelling and could transform input experiences, such systems are not currently prevalent.

To pave the way for such solutions, one must first demonstrate the feasibility of detecting recent input recognition errors and user errors. Prior work by Peacock et al. introduced the use of a gaze-based classification model to detect false positive input recognition errors and user-intended true positive input events [52]. However, in that study, errors were artificially injected by a study system and the model was developed using data from one highly controlled task. The present work seeks to generalize this result to other error types and input events that occur in point-and-select tasks, and test two hypotheses:

- *H1: Input Event Discrimination*: Distinct types of input events and errors will provoke different temporal patterns of gaze dynamics, enabling gaze to be used to discriminate between these events.
- *H2: Cross-Task Consistency*: The temporal patterns of gaze dynamics following a given type of input event will manifest consistently across different scenarios and tasks.

The present work seeks to confirm these hypotheses, to enable the creation of machine learning models that can reliably discriminate between different types of input events and errors across different scenarios and tasks, which in turn could be used to deploy appropriate interventions as needed. To this end, the paper makes the following contributions:

- An analysis of three independent point-and-select datasets which demonstrated that 1) a gaze-based system could *discriminate* different types of input events after their occurrence (i.e., intentional actions, input recognition errors, and user errors) and 2) temporal patterns of gaze features *consistently* manifested themselves after the three input events across three tasks.
- A multi-class deep learning model that took users' natural gaze dynamics following an input event as input and classified an event as an intentional action, input recognition error, or user error, with an AUC-ROC-OVR of 0.78 across three tasks.

Together, these findings demonstrate the potential of using gaze dynamics to identify different types of input events and have implications for the design of future intelligent, error-aware systems that can detect when input recognition errors and user errors have occurred. Such systems could use such information to improve recognition models or take corrective actions to help users recover from errors.

2 RELATED WORK

The present work builds on prior research on input errors, input recognition techniques, and the use of gaze to improve input recognition.

2.1 Input Errors

Broadly speaking, input errors can be classified into two categories, user errors and input recognition errors. User errors occur when the user provides input to a system that does not advance their current objective, such as clicking the wrong button in an interface. In contrast, input recognition errors occur when a system misinterprets (i) whether a user is intending to provide input to a system or not or (ii) the specifics of the input that the user is attempting to provide.

Two main types of system errors can occur in gesture-based input, i.e., *false negatives* and *false positives*. False negatives occur when a system fails to recognize intentional input provided by a user, resulting in no input to the system. False positives occur when a system mistakes non-input behavior for intentional input, resulting in unintended input to the system. Both error types have the potential to degrade user experiences, but false positive errors have been shown to be particularly costly for user experiences, in part because they occur unexpectedly and impose an attentional cost to notice and fix [41]. The present work is focused on building models to detect and distinguish false-positive system errors from user errors and intentional input. We do not address the problem of false negative errors, for which heuristic detection methods have been developed in prior work [36, 49].

Prior work has also shown that user errors can be caused by momentary cognitive deficits such as increased cognitive load, a lack of attention, deficits in working memory, or interruptions [1, 7, 13, 14, 54, 55]. User errors can also result from momentary motor function deficits such as overshooting or undershooting a target [25, 26]. These errors are much more difficult to recognize and reduce because they are driven by failures in a user's behavior and cognition, rather than failures in a system's models. To date, the primarily mechanism for addressing user errors has been to design interfaces such that actions are easily reversible, for example through undo or history mechanisms [2]. The present work explores a new approach to address user errors by determining whether they can be detected using gaze behavior following their occurrence.

2.2 Improving Input Recognition

Input recognition errors have been shown to have negative effects on user experiences [41, 49]. The most common approach to address these errors is to improve the accuracy of sensing systems or enhance the gesture recognition techniques that are used to recognize input based on sensor data [48, 58, 59]. Typically, the recognition method used for a given input technique is fixed, but some work has explored dynamically adjusting a recognizer to create a more optimal user experience. With bi-level thresholding, for example, a system imposes a conservative threshold for recognition to reduce false positive errors but relaxes this threshold following false negatives to allow the user to succeed when trying to perform a gesture a second time [36, 49]. Although bi-level thresholding can help reduce false positives, it can lead to more false negatives, which may not be appropriate for some applications, such as fast-paced games in VR.

In addition to refining input recognition for a given modality, prior work has explored combining multiple modalities to obtain a more accurate idea of a user's intentions. Researchers have, for example, combined gaze with hand motion to support object manipulation in VR [75], combined gaze with key presses to control scrolling [40], and combined gaze with gestural input to reduce location errors while pointing at objects in VR [73]. These approaches use additional sensing modalities *prior to* selection to improve input detection.

While many promising approaches have been developed to reduce recognizer errors, it is unlikely that they can be eliminated completely. An alternative approach has been to design systems that are aware of when they have made recognition errors, so that the system might assist a user in recovering from such errors [62]. In this vein, recent work by Peacock et al. showed that gaze dynamics could be used to differentiate user initiated input actions from input recognition errors (simulated by injecting clicks) as early as 50 milliseconds after their occurrence, with accuracy peaking at 550 milliseconds [52]. While these were encouraging initial results, Peacock et al.'s study had several limitations. First, they only studied one task, leaving open the question of task generalizability. Second, false positive input errors were simulated using injected button clicks, leaving it unclear whether their results would generalize to the false positive errors that naturally occur with gesture recognizers. Third, they only studied one type of error, i.e., input recognition errors (using a two class model) and therefore, it is unclear if gaze can be used to detect other types of errors such as user errors. The present work addresses these open questions, establishing that gaze-based input recognition error detection is generalizable across the task used by Peacock et al. and for two new tasks, including one in which false positive errors naturally occurred in a pinch-based gesture recognition system.

2.3 The Utility of Gaze in Detecting User Interaction

Natural gaze behavior has been shown to reflect internal cognitive and motor states [17, 23, 28, 51] and to be sensitive to reward. For example, the saccade vigor (i.e., peak velocity as a function of amplitude), is greater if the stimulus toward which an eye movement is being made is associated with a reward [57, 60]. Gaze behavior has also been found to have distinct patterns for goal-oriented and non-goal oriented behavior [63]. As an example, gaze is known to be tightly linked to manual selection (e.g., pointing) [10, 16, 27]. That is, prior research has shown that hand movements are tightly linked with eye movements [3, 47]. In particular, eye movements precede hand movements by a few milliseconds [20, 21, 67] and thus patterns of eye gaze behavior are a good predictor of future hand motion [74].

Furthermore, people have been found to fixate for longer on incongruent objects (e.g., an octopus in a farm scene) than congruent objects in an environment (e.g., a scarecrow in a farm scene), implying that there are differences in gaze behavior that depend on environmental inconsistencies [29, 38, 43]. Related to this, gaze behavior has also been shown to reflect expectation violations [19, 50]. Given that gaze is an indicator of overt attention [11], where gaze locations are consistent with one's location of attention, gaze also can be used to index object saliency [33] during feature attention [12]. Together, this prior work has demonstrated that gaze is typically coupled with attention and that it is sensitive to surprising and

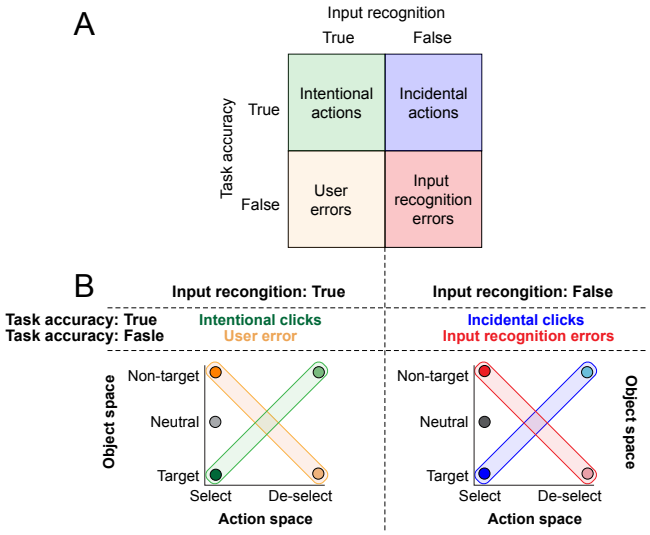

Figure 2: A map of an input event. (A) A 2x2 space of input events that occur when an input event is detected by a recognizer. The event space has two axes: input recognition accuracy and task accuracy. (B) Unpacking of the general input event space described in (A), adapted to the tasks in this study. First, input events were classified based on whether the system recognized the user's intended action (left panel) or if the system recognized an action that was not performed by the user (right panel). Actions were classified based on whether that action was a selection or a deselection (i.e., undo), and whether the object was a target, non-target, or a neutral space.

unexpected events in the environment. As such, gaze is a promising signal for the detection of recent errors, both from the input recognizer and the user themselves.

Recent studies have also looked into other forms of implicit gaze interactions. For example, changes in a user's gaze patterns can indicate the onset of cybersickness [72]. User gaze patterns can also be used to efficiently manage visual content, overload, and clutter [22, 65] which is increasingly becoming an important aspect of AR. There has also been an increased interest in gaze guidance in XR environments [39, 46]. Furthermore, combined with information from the surrounding facial muscles and pupillometry, gaze information could be used to create an empathic connection between remote collaborators [45]

In summary, the reviewed literature suggests that gaze dynamics might be useful when distinguishing between different types of input events. However, the closest related work, i.e., Peacock et al. [52], only looked at distinguishing injected clicks from intentional clicks for a particular task. The present work explores the use of gaze behavior to distinguish input events in greater depth, examining three classes of input events (i.e., intentional actions, input recognition errors, and user errors) across three distinct tasks.

3 DATA ANALYSIS AND MODELING

3.1 Input event space

This study aimed to identify and analyze gaze dynamics patterns for intentional actions and different types of errors that occur during VR-based point-and-select interaction. These input events can be conceptualized and represented as a 2x2 space with input recognition accuracy and task accuracy as the two axes, leading to 4 possible outcomes of a detected input event: (Fig. 2).

- *Intentional actions*, where a user performs intentional correct actions and a system recognizes these actions correctly, leading to a positive outcome.
- *Incidental actions*, where a user did not provide an intentional input but a system falsely recognized an action, leading to a positive outcome (e.g., a system selected a task-relevant object but the user did not initiate this action).
- *User errors*, where a user performed intentionally wrong actions (e.g., by clicking on a task-irrelevant object) which a system recognized correctly, thus leading to a negative outcome.
- *Input recognition errors*, where a user did not provide an intentional input but a system falsely recognized an action, leading to a negative outcome (e.g., a system selected a task-irrelevant object but the user did not initiate this action).

This framework can be further unpacked to apply to the datasets considered in this research (as described in the next section) by transforming the input recognition and task accuracy axes into object space and action space axes (Fig. 2B). At the level of the object space, the object that was selected or deselected could be a target, a non-target, or a selection in an empty space that did not have any object at all (i.e., "neutral"). When considering those objects that a user can interact with, a user's action could be to select an object or to deselect an already selected object (to undo a prior selection). Although unpacking this framework is specific to the tasks considered in this paper, it is extensible to other object and actions spaces as well. For example, in other situations, the action space could have two different actions, such as left versus right click, or even *n* different actions.

3.2 Datasets

Three datasets with different types of input systems (i.e., a handheld controller with clicks vs a pinch gesture recognizer), varying levels of experimental constraints, and task demands were used for the data analysis to test the two hypotheses. These datasets were composed of eye-tracking data in VR from point-and-selection tasks and contained input recognition errors and user errors. These datasets were previously generated from three prior and independent studies and were used here with permission from the authors of those studies.

Our aim was to test the two hypotheses (i.e., *input event discrimination* and *cross-task consistency*) for gaze dynamics while intentional actions, user errors, and naturally occurring false positive-input recognition errors occurred during mid-air gesture based interactions using a realistic interaction task. However, instead of studying a less constrained system with a gesture recognizer, we first studied the gaze dynamics during a simplified, well-controlled

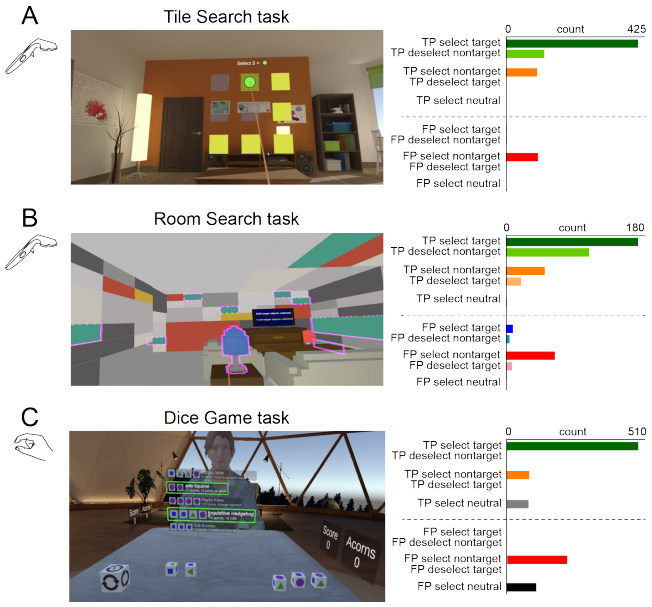

Figure 3: Datasets and the distribution of input events per dataset. (A) Left: Screenshot of the Tile Search task. The mode of input is represented using an icon to the left (in this case, a hand-held controller); Right: The distribution of events (as mapped in Fig. 3A) for the Tile Search task. TP refers to true positives, which occur when the input recognition is correct, and FP refer to false positives, which occur when the input recognition is incorrect. See Fig. 2B. (B) Left: Screenshot of the Room Search task; Right: The distribution of events for the Room Search task. (C) Left: Screenshot of the Dice Game task. The mode of input is represented using an icon to the left (in this case, a hand gesture); Right: The distribution of events for the Dice Game task.

task (*Tile Search Task*) where the input recognition errors were experimentally injected. These results were then compared to those from a less constrained task that had experimentally-injected input recognition errors (*Room Search Task*). Finally, the investigation was expanded to a more realistic task (i.e, *Dice Game Task*) using mid-air gestures, where input recognition errors occurred naturally.

3.2.1 Tile Search Task. The Tile Search task (Fig. 3A) required participants to uncover and select target objects in a grid, using a ray-cast pointer from an HTC VIVE controller. On each "page", a 3 x 3 grid of tiles was displayed. Six of the tiles were randomly selected and colored yellow, whereas the other three were grayed out and could not be interacted with. Participants searched through the six yellow tiles for a specified number of target objects (e.g., "Select [2] × [green circles]"). To reveal the contents of a tile, participants dwelled a ray cursor on the tile for 1.25 seconds while a circular progress indicator filled. The tile then flipped to reveal one of six icons: a green circle, a red heart, an orange triangle, a yellow star, a blue moon, or a purple plus sign. If the icon matched the target (e.g., a green circle), the participant would need to try to select the tile by briefly breaking and then re-engaging contact between their thumb and the controller's touch pad. If the tile was not selected within 1 second, the tile flipped back over automatically to hide the object. If selected, click feedback was provided (described below) and the tile would close (i.e., flip back over) 0.5 seconds after the click. Once the specified number of target objects was selected, a new page of objects was displayed.

When the participant flipped open a tile to reveal 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 exited the boundaries of the tile, whichever occurred first. When an error was injected, the system would act as though the user had performed a click, selecting the tile with the non-target object and providing identical feedback to a user-initiated click. To deselect an erroneously selected object, the participant needed to first re-open the tile and then click to deselect it. To ensure that the error was corrected before moving on, the participant was prevented from opening any other tiles until the non-target object was deselected.

The dataset contained interaction and eye-tracking data for 31 participants who completed this task as part of Peacock et al.'s investigation of gaze behavior as an indicator of input recognition errors [52]. Among them, 12 identified themselves as female and 19 identified themselves as male. The mean age of all participants was 35 years. For each participant, there was data for 12 "blocks" of the task described above, each consisting of 60 tile openings over several pages. Each block contained 9 injected clicks (i.e., 9/60 tiles; 15% error rate per tile opening).

This task was composed of the three types of input events we set out to investigate: correct intentional actions, input recognition errors, and user errors. Therefore, we used this dataset to test *H1 (Input Event Discrimination)*. That is, we explored if the gaze dynamics had different temporal patterns after different types of input events.

3.2.2 Room Search Task. The Room Search task required participants to search through a multi-room virtual environment to find and select 25 objects of a specific target color (i.e., green, yellow, or red), while not selecting objects of non-target colors. Selectable objects included 3D objects in the rooms (e.g., furniture, lamps), and rectangular tiles of various sizes covering the walls, floor, and ceilings. An HTC VIVE controller was held in the participant's dominant hand and objects were selected using a ray-cast pointer controlled by hovering the ray cursor over the object and clicking using a button on the controller. Participants could also navigate through the environment using a standard VR teleportation technique that was invoked using a second HTC VIVE controller that was held in the participant's non-dominant hand.

The inclusion of tiles on the walls, floor, and ceiling of the environment created a situation where the ray cursor was nearly always hovering over a selectable object. On a random schedule, the system would inject false positive clicks, which would select or deselect the currently hovered object. The schedule for click injections was generated at the start of the trial, such that 20 clicks would be injected over 5 minutes. Injected clicks acted identically to user-initiated clicks, toggling the selection state of the object being hovered over and providing identical feedback to user-initiated

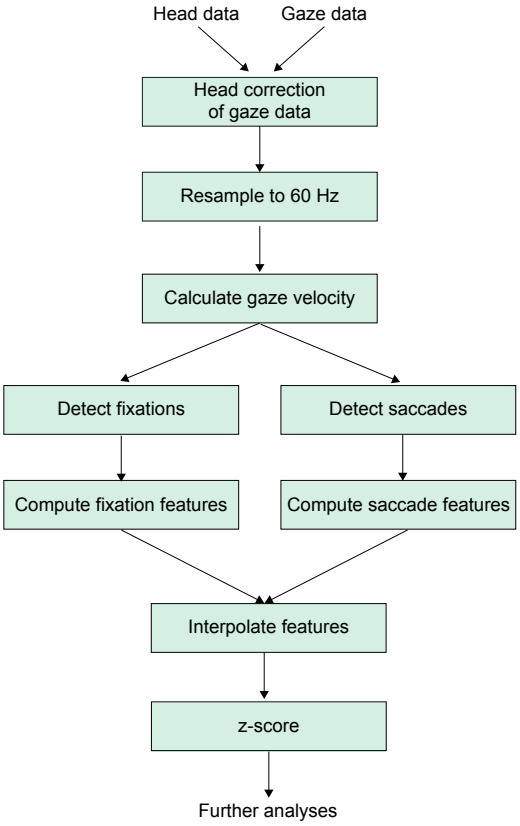

Figure 4: The pipeline used to pre-process the three datasets.

clicks. Unlike the Tile Search task, participants were not required to correct erroneous selections of non-target objects immediately, though they were not able to finish a block while non-target objects were selected.

The dataset contained interaction and eye-tracking data for 19 participants (i.e., 8 of them identified as female, 9 as male, 1 as non-binary, and 1 preferred not to disclose their gender). The mean age of all participants was 37 years. Each participant completed 6 blocks of the task.

3.2.3 Dice Game Task. Different from the Tile Search and Room Search tasks, the dataset for the Dice Game task consisted of input events triggered by a hand gesture recognizer through an IMU-based, wristwatch-driven pinch sensing system similar to Wen et al. [69]. Here, false positive input recognition errors were caused due to the imperfect recognition instead of experimental injections. The participant wore a ring-styled device on the thumb and the index finger similar to ElectroRing [37] to collect the ground truth for pinch selections. And they also wore an HTC Vive Pro Eye HMD with a hand tracker puck [32] whose raycast pointer was used for pointing.

This Dice Game task was a VR version of a Yahtzee-style dice game [70]. The goal was to gain more points than the computer opponent within a 3-min timed block. There are 12 blocks in total and the participant and the computer took turns within each block. Depending the speed of the participant, the number of turns in one block varied. The participant started the game by clicking on the "roll dice" button. In front of the participant, there was a panel displaying the points that they could collect if their dice were rolled into that combination. After the dice were rolled, the participants could "lock"/"unlock" any dice by clicking on it to aggregate it towards the desired combination. They could click on the "roll dice" button 3 times maximum in a turn. Once the combination was achieved, the participant could click on that combination shown in the panel in front of them to collect the points.

This dataset contained interaction and eye-tracking data for 18 participants who completed this task as part of investigation of hand movement behavior as an indicator of intent to interact in [76]. Each session had around 60 minutes of data for each individual. The mean age of all participants was 34 years. Seven of them identified themselves as female and 11 identified themselves as male.

3.3 Pre-processing and Feature Extraction

A common, reusable data pre-processing pipeline was designed and applied across all three datasets (Fig. 4). As such, the extracted features were consistent and comparable across datasets.

First, the raw 3D gaze position vectors were transformed from the eye-in-head frame of reference to an eye-in-world direction using head orientation data [15]. The data was then re-sampled to 60 Hz to circumvent irregular gaze data sampling and to keep the sample rate consistent across participants and datasets. Next, gaze velocity was computed as the angular displacement between consecutive gaze samples divided by the change in time between gaze samples. All gaze samples where the gaze velocity exceeded 800°/s were removed because they represented very fast eye movements that were unfeasible [18]. Any missing values were then linearly interpolated. Next, saccades and fixations were detected. Saccades were detected by performing IN-VT on the filtered gaze velocities by identifying consecutive samples that exceeded 70° [61]. A minimum duration of 17 milliseconds and maximum duration of 200 milliseconds were used for saccades. Fixations were detected by performing I-DT, which computed the dispersion over time windows as the largest angular displacement from the centroid of the gaze samples [61]. Time windows where dispersion did not exceed 1° were labelled as fixations. A minimum duration of 50 milliseconds and maximum duration of 1500 milliseconds were used for fixations.

After detecting the saccades and fixations, 10 features [16] were extracted from the data based on their predominance in past research and the literature [52]. Fixation features included fixation probability, the fixation duration, and the angular displacement between consecutive fixation centroids. Saccade features included the saccade probability, the angular displacement between the current and previous saccade centroids, the saccade amplitude, the saccade duration, and the angular displacement between the current and previous saccade landing points. Gaze velocity and gaze dispersion were also included. To represent these features as a continuous time-series, the value for each sample point was set as the value of the feature from the most recent saccade or fixation event, and carried forward in time until the event was next detected. Empty values were linearly interpolated between events. Next, concurrent

behavioral data including information on the trial, task conditions, etc., were used to label the feature time series with the type of input events. Finally, all of the features were z-scored within-participant, except for the fixation probability and saccade probability, which were used for further analyses.

3.4 Statistical Analysis

One of the aims of the present work was to examine if gaze dynamics after the onset of intentional input, input recognition errors, and user errors had different temporal patterns. To compare the time series data across these classes, an appropriate statistical test was performed on the time series data, for each time point (i.e., t-test to compare two time series or an ANOVA to compare three time series). This resulted in 37 statistical tests for each gaze feature (i.e., 600 milliseconds with 60 Hz rate). Multiple comparisons were corrected for using the false discovery rate (FDR) method. If the corrected p-value was less than 0.05, that time-bin was labelled as "significant".

3.5 Modeling Approach

We wanted to develop a gaze-based machine learning classifier to classify input events into three classes (i.e., intentional actions, input recognition errors, and user errors) by capturing patterns of gaze behavior over time rather than just single time point differences. Therefore, we treated this problem as a time-series classification problem to discriminate between the three classes.

Thus, a temporal convolutional network (TCN) was used to classify these three input events. TCN is a member of the convolutional neural network (CNN) family, which contains convolutional kernels. These kernels have been proven to be effective in learning discriminating features [6], which are crucial for a classification problem, in contrast to a recurrent neural network (RNN) architecture which is more often used to predict a value in a time series [34]. As a complex architecture could lead to overfitting when a sample size is not large enough, TCN employs dilated convolution layers, which enables it to learn longer sequential patterns while maintaining a simple network structure [6]. Therefore it has significant advantages compared to other more complex CNN-based time series classification models, such as fully convolutional neural networks (FCNs) or ResNets that lack this feature.

Specifically, this research used the TCN model architecture defined by Bai et al. [6], which contained 4 dilated convolutional layers. Because convolutional kernels are able to learn discriminating features from raw data, the first order difference of the x, y, z head-corrected eye position data was used as input, instead of the gaze features. The time window 0 - 650 milliseconds after an event's onset was extracted from the original time series as the input sample. The output was one of three classes: intentional action, user error, or input recognition error.

Two types of modeling approaches were used. First, a model was trained on 70% of the data and tested on the remaining 30% of the data for each individual participant, which allowed the models to represent individual differences in gaze features. However, this approach led to models trained on data from all participants, and thus these models were not participant-independent. Second, we trained a separate set of models in a participant-independent manner. Here, the models were trained on 70% of the participants' data and tested on 30% of the participants' data, where the data of a participant belonged to either the training or the testing set, but not both.

As the number of samples for each class were highly imbalanced (Fig. 3), the class weights for both types of models were balanced by setting the weights to be inversely proportional to the number of samples for each class. Five-fold cross validation was also performed on the training data (on both sets of models) to minimize the chance of overfitting to a particular block of data.

3.6 Evaluation Metrics

Confusion matrices were used to ensure balanced performance across the classes. The Area Under the Curve (AUC) of the Receiver Operator Characteristic (ROC) curve for one-vs-rest (AUC-ROC-OVR) was used to quantify model performance. Given that this was a 3-class classification problem, AUC-ROC-OVR evaluated 3 binary class classifications (i.e., one-vs-rest). Specifically, 3 AUC-ROC curves were constructed, one for each class. These curves were compared against the other two classes and then averaged into a single score. AUC-ROC-OVR has the additional benefit of being computationally more efficient than AUC-ROC-OVO, which constructs a different AUC-ROC curve for every possible binary combination of the three classes and then averages all of them. The AUC-ROC-OVR scores ranged from 0 and 1, with larger values indicating better model performance. The baseline was 0.5, which represented the chance level performance of a no-skill classifier.

4 RESULTS

In the present work, we explored whether users' gaze dynamics 1) could be used to discriminate intentional actions, input recognition errors, and user errors moments after they occurred and 2) if they are consistent across different point-and-select tasks, action types, and across injected and real input recognition errors. This would enable us to build a machine learning model to classify these input events based on gaze dynamics alone.

4.1 Gaze Features had Different Temporal Dynamics After Different Input Events

To date, gaze patterns following input recognizer false positive errors and user errors have not been systematically explored. As such, we tested *H1* by exploring whether gaze dynamics following an input event differed for input recognition errors versus user errors. For this analysis, the data from the Tile Search task was used because the input recognition errors were experimentally injected at random times on non-target objects only, meaning this was a well-constrained task suitable for this analysis.

First, the average of each feature value aligned to the input event onset was computed for each time point after the onset of the input event for each participant. A one-way ANOVA was then performed to determine which time points were statistically different for each feature. Multiple comparisons were corrected for using the FDR method. Significant differences ($p < 0.05$ from ANOVA after FDR correction) were found for all features for the three input events (Fig. 5). We also performed post-hoc Tukey HSD tests on each time point for pairs of conditions (see Fig. A1 for example). Together,

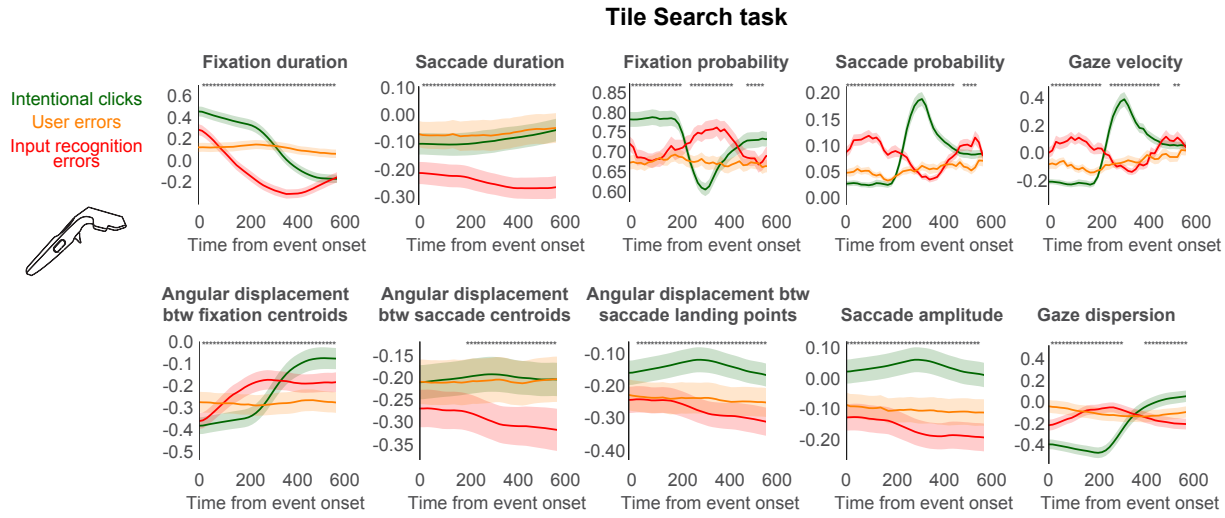

Figure 5: Visualizations of the time series of z-scored gaze features following intentional correct actions (green), input recognition errors (red), and user errors (orange) for the Tile Search task. Asterisks (*) correspond to data in the time series where the three time series significantly differed (corrected $p < 0.05$). Data is represented as Mean ± SEM

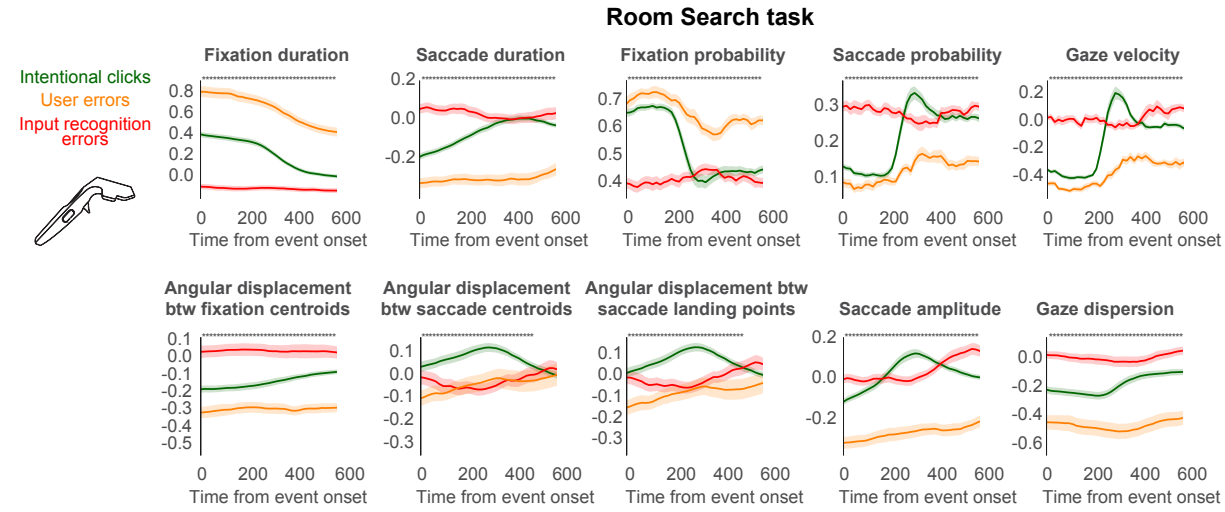

Figure 6: Visualizations of the time series of z-scored gaze features following intentional correct actions (green), input recognition errors (red), and user errors (orange) for the Room Search task. Asterisks (*) correspond to the data samples in the time series that differed significantly (corrected $p < 0.05$). Data is represented as Mean ± SEM.

these findings suggest that gaze features had different temporal patterns following different input events.

4.2 Temporal Dynamics of Gaze Features Were Consistent Across Different Tasks

The finding that gaze dynamics differed for intentional actions, input recognition errors, and user errors demonstrates that gaze dynamics can be a promising signal to use to detect various types of errors. However, to demonstrate the potential of this approach for error detection in working interaction systems, gaze dynamics cannot be specific to a single task. Therefore, we tested *H2* by examining whether the gaze patterns observed in the Tile Search task would generalize to a less controlled task, the Room Search dataset.

Similar to the Tile Search task, in the Room Search task, participants provided input through clicks on a hand-held controller and the input recognition error injected clicks experimentally. However, there were several differences between the two tasks that made the Room Search task more realistic. First, once the error was injected, the participant had to correct it immediately before moving on to

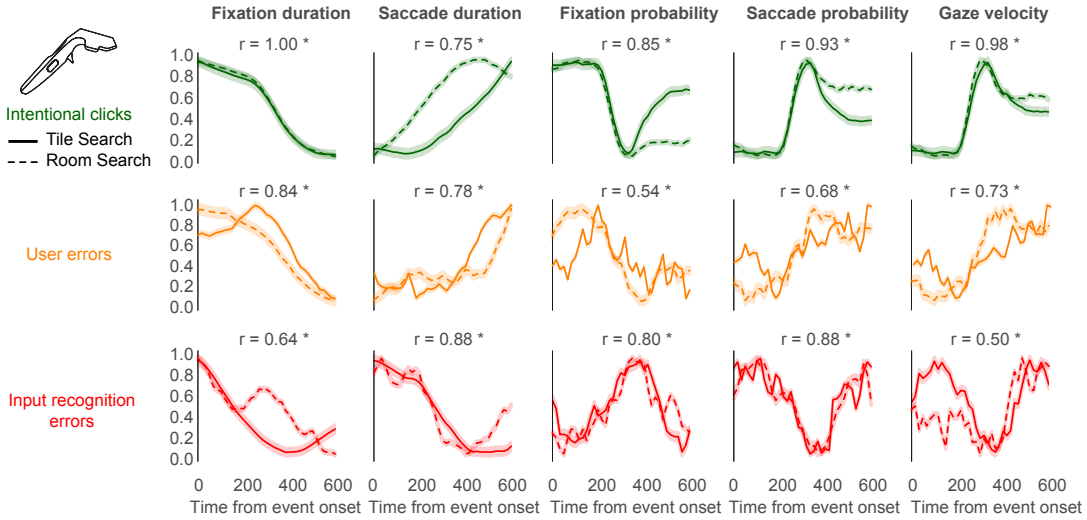

Figure 7: Visualizations of the time series of min-max normalized gaze features following intentional correct actions (green), input recognition errors (red), and user errors (orange) for the Tile Search task (solid line) and Room Search task (broken line). The Pearson Correlation r value between the pair of gaze features is shown on the top of each panel along with a an asterisk (*) if $p < 0.05$. Data is represented as Mean ± SEM. For simplicity, only the five most commonly used and evaluated eye tracking features are displayed.

any other tile in the Tile Search task, while in the Room Search task, the participant had the freedom to correct the input recognition error at any point during the session. This aspect of the Tile Search task required the participant to return their gaze to the tile that had been just viewed to correct it. Therefore, it could be argued that this action could invoke a stereotypical gaze pattern that would occur following an input recognition error that would have been absent for correct actions, resulting in differences in gaze patterns between input types. This needed not be the case in the Room Search task. Second, the injected errors were injected only for non-target objects in the Tile Search task while they were injected for both target and non-target objects in the Room Search task, making it more realistic and increasing the number of input event types (Fig. 3). Third, the participant was mostly static during the Tile Search task and used a limited range of head movements to perform the task, whereas in the Room Search task, the participant could freely navigate the environment and could select or deselect objects in a less constrained manner. Given the variations between these two task constraints, we investigated if the gaze patterns observed after the three types of input events were consistent across both tasks.

Following the previous approach (Fig. 5), the average of each feature's values (aligned to the input event onset) for each time point was computed for each participant. Then, a one-way ANOVA was performed on each data point to determine which time points were statistically different for each feature and corrected for multiple comparisons using the FDR method. Similar to the results of the Tile-Search task, significant differences ($p < 0.05$) were found among the three input events for all features after the onset of input events for Room-Search task as well (Fig. 6).

The gaze patterns between the two tasks were qualitatively compared by min-max normalizing the mean of each gaze feature per task and visualizing them (Fig. 7). To quantitatively compare them, the Pearson correlation was calculated between gaze features for the two tasks for each class and showed that the gaze features within the Tile Search and Room Search tasks had highly correlated gaze patterns (Fig. 7).

The temporal patterns for some gaze features such as the fixation probability and the saccade probability were initially highly correlated but then separated after about 300 milliseconds. This could due to the differences in task features between the two tasks. That is, in the Tile Search task, participants tended to perform a serial visual search, even though they were not instructed to do so, while in the Room Search task, participants tended to perform a random visual search. The differences in the motor preparation for consecutive actions between these two forms of visual searches could explain the differences in gaze temporal patterns. Together, these findings suggest that the pattern of some gaze features following different input events were generalizable across different tasks.

Given that the Room Search task was less constrained, there was a greater frequency of de-selections for intentional actions, input recognition errors, and user errors (Fig. 3). Therefore, we examined if, within each input event type, gaze features were consistent across different action types (i.e., selection and deselection). First, the average of each input triggered feature's value for each action type was computed for each time point and participant (i.e., selection and de-selection) . A paired t-test was then performed for each sample point to determine which time points were statistically different between the two actions for each feature. No statistically significant differences were found between the gaze features for the two action types for any of the gaze features (Fig. 8) suggesting that the patterns of gaze features were consistent across action types.

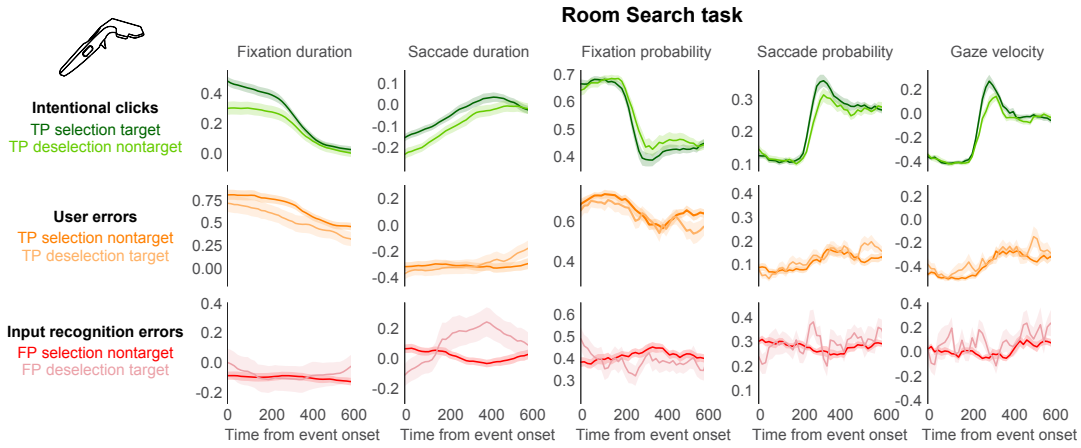

Figure 8: Selection and deselection actions were found to have similar patterns of z-scored gaze features for each input event type within the Room Search task dataset. None of the features had significant differences between the two action types. Data is represented as Mean ± SEM.

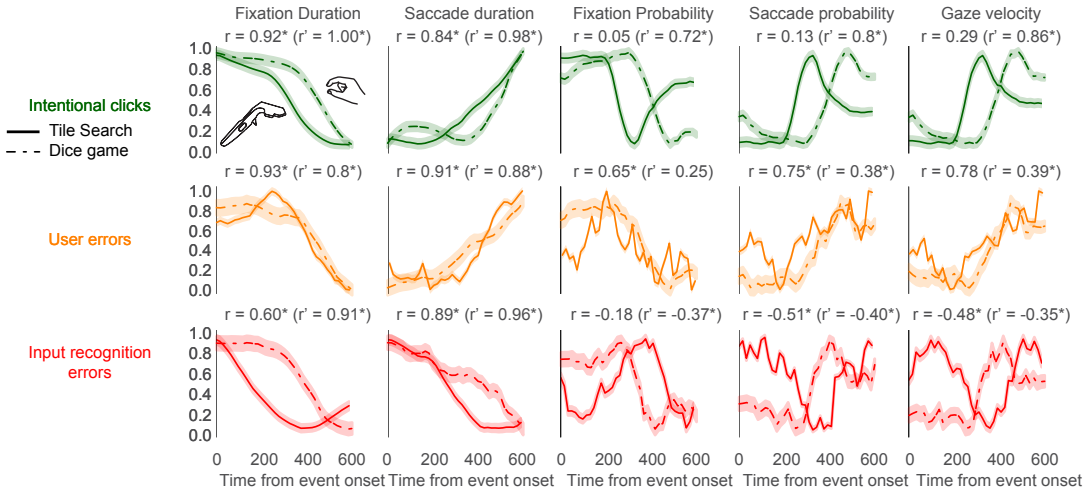

Figure 9: Min-max normalized gaze features for different input event types for the Dice Game task with the gestural input system (broken line) and Tile Search task with the controller key press (solid line). Data is represented as Mean ± SEM. The Pearson Correlation r value between the pair of gaze features is shown on the top of each panel along with a * if $p < 0.05$. The Pearson Correlation r value between the pair of gaze features with the features for the Dice Game task shifted to the left by 150 milliseconds is shown on the top of each panel is shown as r', along with a (*) if $p < 0.05$ in parenthesis. See Fig. A2 for lag-corrected gaze dynamics.

Together, these findings suggest that gaze features depended on the type of input event and not on the means (i.e., action type) by which it was achieved.

4.3 Temporal Dynamics of Gaze Features for Injected vs Real Input Recognition Errors

Across the Tile Search (Fig. 5) and Room Search tasks (Fig. 6), the temporal patterns of gaze features differed following intentional actions, input recognition errors, and user errors. However, in both cases, input recognition errors were simulated through semi-random injections. While these types of errors provide initial insights and advance our understanding of gaze behavior during human-computer interactions, they are not an accurate representation of the input recognition errors that would naturally occur in these interaction settings. It is possible that the gaze dynamics observed thus far are not representative of model-generated input recognition errors. As such, the Dice Game task was used to explore whether the patterns observed thus far would extend to a task controlled by an IMU-based pinch gesture recognizer.

Similar to the previous approach, the gaze patterns between the Tile Search and the Dice Game tasks were qualitatively compared by min-max normalizing the mean of each gaze feature per task

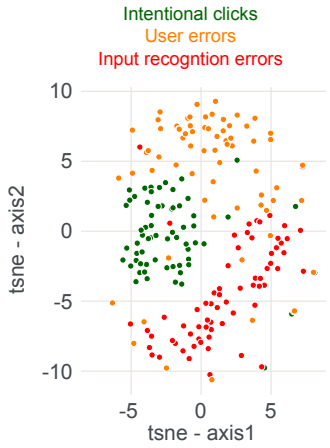

Figure 10: Gaze dynamics for the types of input events across the datasets clustered separately in a low dimensional embedding space.

and visualizing them (Fig. 9). As a result, there was a consistent average offset of 150 milliseconds for the gaze features for intentional actions in the Dice Game task, relative to the Tile Search task. This could be due a combination of several system specific parameters that were different between the controller vs gesture recognizer.

Therefore, to quantitatively compare the gaze features' temporal patterns between the two tasks, the Pearson correlation was computed between gaze features for the two tasks, as well as between gaze features for the two tasks after correcting for the systemic lag of 150 milliseconds (Fig. A2).

Although gaze features following intentional clicks had a consistent offset due to a systemic lag, there was no offset for the gaze features after user errors. The gaze features after input recognition errors did however, have inconsistent offsets. This is because the gesture recognizer system had a higher tendency to produce false positives following a sudden increase in hand velocity. However, in the Tile Search task and the Room Search task, false positive input recognition errors were injected at specific time windows. Given the different origins of these two types of input recognition errors and their occurrence during an interaction setting, differences in gaze behavior following these two types of events was expected.

A similar pattern of gaze behavior was also found between the Room Search task and Dice Game task for each of the three input event types (Fig. A3 with and without the offset correction). This offset issue is discussed in further detail in the Discussion section.

4.4 A Deep Learning Model Can Classify Input Events Using Only Gaze Dynamics

The results thus far have demonstrated that the gaze dynamics following intentional actions, input recognition errors, and user errors were different and that the differences were consistent across tasks, selection versus deselection actions, and gesture recognizer errors versus injected input recognition errors. Together, these findings demonstrate the feasibility of building a model that can detect the type of input event using only gaze dynamics.

As a first step to building the model, the gaze dynamics for the three input events across all three tasks were visualized in a low dimensional space. A t-stochastic neighbor embedding (t-SNE) approach [30] demonstrated that the gaze dynamics for the three input events clustered together in a low dimensional embedding space (Fig. 10). Specifically, both injected and real input recognition errors were clustered, suggesting that there was a shared intrinsic pattern between these two temporal dynamics which was not apparent in the above data analyses. This analysis thus suggests that a multi-class classifier could be trained to classify the three types of input events using gaze dynamics as input.

For each task, a separate TCN model was also trained using the first order differences between the x, y, z head-corrected eye position data. Epochs of data following each input event were isolated, with time 0 reflecting the input event onset and extending to ~650 milliseconds (i.e., 40 sample points) after the input event onset.

The AUC-ROC-OVR scores showed above chance performance for all three models (chance = 0.5 for AUC-ROC-OVR) (Fig. 11). The respective confusion matrices also showed well above chance precision for each class for each model (chance=0.33 for confusion matrix). Furthermore, the precision for each class and model was well distributed, suggesting that the models were not performing well on one class at the cost of another.

Motivated by the prior results that suggested that the gaze dynamics for all three classes of input events clustered together in a low dimensional embedding space across all three datasets, we hypothesized if we could train a single combined classifier to classify the three input classes across all three datasets. To investigate this, we trained a single TCN model with the 70% of the data from all three tasks combined, as input. We tested this model on the remaining 30% of data from all three tasks combined and found an AUC-ROC-OVR of 0.78, which was well above chance level (0.5). To ensure this model performed well on all the tasks individually, we also tested this model on the last 30% of the data from each task separately and found AUC-ROC-OVR scores of 0.75, 0.70 and 0.82 respectively for the Tile Search, Room Search and Dice Game tasks (Fig. 12A). These scores were only a 2-5% reduction with compared to individual models trained and tested on each task separately. Despite this slight reduction in performance, the combined model had a significant advantage over the individual models because it was trained on all the tasks together. Thus, these results suggest that it could potentially generalize to a new task. By computing four confusion matrices for each of the four testing conditions (i.e., all tasks together and each of the three tasks separately), we confirmed that the model also performed well on each class for all the tasks together (Fig. 12B) and each task separately (Fig. 12C-E).

Since the above models were trained and tested on data from the same participants, these models were not participant-independent. Therefore, we also trained a single, combined, participant-independent classifier that classified the three input classes across all three datasets by training on 70% of the participants and testing on the remaining 30% (Fig. A4). This type of model's performance (AUC-ROC-OVR = .78) was similar to the previous participant-dependent model's performance.

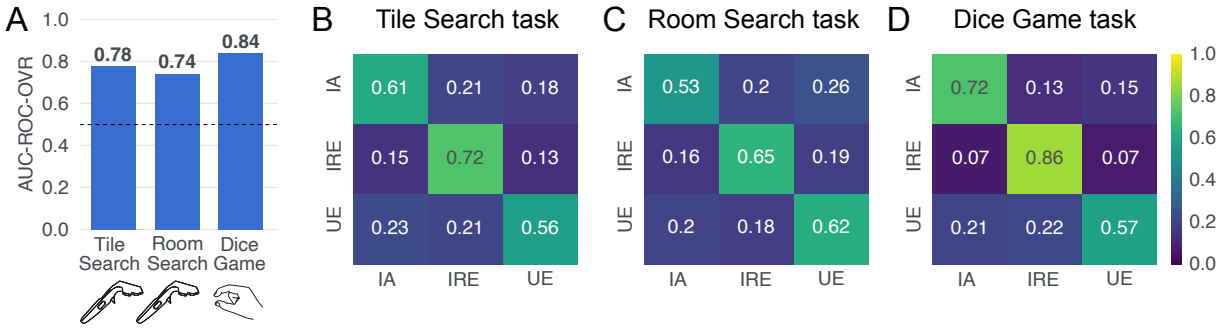

Figure 11: TCN deep learning models were found to be able to classify input events using only eye gaze data. (A) AUC-ROC-OVR scores from the 3 TCN models, each trained and tested on each of the three tasks independently. (B) A confusion matrix of the model for the Tile Search task from panel A. (C) A confusion matrix of the model for the Room Search task. (D) A confusion matrix of the model for the Dice Game task. (IA denotes intentional action, IRE denotes input recognition error, and UE denotes user error). Each cell in B-D denotes the precision for each class.

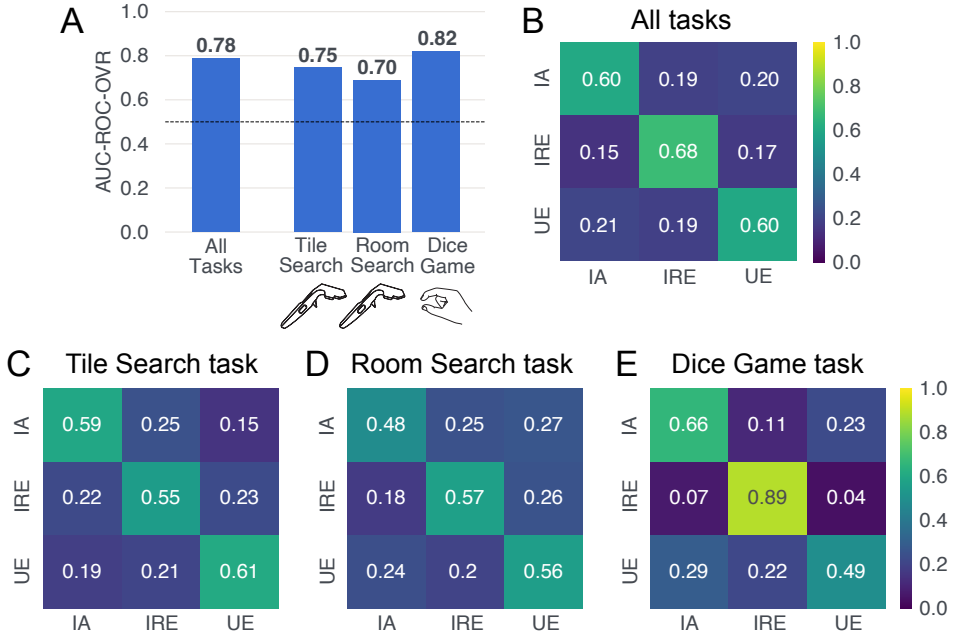

Figure 12: A single, combined TCN deep learning model was found to be able to classify input events based on eye gaze data, across 3 different tasks. (A) AUC-ROC-OVR scores from the combined TCN model trained on all three tasks together and tested on all three tasks together (from left to right: the Tile Search task, Room Search task and Dice Search task). (B) A confusion matrix of the combined model trained on all three tasks and tested on all three tasks. (C) A confusion matrix of the combined model trained on all three tasks and tested on the Tile Search task. (D) A confusion matrix of the combined model trained on all three tasks and tested on the Room Search task. (E) A confusion matrix of the combined model trained on all three tasks and tested on the Dice Game task. (IA denotes intentional action, IRE denotes input recognition error, and UE denotes user error). Each cell in B-E is the precision for each class.

Together, these results demonstrate that using only a user's natural gaze behavior, deep learning models can distinguish between user errors, input recognition errors, and intentional selections.

5 DISCUSSION

In this study we tested two hypotheses for the gaze dynamics after event events. First, we demonstrated that it is possible to classify three different types of input events using only a user's gaze behavior after an input event (*H1*) and then we showed that the gaze dynamics following these events were consistent across

distinct tasks, action types (selection vs. deselection), and injected versus recognizer-driven input errors (*H2*).

5.1 The Generalizability of Gaze Dynamics

Gesture-based recognition systems are susceptible to input recognition errors, which negatively affect user experiences. This research proposed the use of user gaze patterns following an input event to detect these errors, which then could be used to facilitate adaptive error mediation. However, to build a reliable detection model for input recognition errors, gaze behaviour following these errors must manifest consistently across tasks and input types.

The data for both the Tile Search and Room Search tasks used in this research was collected using an HTC VIVE Pro controller, whereas the data for Dice Game was collected using a pinch gesture recognition system. This difference in input medium led to a lag of 150 milliseconds in the temporal dynamics between these two systems (Fig. 9, Fig. A3). Notably, this lag was consistent for all the gaze features for intentional selections, was not present for user errors, and was variable for input recognition errors. This could be due to several reasons.

First, the pattern of gaze dynamics following intentional action was shifted consistency across the features (Fig. 9 by 150 milliseconds. This could be due to a combination of several factors, two of those being: (1) the computer vision based hand tracking for the target and IMU-based pinch recognition for selection were both less reliable and had some lag compared to the controller, which might have led participants to dwell after a selection to make sure the selections registered, (2) the cognitively more complex decision making that was required during selection in the Dice Game task might have led to slower movements between selections overall. More systematic future research is required to study the contributions of these confounding factors.

Next, the gaze patterns after an injected input recognition error were different from those after a real recognition error. This is probably due to differences in *when* input recognition errors occurred with the natural gesture recognizer in Dice Game task relative to *when* the errors were experimentally injected in the Tile Search and Room Search tasks. For example, the IMU-based recognizer used in the Dice Game task was more likely to produce input recognition errors during sudden changes in hand movement velocities or rapid/jerky hand movements (i.e., during pointing). However the window during which the simulated input recognition errors were pseudo-randomly injected in the other two tasks was agnostic of the concurrent hand movement velocity. Although the data analysis showed differences in gaze dynamics between these two systems for input recognition errors, in a low dimensional embedding space, the gaze dynamics for both these errors were found to cluster together (Fig. 10). Furthermore, a TCN model was able to classify the gaze dynamics for both of these types of errors as the same class and distinguish this class from all other classes with well above chance performance (Fig. 12). These results suggest that the gaze features for these two types of errors might share some latent patterns that a deep learning model could leverage during classification.

The results of the present study significantly set it apart from those of previous studies that have used gaze towards error detection such as Peacock et al [52]. For example Peacock et al used a simple two class logistic regression model to classify intentional actions and input recognition errors using a set of 10 gaze features. However, this approach might not work towards building a generalizable model because different gaze features might be important in different tasks and contexts. For example, saccade amplitude and duration might be important features for a task with sparse targets requiring longer saccades while saccade velocity might be an important feature for a task involving rapid decision making and urgency. In this study we developed a generalizable model by using the first order differences of gaze positions rather than using gaze features circumventing the problem of task dependent feature selection. Second, we used a robust deep learning model (TCN architecture) and performed a three-class classification. Although our 3-class deep learning model has almost similar performance (AUC-ROC-OVR of 0.78) as Peacock et al.'s 2-class logisitc regression model (AUC-ROC of 0.80), it can detect three types of input events and performs well on three different tasks with significantly higher than chance level precision for all the classes (Fig. 11, Fig. 12, Fig. A4).

5.2 Mediation Techniques and Applications

Input recognition errors are particularly costly for user experiences so designing systems that are capable of detecting them in the moments after they occur affords the possibility of also designing error-type dependent error mediation techniques that could assist users with error recovery.

5.2.1 Model Refinement and Personalization. One promising application for the present work is to label data to improve or personalize input recognition models. For example, if an input recognition system were able to detect an input recognition error, it could use this information to retrain its recognition model for an individual user to minimize such errors in the future. In some cases, systems might be able to detect errors without the model presented in this paper, particularly when an undo option is available, however, many of the actions taken in most systems cannot be undone (e.g., sending a message). Even if actions can be undone, users might not always to undo an action, particularly if the action is costly. As such, the present model provides a new option for detecting probable errors and demonstrates the feasibility of this approach. Specifically, this research demonstrated the reliable discrimination of intentional actions versus input recognition errors across several tasks.

5.2.2 Adaptive Error Mediation. This research also showed how gaze dynamics could be used to detect both input recognition errors and user errors. Future systems could use this information to provide adaptive error recovery techniques that are specific to certain error types. For example, to assist users with the correction of an input error, a system might launch a confirmation dialogue, whereas to assist users with a potential user error, a system might instead highlight an item. In other words, users might prefer different types of adaptive mediation techniques depending on if an error is recognizer-generated versus user-generated.

In future work, adaptive mediation techniques could be explored along three axes: what, when, and how. First, *what* are the different means of notifying the user of a recent potential error? For example, systems could use a heads-up display (HUD) that would be view invariant and noticeable regardless of where the user is looking, or, alternatively, they could provide a notification just above an incorrect object that might not always be in the user's current field of view. These different placements could have important implications in the management of visual clutter in the XR systems. Second, *when* should these mediation techniques be implemented? Previous work by Peacock et al. suggested that models perform best around 600 milliseconds after an input event [52], however, depending on the probability of error detection in natural VR interactions, this duration could be flexible. For example, perhaps the system should take an evidence accumulation approach, launching adaptive mediation as soon as it achieves an appropriate level of confidence about a probable error. Finally, *how* should these mediation techniques be implemented? Depending on the confidence level of the error detection, a system could adaptively propose different mediation techniques for smooth recovery from input recognition errors. For example, if detection confidence was high, a system could automatically correct an error by undoing the action without the user requiring user input (i.e., active recovery). This could increase a user's trust in the recognition system and create richer, more fluid interaction. If the detection confidence level was low, then a system could notify the user of a potential error and provide them with options to undo or cancel the recent action (i.e., passive recovery). Such system-driven, interactive error mediation techniques could save the user time during error recovery and enable low friction user experiences.

5.2.3 Developing New Adaptive Designs. As a final potential application, the findings that (1) the gaze dynamics for injected and real input recognition errors cluster together in a low dimensional embedding space (Fig. 10) and (2) a deep learning model can be used to potentially classify both of these types of errors under a single class and distinguish this class from the user error and intentional actions (Fig. 12) has implications for how mediation techniques could be studied in the future. In particular, it suggests that training data for gaze-based error detection models, or experiments to test these models, need not use a gesture recognizer. Instead, such training data could integrate injected errors. This could enable for more efficient testing of error recognition models and the mediation techniques that use these models.

5.3 Limitations and Future Work

Although we show that a deep learning model can be used to classify the input events based on gaze dynamics, the performance of the present model (AUC-ROC-OVR = 0.78) might not be sufficient for use in a closed-loop interaction system. That is, it is still an open question if this level of task accuracy will suffice for the user to see a significant difference in their experience in real world VR interaction. This will depend on the application of the model (e.g., input recognizer personalization, adaptive error mediation) and the specific design of the application. The TCN model demonstrated substantial performance improvements and could be improved in future work by exploring different architectures or adding additional or larger datasets. Another way to improve the model could be by using a multi-modal approach. That is, although gaze was used to detect input events, it is not the only indicator of error. Other modalities, such as hand movement and/or the recent history of selections, could provide rich information about ongoing interaction. Although such modalities could be cheaper to use and potentially increase accuracy along with gaze, they could also be slower. For example, eye movements have been found to precede hand movements [4, 53], suggesting that eye movements might be the earliest indicator of the user's next interaction goal. Future work should examine if model performance can be improved by integrating other modalities as input.

Furthermore, the present study focused on three tasks that were all point-and-click based and only situations where one gesture (i.e., pinch gesture) or one click could be performed. Users' gaze behavior, however, might differ for other types of gestures and task interactions such as scrolling or swipe gestures or in situations where there are multiple gestures available, each for a different action. Thus, further investigation is needed into the utility of gaze behavior in such settings and the generalizability of gaze dynamics for different input events.

Finally, this research only focused on *reactive* users gaze behavior *after* an event and demonstrated the potential of using deep learning models to *detect* input events based on gaze features. However, future work needs to explore models that could use the gaze behavior *before* an event to *predict* the type of input events that would lead to an action [16]. These types of models could have levels of pro-active error control and mediation and assist users with low friction interactions.

6 CONCLUSION

The present research demonstrates that gaze dynamics alone can be used to detect input recognition errors and user errors soon after they occur. Systems employing such dynamics can thus use this information to solve the challenge of imperfect input recognition. That is, rather than improving a recognition model, systems could leverage natural user behaviors immediately following an input event to detect errors. The rapid detection of recent input recognition errors has critical implications for the development of intelligent systems that can assist with error recovery and can help increase the usability of input systems by supporting low friction interactions.

REFERENCES

[1] Aliza Abeles, Ann Blandford, Paul Cairns, Anna L Cox, Simon YW Li, and Richard M Young. 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*, Vol. 28.

[2] Gregory D Abowd and Alan J Dix. 1992. Giving undo attention. *Interacting with computers* 4, 3 (1992), 317–342.

[3] Richard A Abrams, David E Meyer, and Sylvan Kornblum. 1990. Eye-hand coordination: oculomotor control in rapid aimed limb movements. *Journal of experimental psychology: human perception and performance* 16, 2 (1990), 248.

[4] RW Angel, W Alston, and H Garland. 1970. Functional relations between the manual and oculomotor control systems. *Experimental Neurology* 27, 2 (1970), 248–257.

[5] Caroline Appert, Olivier Chapuis, and Emmanuel Pietriga. 2012. Dwell-and-spring: undo for direct manipulation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1957–1966.

[6] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. 2018. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271* (2018).

[7] 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 (2006), 685–708.

[8] Nikola Banovic, Tovi Grossman, and George Fitzmaurice. 2013. The effect of time-based cost of error in target-directed pointing tasks. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1373–1382.

[9] Nikola Banovic, Varun Rao, Abinaya Saravanan, Anind K Dey, and Jennifer Mankoff. 2017. Quantifying aversion to costly typing errors in expert mobile text entry. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 4229–4241.

[10] 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*. 83–90.

[11] Ty W Boyer and Matthew Wang. 2018. Direct gaze, eye movements, and covert and overt social attention processes. *Attention, Perception, & Psychophysics* 80, 7 (2018), 1654–1659.

[12] Katherine Breeden and Pat Hanrahan. 2017. Gaze data for the analysis of attention in feature films. *ACM Transactions on Applied Perception (TAP)* 14, 4 (2017), 1–14.

[13] Michael D Byrne and Susan Bovair. 1997. A working memory model of a common procedural error. *Cognitive science* 21, 1 (1997), 31–61.

[14] Michael D Byrne and Elizabeth M Davis. 2006. Task structure and postcompletion error in the execution of a routine procedure. *Human Factors* 48, 4 (2006), 627–638.

[15] 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, 777–782.

[16] Brendan David-John, Candace 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*. 1–7.

[17] 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 (2013), 5–5.

[18] Stefan Dowiasch, Svenja Marx, Wolfgang Einhäuser, and Frank Bremmer. 2015. Effects of aging on eye movements in the real world. *Frontiers in human neuroscience* 9 (2015), 46.

[19] Sarah L Elliott, Mark A Georgeson, and Michael A Webster. 2011. Response normalization and blur adaptation: Data and multi-scale model. *Journal of Vision* 11, 2 (2011), 7–7.

[20] B Fischer and L Rogal. 1987. EYE-HAND-COORDINATION: A REACTION TIME STUDY IN MAN AND MONKEY. In *Eye Movements from Physiology to Cognition*. Elsevier, 162–163.

[21] Marco Furtner and Pierre Sachse. 2008. The psychology of eye-hand coordination in human computer interaction. In *Proc. HCI*, Vol. 8. 144–149.

[22] Christoph Gebhardt, Brian Hecox, Bas van Opheusden, Daniel Wigdor, James Hillis, Otmar Hilliges, and Hrvoje Benko. 2019. Learning cooperative personalized policies from gaze data. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 197–208.

[23] 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)*. IEEE, 1–5.

[24] Sylvie Gibet, Nicolas Courty, and Jean-François Kamp. 2005. Gesture in Human-Computer Interaction and Simulation 6th International Gesture Workshop, GW 2005, Berder Island, France, May 18-20, 2005, Revised Selected Papers. In *Conference proceedings GW*. Springer, 239.

[25] Denis Glencross and Nicholas Barrett. 1983. Programming precision in repetitive tapping. *Journal of Motor Behavior* 15, 2 (1983), 191–200.

[26] Peter A Hancock and Karl M Newell. 1985. The movement speed-accuracy relationship in space-time. In *Motor behavior*. Springer, 153–188.

[27] 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, 1 (2003), 6–6.

[28] 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*. 185–194.

[29] John M Henderson, Phillip A Weeks Jr, 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.

[30] Geoffrey E Hinton and Sam Roweis. 2002. Stochastic neighbor embedding. *Advances in neural information processing systems* 15 (2002).

[31] Alexander Hoffmann, Daniel Spelmezan, and Jan Borchers. 2009. TypeRight: a keyboard with tactile error prevention. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 2265–2268.

[32] HTC. 2021. Vive Trcker. *https://www.vive.com/us/accessory/tracker3/* (2021).

[33] Tomoki Ishikawa and Takahiro Yakoh. 2021. Saliency prediction based on object recognition and gaze analysis. *Electronics and Communications in Japan* 104, 2 (2021), e12303.

[34] Hassan Ismail Fawaz, Germain Forestier, Jonathan Weber, Lhassane Idoumghar, and Pierre-Alain Muller. 2019. Deep learning for time series classification: a review. *Data mining and knowledge discovery* 33, 4 (2019), 917–963.

[35] Howell Istance, Richard Bates, Aulikki Hyrskykari, and Stephen Vickers. 2008. Snap clutch, a moded approach to solving the Midas touch problem. In *Proceedings of the 2008 symposium on Eye tracking research & applications*. 221–228.

[36] 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 Transactions on Interactive Intelligent Systems* 9, 2-3 (25 4 2019), 1–30. https://doi.org/10.1145/3181672

[37] Wolf Kienzle, Eric Whitmire, Chris Rittaler, and Hrvoje Benko. 2021. ElectroRing: Subtle Pinch and Touch Detection with a Ring. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems* (Yokohama, Japan) *(CHI '21)*. Association for Computing Machinery, New York, NY, USA, Article 3, 12 pages. https://doi.org/10.1145/3411764.3445094

[38] Kathryn Koehler and Miguel P Eckstein. 2015. Scene Inversion Slows the Rejection of False Positives through Saccade Exploration During Search.. In *CogSci*.

[39] Scott A Kuhl, William B Thompson, and Sarah H Creem-Regehr. 2009. HMD calibration and its effects on distance judgments. *ACM Transactions on Applied Perception (TAP)* 6, 3 (2009), 1–20.

[40] Manu Kumar, Terry Winograd, and Andreas Paepcke. 2007. Gaze-enhanced scrolling techniques. In *CHI'07 Extended Abstracts on Human Factors in Computing Systems*. 2531–2536.

[41] 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 *The 34th Annual ACM Symposium on User Interface Software and Technology*. 54–68.

[42] Page Laubheimer. 2015. Preventing user errors: avoiding unconscious slips. *Nielsen Norman Group* (2015).

[43] Geoffrey R Loftus and Norman H Mackworth. 1978. Cognitive determinants of fixation location during picture viewing. *Journal of Experimental Psychology: Human perception and performance* 4, 4 (1978), 565.

[44] Marco Loregian. 2008. Undo for mobile phones: does your mobile phone need an undo key? do you?. In *Proceedings of the 5th Nordic conference on Human-computer interaction: building bridges*. 274–282.

[45] Katsutoshi Masai and Kai Kunze. [n.d.]. Maki sugimoto, and Mark Billinghurst. 2016. Empathy Glasses. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA'16). ACM, New York, NY, USA*. 1257–1263.

[46] Ann McNamara, Reynold Bailey, and Cindy Grimm. 2008. Improving search task performance using subtle gaze direction. In *Proceedings of the 5th Symposium on Applied Perception in Graphics and Visualization*. 51–56.

[47] David E Meyer, Richard A Abrams, Sylvan Kornblum, Charles E Wright, and JE Keith Smith. 1988. Optimality in human motor performance: ideal control of.

[48] George Nagy, Thomas A Nartker, and Stephen V Rice. 1999. Optical character recognition: An illustrated guide to the frontier. In *Document recognition and retrieval VII*, Vol. 3967. SPIE, 58–69.

[49] 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*. 147–150.

[50] Abigail Noyce, Jessica Maryott, and Robert Sekuler. 2010. Unexpected events, predictive eye movements, and imitation learning. *Journal of Vision* 10, 7 (2010), 755–755.

[51] Candace E Peacock, Brendan David-John, Ting Zhang, T Scott Murdison, Matthew J Boring, Hrvoje Benko, and Tanya R Jonker. 2021. Gaze signatures decode the onset of working memory encoding. In *CHI2021 Eye Movements as an Interface to Cognitive State (EMICS) Workshop Proceedings. ACM*.

[52] Candace E. Peacock, Ben Lafreniere, Ting Zhang, Stephanie Santosa, Hrvoje Benko, and Tanya R. Jonker. 2022. Gaze as an Indicator of Input Recognition Errors. In *ACM Symposium on Eye Tracking Research and Applications*. 14 pages. (in press).

[53] C Prablanc, JF Echallier, E Komilis, and M Jeannerod. 1979. Optimal response of eye and hand motor systems in pointing at a visual target. *Biological cybernetics* 35, 2 (1979), 113–124.

[54] Raj M Ratwani, J Malcolm McCurry, and J Gregory Trafton. 2008. Predicting post-completion errors using eye movements. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 539–542.

[55] Raj M Ratwani, J Gregory Trafton, and Christopher Myers. 2006. Helpful or harmful? Examining the effects of interruptions on task performance. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 50. SAGE Publications Sage CA: Los Angeles, CA, 372–375.

[56] Jun Rekimoto. 1999. Time-machine computing: a time-centric approach for the information environment. In *Proceedings of the 12th annual ACM symposium on*

User interface software and technology. 45–54.

[57] Thomas R Reppert, Karolina M Lempert, Paul W Glimcher, and Reza Shadmehr. 2015. Modulation of saccade vigor during value-based decision making. *Journal of Neuroscience* 35, 46 (2015), 15369–15378.

[58] Gerhard Rigoll, Andreas Kosmala, and Stefan Eickeler. 1997. High performance real-time gesture recognition using hidden markov models. In *International Gesture Workshop*. Springer, 69–80.

[59] Jaime Ruiz and Yang Li. 2011. DoubleFlip: a motion gesture delimiter for mobile interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 2717–2720.

[60] Kareisha Sackaloo, Emily Strouse, and Martin S Rice. 2015. Degree of preference and its influence on motor control when reaching for most preferred, neutrally preferred, and least preferred candy. *OTJR: Occupation, Participation and Health* 35, 2 (2015), 81–88.

[61] 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 & applications*. 71–78.

[62] Philippe Schmid, Sylvain Malacria, Andy Cockburn, and Mathieu Nancel. 2020. Interaction Interferences: Implications of Last-Instant System State Changes. In *Proceedings of the 33rd Annual ACM Symposium on User Interface Software and Technology*. 516–528.

[63] Naveen Sendhilnathan, Debaleena Basu, Michael E Goldberg, Jeffrey D Schall, and Aditya Murthy. 2021. Neural correlates of goal-directed and non–goal-directed movements. *Proceedings of the National Academy of Sciences* 118, 6 (2021).

[64] Hemant Bhaskar Surale, Fabrice Matulic, and Daniel Vogel. 2017. Experimental analysis of mode switching techniques in touch-based user interfaces. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 3267–3280.

[65] Markus Tatzgern, Valeria Orso, Denis Kalkofen, Giulio Jacucci, Luciano Gamberini, and Dieter Schmalstieg. 2016. Adaptive information density for augmented reality displays. In *2016 IEEE Virtual Reality (VR)*. 83–92. https://doi.org/10.1109/VR.2016.7504691

[66] Diman Zad Tootaghaj, Adrian Sampson, Todd Mytkowicz, and Kathryn S McKinley. 2017. High five: improving gesture recognition by embracing uncertainty. *arXiv preprint arXiv:1710.09441* (2017).

[67] Neff Walker, David E Meyer, and John B Smelcer. 1993. Spatial and temporal characteristics of rapid cursor-positioning movements with electromechanical mice in human-computer interaction. *Human Factors* 35, 3 (1993), 431–458.

[68] Colin Ware and Harutune H Mikaelian. 1986. An evaluation of an eye tracker as a device for computer input2. In *Proceedings of the SIGCHI/GI conference on Human factors in computing systems and graphics interface*. 183–188.

[69] Hongyi Wen, Julian Ramos Rojas, and Anind K. Dey. 2016. Serendipity: Finger Gesture Recognition Using an Off-the-Shelf Smartwatch. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '16)*. Association for Computing Machinery, New York, NY, USA, 3847–3851. https://doi.org/10.1145/2858036.2858466

[70] Wikipedia. 2022. Yahtzee. *https://en.wikipedia.org/wiki/Yahtzee* (2022).

[71] Huiyue Wu and Jianmin Wang. 2016. A visual attention-based method to address the midas touch problem existing in gesture-based interaction. *The Visual Computer* 32, 1 (2016), 123–136.

[72] Jiawei Yang, Guangtao Zhai, and Huiyu Duan. 2019. Predicting the visual saliency of the people with VIMS. In *2019 IEEE Visual Communications and Image Processing (VCIP)*. IEEE, 1–4.

[73] 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 (2021), 1155.

[74] Weilie Yi and Dana Ballard. 2009. Recognizing behavior in hand-eye coordination patterns. *International Journal of Humanoid Robotics* 6, 03 (2009), 337–359.

[75] 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*. 1–13.

[76] Ting Zhang, Zhenhong Hu, Aakar Gupta, Chi-Hao Wu, Hrvoje Benko, and Tanya R. Jonker. 2022. RIDS: Implicit Detection of a Selection Gesture Using Hand Motion Dynamics During Freehand Pointing in Virtual Reality. In *Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology*. https://doi.org/10.1145/3526113.3545701

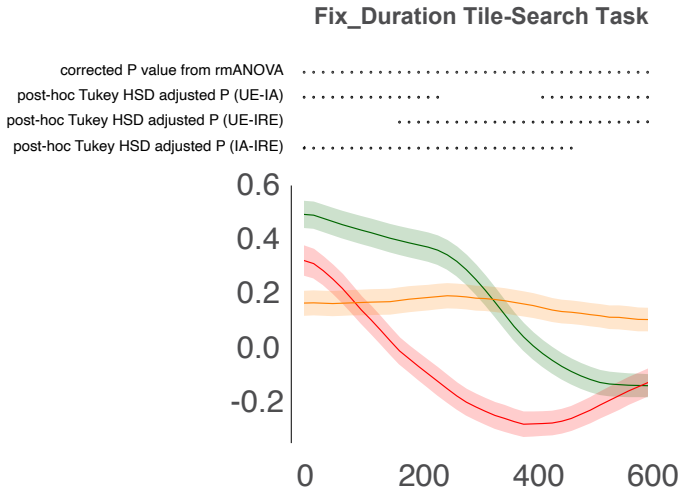

Figure A1: Visualizations of the time series of z-scored fixation duration feature, following intentional correct actions (green), input recognition errors (red), and user errors (orange) for the Tile Search task. Circular markers on the top correspond when the data significantly differed (corrected $p < 0.05$) between different conditions for different statistical tests as noted. This figure uses the same format as Fig. 5.

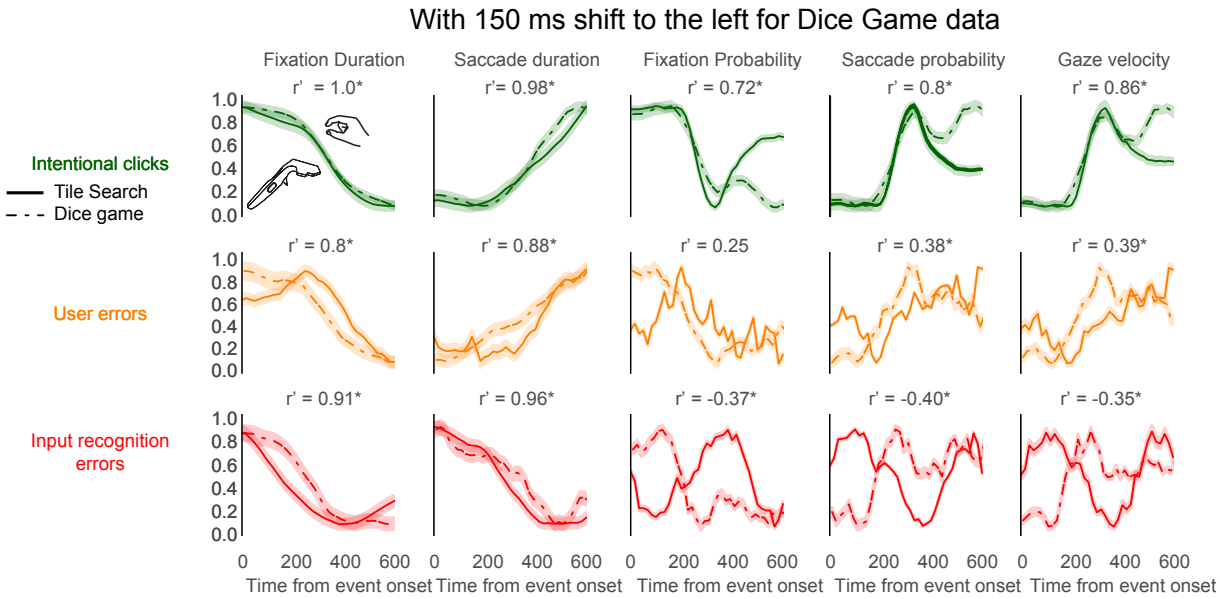

Figure A2: Gaze features for different input event types for the Tile Search task with the controller key press (solid line) from Fig. 9 and the Dice Game task with the gestural input system (broken line) temporally shifted to the left by 150 milliseconds. This figure uses the same format as Fig. 9

Figure A3: Gaze features for different input event types for the Room Search task with the controller key press (solid line) and the Dice Game task with the gestural input system (broken line) without any temporal shift (A) and with 150 milliseconds temporal shift to the left (B). This figure uses the same format as Fig. 9.

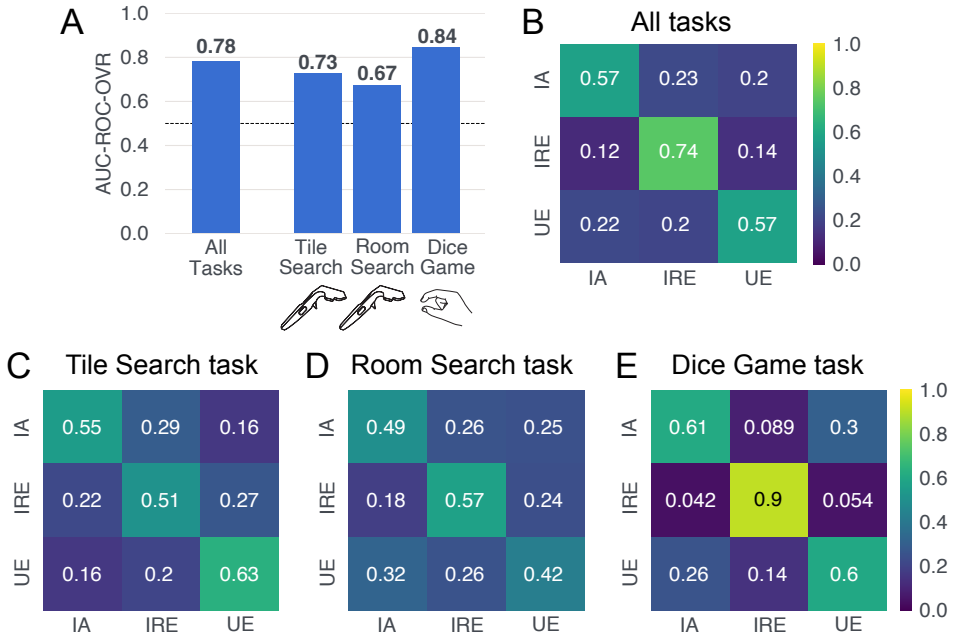

Figure A4: A single, combined, participant-independent TCN deep learning model was found to classify input events based on eye gaze data alone after their occurrence, across 3 different tasks. (A) AUC-ROC-OVR scores from the combined, participant-independent TCN model trained on all three tasks together and tested on all three tasks together (from left to right: the Tile Search task, Room Search task and Dice Search task). (B) A confusion matrix of the combined, participant-independent model trained on all three tasks and tested on all three tasks. (C) A confusion matrix of the combined, participant-independent model trained on all three tasks and tested on the Tile Search task. (D) A confusion matrix of the combined, participant-independent model trained on all three tasks and tested on the Room Search task (E) A confusion matrix of the combined, participant-independent model trained on all three tasks and tested on the Dice Game task. (IA denotes intentional action, IRE denotes input recognition error, and UE denotes user error). Each cell in B-E is the precision for each class. This figure uses the same format as Fig. 12.