Artificial Intelligence for Engineering Design, Analysis and Manufacturing

www.cambridge.org/aie

A user model to directly compare two unmodified interfaces: a study of including errors and error corrections in a cognitive user model

Farnaz Tehranchi[1], Amirreza Bagherzadeh[2] and Frank E. Ritter[3]

[1]School of Engineering Design and Innovation, The Pennsylvania State University, University Park, PA, USA; [2]Department of Industrial and Manufacturing Engineering, The Pennsylvania State University, University Park, PA, USA and [3]College of IST, The Pennsylvania State University, University Park, PA, USA

Research Article

Cite this article: Tehranchi F, Bagherzadeh A and Ritter FE (2023). A user model to directly compare two unmodified interfaces: a study of including errors and error corrections in a cognitive user model. *Artificial Intelligence for Engineering Design, Analysis and Manufacturing*, **37**, e27, 1–11
https://doi.org/10.1017/S089006042300015X

Received: 13 June 2022
Revised: 12 May 2023
Accepted: 26 June 2023

Keywords:
cognitive models; user experience and usability; design; vision and motor knowledge; errors

Corresponding author:
Farnaz Tehranchi;
Email: farnaz.tehranchi@psu.edu

Abstract

User models that can directly use and learn how to do tasks with unmodified interfaces would be helpful in system design to compare task knowledge and times between interfaces. Including user errors can be helpful because users will always make mistakes and generate errors. We compare three user models: an existing validated model that simulates users' behavior in the Dismal spreadsheet in Emacs, a newly developed model that interacts with an Excel spreadsheet, and a new model that generates and fixes user errors. These models are implemented using a set of simulated eyes and hands extensions. All the models completed a 14-step task without modifying the system that participants used. These models predict that the task in Excel is approximately 20% faster than in Dismal, including suggesting why, where, and how much Excel is a better design. The Excel model predictions were compared to newly collected human data (N = 23). The model's predictions of subtask times correlate well with the human data (r2 = .71). We also present a preliminary model of human error and correction based on user keypress errors, including 25 slips. The predictions to data comparison suggest that this interactive model that includes errors moves us closer to having a complete user model that can directly test interface design by predicting human behavior and performing the task on the same interface as users. The errors from the model's hands also allow further exploration of error detection, error correction, and different knowledge types in user models.

Introduction

Computational models that simulate human intelligence and behavior are not yet always complete and not always capable of interacting with the environment. Newell (1990), in his *Unified Theories of Cognition*, explains his dream of developing reusable mechanisms for cognitive and computational architectures to complete a task. Cognitive architectures are designed explicitly for modeling reusable mechanisms, such as Soar (Newell, 1990; Laird, 2012), ACT-R (Anderson, 2007; Ritter et al., 2018), EPIC (Kieras and Meyer, 1997), and PyIBL (Gonzalez et al., 2003). A user model combines task knowledge and a cognitive architecture with its fixed mechanisms to apply the knowledge and generate behavior.

Developing a user model at each design step supports the risk-driven spiral system development approach that includes human factors concerns (Pew and Mavor, 2007). Also, user models can be used to discuss types of users, types of tasks, and the number or complexity of the tasks in the system while keeping track of the design requirements (Ritter, 2019), and has been long desired (e.g., Card et al., 1983; Byrne et al., 1994; Elkind et al., 1989).

Different approaches investigate adding interaction to a cognitive architecture to develop cognitive models. Developing human-like models that perform users' behavior has been envisioned before (e.g., Byrne et al., 1994; Lohse, 1997; Pew and Mavor, 2007) but has not been widely applied. Some practical approaches that modelers use to have models interact with a simulated world are ESegMan (Tehranchi and Ritter, 2017), ACT-R/PM (Byrne and Anderson, 1998), Cognitive code (Salvucci, 2009, 2013), JSON ACT-R (Hope et al., 2014), SegMan (St. Amant et al., 2005), and ACT-CV (Halbrügge, 2013). In these approaches, it will be difficult to reuse these models (their task knowledge) because of the domain-specific nature of the models – these models can only interact with the interfaces that are modified or created for these tools – the interfaces have been modified to provide the models' access to information on the display in that application using mark-up languages or other APIs. Also, the interfaces have been adjusted to accept input commands directly from models.

In contrast, we use the Java Segmentation and Manipulation (JSegMan) tool (Tehranchi and Ritter, 2018a,b), which works independently of interface design. This tool allows cognitive models to see objects on the screen using pattern recognition and execute actions such as clicking and typing using the operating system's event queue. With JSegMan, cognitive user models include a simulated eye that can see what is on an interface and a simulated hand that allows models to move the mouse, click, and type. Models developed with JSegMan and ACT-R are called Eyes and Hands models. This approach allows models to interact with any unmodified interfaces.

The Kim spreadsheet task (KST; Kim, 2008), unlike previous research on text-editing tasks, can give a balanced set of knowledge and skills, including procedural, declarative, and perceptual-motor knowledge. It also duplicates some common work tasks, such as manipulating spreadsheets. It is explained further below.

Using interactive cognitive models with reliable predictions can be cost-effective in designing decision-making processes involving human–computer interaction (Ritter et al., 2000) and developing cognitive assistance models (Klaproth et al., 2019). Interactive cognitive models, such as Eyes and Hands models (Tehranchi and Ritter, 2018a), will reduce the discrepancy between model and user capabilities, better predict human performance in interactive designs, provide more realistic input, simulate error correction, and be useful for AI interface agents. Recent models that interact with the Space Fortress video game (Anderson et al., 2019; Fincham et al., 2022) both interact with a representation of the game state placed directly into working memory by the task simulation. A recent model of driving (Held et al., 2022) interacted directly with a simulation of the car, receiving the ground truth as symbols into visual memory, although it did move its visual attention around a simulation of a visual field.

Interface design can benefit from user models that can predict and compare task times on interfaces without modifying the interfaces. It would also be useful to include errors and error corrections because users will always make mistakes and generate errors. We present several user models that interact directly with interfaces and are able to make errors and correct them.

The growth of technologies has led to more complex and reliable systems and user interfaces (Philippart, 2018; Zhan et al., 2019). This complexity has resulted in human factors errors, such as erroneous behavior, poor judgment, failure to properly follow the standard procedures, and has become a primary cause of accidents (Chen et al., 2019, 2020).

Different studies have explored the role of human error in reliability analysis and focused on various aspects of human performance and cognition but have not been used to extend cognitive architectures. For example, Di Pasquale et al. (2020) conducted a scoping review on aging and human–system errors in manufacturing, and Nor et al. (2021) provided a 20-year review of reliability engineering applications in various industries. In addition, recent research has started to acknowledge the importance of understanding human factors and their influence on system reliability (Pan et al., 2017; Alvarenga and e Melo, 2019; Tao et al., 2020). Yet there is a notable gap regarding using cognitive architecture to model the cognitive process of human error in reliability analysis literature.

A few studies have begun to explore cognitive modeling in reliability analysis, such as Shahab et al.'s (2022a,b) two-part series on Hidden Markov Model (HMM) based models of control room operator's cognition during process abnormalities, which focused on formalism and model identification (Part 1), and application to operator training (Part 2). Furthermore, Zhao and Smidts (2021a,b) developed a cognitive modeling and simulation environment for human performance assessment (CMS-BN). These studies demonstrate the potential of incorporating cognitive architectures in reliability analysis; however, their explicit integration remains limited.

We present three Eyes and Hands models: a Dismal model, an Excel model, and an Error model. These models perform a non-iterative spreadsheet task containing 14 subtasks in two spreadsheet tools. The Excel model's predictions are compared to human data (N = 23) on the same interfaces. The Dismal model performs the same task in the Dismal spreadsheet (Ritter and Wood, 2005; Paik et al., 2015; Tehranchi and Ritter, 2018a,b). The Error model is developed to investigate errors and error correction behaviors in the Microsoft Excel environment. With a model that can make predictions of usability and correlate with human data very well, we then develop an Error model to move closer to having a complete user model that can predict more human behavior while completing a task, including the initial error generation, as well as the restorative steps of detection and error correction.

We conducted a study asking participants to perform the Kim (2008) spreadsheet task in Excel. The Excel version has identical subtasks but with different keystrokes and interpretations of the visual display (Kim, 2008; Paik et al., 2015). Our study explores visual attention and errors in developing a user model. The mean task completion time of 23 users performing the KST in Excel is presented and compared with the model's predictions. The results compared to published data show that the Excel model is faster, which matches the models' predictions that users will take less time to complete the task in Excel than in the Dismal spreadsheet.

We also present further analysis of input errors. While the time predictions correlate very well between the model and data, errors are seen in the data but not in the model. Landauer's (1987) analysis suggests that perhaps 35% of the users' task time can be attributable to error generation and recovery. This suggests that analysis of the source of errors and their correction is important, and that error generation and recovery need to be included in user models.

Spreadsheet errors can have costly, even detrimental, consequences (Fisher et al., 2006). Powell et al. (2008) provide a comprehensive review of different types of errors, their frequency, and strategies to avoid errors through proper spreadsheet design. Panko and Aurigemma (2010) further refine the understanding of spreadsheet errors by revising the Panko–Halverson taxonomy, which classifies errors based on their nature and severity. Panko (2016) explores the cognitive analysis approach to understand spreadsheet errors which include focusing on how users think, make decisions, and process information while working with spreadsheets. All these processes can lead to error generation in spreadsheet tasks.

Furthermore, Kankuzi and Sajaniemi (2013, 2014) have shown that experts employ unique mental models when interpreting spreadsheet data, identifying errors, and rectifying them. Cunha et al. (2018) leveraged these findings by incorporating different types of error proposed by Bishop and McDaid (2008) to create a user-friendly interface aimed at facilitating spreadsheet authors in comprehending errors and correcting them in spreadsheets created by other authors. Recent research, such as Huang et al. (2020), has focused on developing new techniques like WARDER for effective spreadsheet defect detection by utilizing validity-based cell cluster refinements. Chalhoub and Sarkar (2022) investigate the user experience of structuring data in spreadsheets and provide insights into how users think about data organization, which can help inform future spreadsheet design improvements.

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press

Despite the extensive body of research available on spreadsheet errors, once again, the literature on utilizing cognitive architecture to model the cognitive process of human error in spreadsheet tasks remains inadequate.

To start to account for errors in cognitive architectures, a simple Error model is developed to demonstrate how a model could generate errors. This model starts to show how time can be allocated to error detection, error correction, and different types of knowledge.

JSegMan

If cognitive models interact with user interfaces, then the models will be easier to develop and apply. This approach was called a Cognitive Model Interface Management System (CMIMS; Ritter et al., 2001), which is an extension of the concept of a User Interface Management System (Myers, 1995). The CMIMS's research area can be exploited to help develop cognitive models and agents and support the users who interact with interfaces. It will also help elevate testing user interfaces, making this process more approachable and easier to do so that the testing can be successfully completed. Using the model to test interfaces and even systems as they are built has been called for by the National Research Council in a report about how to build large systems (Pew, 2007).

Multi-disciplinary aspects of artificial intelligence, computer vision, cognitive science, and psychology can be utilized to better shape human-like agents by incorporating new capabilities. JSegMan explores how the phenomenon of vision and motor knowledge can be incorporated into a novel, improved user modeling approach.

JSegMan was developed for cognitive models to provide them with access to the world. This is an important component for nearly all models of science so that they are completer, and for all applications where it would be useful for a user model to interact with an existing, but the unmodifiable interface (Ritter, 2019; Wallach et al., 2019). JSegMan takes some ideas from SegMan (Ritter et al., 2007; St. Amant et al., 2007), extends them, and moves the system into Java. The only other software that can interact with interfaces, like JSegMan, requires the interfaces to be built within specific tools (Ritter et al., 2000; Byrne, 2001; Wallach et al., 2019).

We start by describing the eyes and hands in more detail. JSegMan is a simulated eyes and hands tool that allows an ACT-R user model to receive input by reading the screen and produce output by passing commands to the operating system to execute actions such as a mouse click. JSegMan has been used to interact with the Dismal spreadsheet (Kim, 2008; Paik et al., 2015). Using JSegMan, we found missing knowledge that needed to be included and adjusted hand positions that were not considered. The revised model could complete the task (Tehranchi and Ritter, 2018a,b). JSegMan was recently revised to see objects on the screen in a more powerful way. The new vision simulation is based on the Pyramid Template-Matching algorithm (Bradski, 2004) that reduces visual pattern resolution and size sensitivity (i.e., templates).

JSegMan thus helps realize a long-held desire to use human performance models more directly in computer-aided engineering. The models are either used to test an existing system or are developed with the system as it is developed as a way to represent users as an important system component.

Research objectives

The research objectives of this work are to provide a novel approach to improve and integrate cognitive models by facilitating how cognitive models interact with the world and, in this regard, quantitatively include the perception and actions of cognitive models.

First, interaction with an executive platform of models will be investigated with the ACT-R architecture. Second, a new type of knowledge and skills with respect to interaction will be modeled to explore the necessity of new knowledge for interaction. Third, we will examine if including the new knowledge leads to better prediction of the performance time. There are three goals to achieve these objectives.

Evaluating user interfaces: A complete model of a spreadsheet task in two environments (Dismal and Excel) will be developed to illustrate the applicability of this approach in the real-world task of evaluating two interfaces. We will examine if and how cognitive models can help design and evaluate two spreadsheet applications. Which interface is faster to use and why? Which requires less knowledge? How many keystrokes/mental operators are needed? What could be done to speed up either interface? To answer these questions, we will develop a complete model to interact with two unmodified interfaces to compare them.

Modeling error detection and correction: We will also examine how adding the capability of making errors improves cognitive models. What are some of the mechanisms and user knowledge needed for modeling errors, and how do they contribute to human performance modeling? To answer these questions, we will develop an error model to simulate a user's errors.

Where does the time go? We will investigate and break down the time allocations among vision and motor knowledge to understand the differences between model predictions and human data.

In summary, our research questions (RQs) are:

RQ1: Could a user model be demonstrated on two different interfaces and make useful predictions about performance on these two systems?
RQ2: Could cognitive user models be extended to include error detection and correction?
RQ3: How can the differences between model predictions and human data be improved?

Method

We gathered data to test the model's predictions for Excel. We used existing data (Kim, 2008; Kim and Ritter, 2015) to test the model's predictions for the task using the Dismal spreadsheet.

Kim spreadsheet task

Figure 1 shows the Excel task environment and the final state of the KST. The KST was created to study learning and retention (Kim, 2008). Participants start with the initial file for the KST and complete the 14-subtasks of the KST. Table 1 lists the 14 subtasks of the KST. These subtasks are similar to regular computer tasks in spreadsheet tools. They are more complex than common text-typing tasks because of the included logic and patterns. In the 2008 study, participants returned in four consecutive days to repeat the task and then came back in 6, 12, or 18 days.

The Dismal spreadsheet of Emacs was carefully chosen because no participants had prior knowledge of the task environment. Both that task and task environment have been good candidates for investigating learning and retention. The published data contain participants' subtask completion times.

Kim's (2008) data indicated the existence of errors. But, errors were not recorded and considered for analysis. KST is a good candidate for model development and further analysis, because

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press

Subject				
9/17/22				
Command Name	Frequency	Normalization	Length	Typed Characters
log	20	14.38848921	3	60
learn	6	4.316546763	5	30
excise-chunks	12	8.633093525	13	156
excise-task	5	3.597122302	-11	-55
go	23	16.54676259	2	46
help	19.043	13.7	4	76.172
excise-all	6.95	5	10	69.5
load	9.035	6.5	4	36.14
excise	14.039	10.1	6	84.234
time	24.047	17.3	4	96.188
Total	139	100		
Your Total	139.114	100.0820144	40	599.234

Figure 1. The final task state for the Excel version of the Kim spreadsheet task. Participants open a file, save it with a new name, add new rows, and calculate the value of 36 cells.

Table 1. The 14 subtasks of the Kim spreadsheet task. These steps are performed in both interfaces, the Dismal spreadsheet and Excel

	Subtasks
(1)	Open File
(2)	Save As
(3)	Calculate Frequency (B7 to B11)
(4)	Calculate Total Frequency (B14)
(5)	Calculate Normalization (C2 to C6)
(6)	Calculate Total Normalization (C14)
(7)	Calculate Length (D2 to D11)
(8)	Calculate Total Length (D14)
(9)	Calculate Typed Characters (E2 to E11)
(10)	Calculate Total Typed Char. (E14)
(11)	Insert Two Rows
(12)	Type in Name (A1)
(13)	Insert Current Date (A2)
(14)	Save As…

the resulting models based on KST support investigations of different types of skills and can be run with the two spreadsheet tools (i.e., Dismal and Excel). Their predictions for each of the 14 subtasks could be computed for each of the two interfaces.

Participants

Twenty-three undergraduate and graduate students at The Pennsylvania State University (Penn State) were recruited to participate in this experiment. Participants were paid 12 dollars for a study session, and all successfully completed the study. Table 2 summarizes the demographic information of the participants.

In addition, six Penn State students participated in a pilot study. Their results are used to update the instruction and determine the doability of the instruction. Pilot study results suggested that we needed to modify the task instructions and initial file to be performed in Excel. Also, we learned to disable formula autocomplete, background error checking, auto-complete for cell values, and disable the formula bar to limit participants' options and make Excel more similar to Dismal.

Table 2. Participants' demographic information, including the self-reported level of competency in using Microsoft Excel, gender, and age

Demographic information		
Level of competency in using Microsoft Excel (self-reported)	Basic	4
	Intermediate	17
	Advanced	1
	Never use Excel	1
Gender	Female	5
	Male	18
Age	Average	25.13
	Standard deviation	5.87

Materials

An initial file was created in Excel for participants to complete the 14 subtasks of the KST, similar to data gathered in a previous study of the task (Kim, 2008). A regular mouse and keyboard were used to input commands. Excel was used because of its popularity. Students are more familiar with Excel than Dismal. Excel is also an example of an application showing our approach's possibility (i.e., eyes and hands models). Using Excel also allows us to illustrate the use of JSegMan to compare the usability of two interfaces and show how JSegMan interacts with a commercial interface. Users in our study were familiar with the Excel environment, but users in Kim's (2008) study did not know the Dismal spreadsheet. In both environments, users had to study the task, the initial file, and, to a slight extent, the interface.

To make the Excel environment more similar to the Dismal environment and to reduce interaction with previous files, these features were turned off in Excel: formula autocompletes, formula bar, cell values autocomplete, and background error checking. Also, the Excel environment is more visually cluttered than Dismal. Therefore, the Excel ribbon was collapsed to make it simpler. Figure 2 shows the study environment with two Excel windows; the left window is the task environment, and the right window is the instruction window.

As shown in Figures 1 and 2, the initial Excel file consists of five columns (A–E). Column B has frequencies of each command listed from rows 2 to 6. Column C has normalized frequencies listed in rows 7–11. Five blank cells are filled in by participants in the B and C columns (e.g., B7–B11 and C2–C6). Columns D and E had 10 blank cells that are filled in by participants. The total of the frequency column (rows 2–11) and the normalization column (rows 2–11) are provided to the participants. The total frequency is used to calculate normalization and frequency.

None of the participants had prior knowledge of the details of the KST. An eye tracker recorded keystrokes, mouse clicks, mouse movements, and task completion time (Lankford, 2000).

Design

In this within-participants study, the subtask times are the dependent measure. All participants used menu-based commands with a mouse and keyboard. Participants were not allowed to use key-based commands (e.g., keystroke accelerators such as CTRL+C for copying

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press

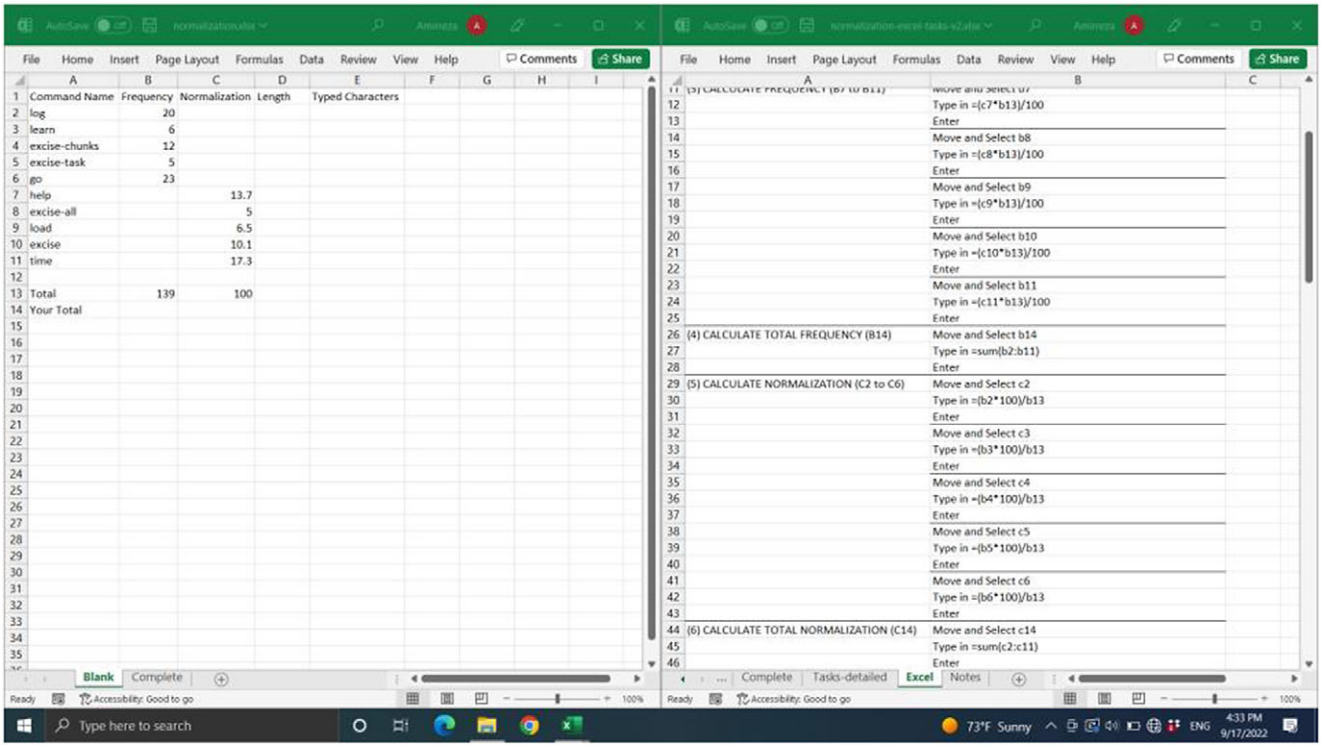

Figure 2. Screenshot of the study environment containing two Excel windows. The initial state (for Excel) is on the left side of the screen, and the KST instructions for Excel are on the right side.

text). Also, we limited some Excel features, such as autocomplete to make the Excel environment similar to the Dismal environment.

Procedure

Participants were seated at their comfortable reading distance. They were welcomed, and the welcoming script was presented. They first went through eye-tracker calibration. Participants were instructed to use the two Excel windows (Fig. 2): one for completing the task and the other containing the instruction. Participants then started to complete the 14 subtasks using the instruction while being eye-tracked. They opened the initial state spreadsheet and saved the file as another name. They then completed the 14-step task by calculating and filling in blank cells using equations, including 5 data normalization calculations, 5 data frequency calculations, 10 calculations of length, 10 calculations of total typed characters, 4 summations of each column, and finally an insertion of the current date using an Excel command, ("=today()"). Finally, participants were asked to complete a questionnaire that included their demographic information and Excel literacy level.

Results

In this section, we report our results. We start with the evaluation of the two spreadsheet tools. We then look at the error data and the error model. Finally, we investigated time in models and human data.

Evaluating user interfaces

All 23 participants completed their session. As a dependent variable, the task completion time was recorded in milliseconds by the GazeTracker eye-tracking system (Lankford, 2000).

The average task complication time was 717.85 s [standard deviation (SD) = 177.89]. The GazeTracker data for each trial have approximately 10,100 records (data points). Pupil data were recorded in .25 s periods. All data points recorded within the given time period must change by at least the minimum change amount (5 hits, a measure in the eye tracker) and cannot change by more than the maximum change amount (50 hits).

The Excel model was developed using the ACT-R architecture with updated JSegMan. This model uses 48 unique visual objects (VO), 29 production rules (PR), and 732 declarative memory chunks (DM). The model that makes predictions for this task on the Dismal interface (Tehranchi and Ritter, 2018a) has the same number of PRs but larger, VO = 52 and DM = 1,159, sets of knowledge of the interface elements than the Excel model. The Excel model interacts with the same KST initial Excel file that the participants used (Fig. 2).

Figure 3 shows a comparison of the Excel model subtask time predictions and participants' data, resulting in a correlation (Pearson correlation coefficient) of r = .84 (r^2 = .71) and mean square error (MSE) = 792.6 (N = 10, Average SD < 2). Pearson's r indicates that there is a strong linear relationship between the model predictions and the 1–14 subtasks of human data. Thus, the ACT-R model's time predictions correlate fairly well with the performance of all subtasks in Excel. The Excel model predicts noticeably more response time for subtasks containing typing formulas. The Excel model has to exactly follow the instructions because it is built based on the knowledge of the task. However, participants can understand the pattern in the formula and perform without following the instructions directly.

In contrast, the Excel model predicts less performance time for the first two subtasks. For instance, the second subtask is "save as" and contains: go to and select file, go to and select save as, go to and

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press

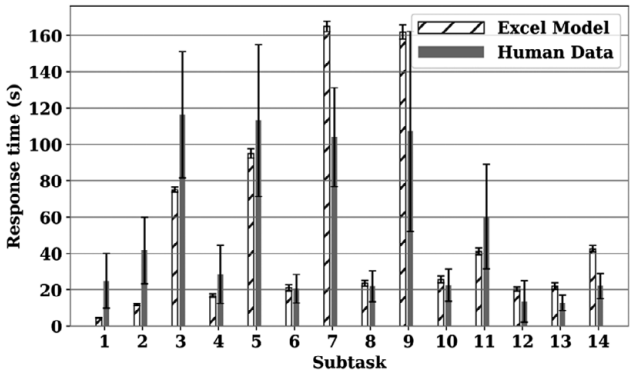

Figure 3. Average time per subtask for participants ($N = 23$) versus Excel model ($N = 10$) with error bars (SD). SD on the Excel model is smaller than the participants.

select file name area, type subject, and enter. Modeling visual behavior and visual search is straightforward with the help of pre-defined visual patterns in JSegMan. Although most participants have intermediate knowledge of Excel, they still find it quite challenging to complete this subtask.

The visual elements needed to perform that task are different between the models for the different spreadsheet tools. The Excel model needed knowledge of 48 visual objects. The Dismal model needed 52 visual objects. The Excel model has to know less visually, and presumably, so do users.

A paired-samples *t*-test is used because each row in the dataset corresponds to one subtask. The *t*-test reveals that we do not have enough evidence to conclude the means of the Excel model and participants' data are significantly different from each other, $t(13) = -0.2$, $p >> .05$.

The boxplots in Figure 4 show task performance for subtasks 1–14 for the 23 participants. The spread around the mean is higher in subtasks that combine motor and vision skills. Some participants understand the formula's pattern of these subtasks; therefore, they complete them faster than others who constantly check the instructions.

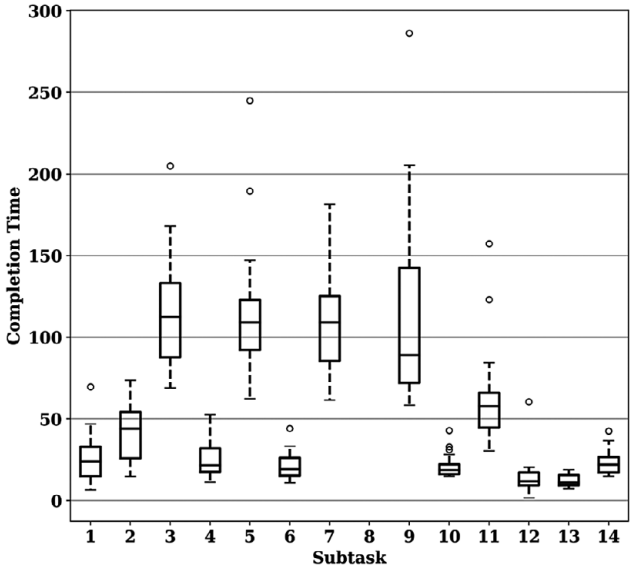

Figure 4. Boxplots showing participants' ($N = 23$) task completion time (in seconds) per subtask.

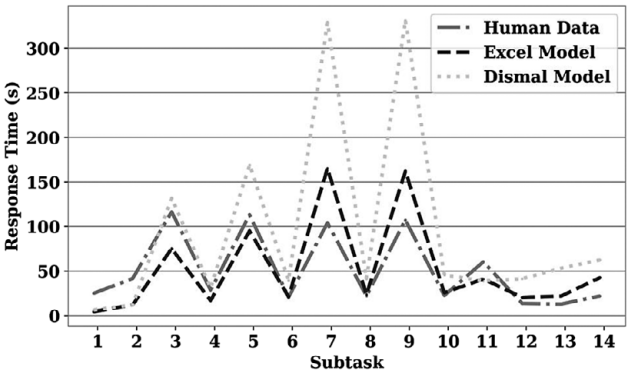

Figure 5. The models' performances on the two interfaces (Dismal and Excel) are compared by subtask. The Dismal model predicts that the Dismal interface (light gray, dotted line) is slower.

Figure 5 shows a comparison of the two models (run 10 times) and the Excel data. The models predict that it takes more time to perform the subtasks using the Dismal environment. Inspection of the models shows that this difference is because performing the same task requires more actions in the Dismal environment.

This comparison suggests that the KST can be used to evaluate the two interfaces and that performing this task in Excel is faster than in Dismal and, thus, may be seen as more user-friendly. This different response time is more noticeable in the subtasks that require inserting a cell formula. In Dismal, these subtasks require additional steps that do not exist or are not necessary for Excel.

Predicted times between the two interfaces can be tested to see how different they are. A paired-samples *t*-test is used because each row in the dataset corresponds to one subtask and is done by the same participant. The *t*-test reveals that the means of the Dismal model and Excel model datasets are significantly different from each other, $t(13) = -2.9$, $p < .05$. Therefore, a cognitive user model could perform a complex task two different interfaces and made useful predictions about performance on these two user interfaces (RQ1).

Understanding human errors

We next examine the detail of human behavior not initially seen in the models' predictions to answer the second RQ (RQ2). The Excel model correlates very well with the participants' data but has a high RMSE. These results raise questions on how the quality of cognitive models for human behavior can be increased, how keystroke errors can change models' performance, and what type of knowledge needs to be added to models when developing interactive cognitive models. We have suspicions that it will be worth investigating, including keystroke errors in the models, because typists make errors (Salthouse, 1986; Landauer, 1987), although the prediction-data fit in Figure 5 does not visually support this approach completely clearly.

Both the Dismal model and the Excel model do not predict human errors. Human errors can be defined as actions that are not suitable for achieving the desired goal (Reason, 1990; Ritter et al., 2014, Chapter 10). Building robustness to human errors in a system requires acknowledging human behavior, including errors in system design. We designed this study in a way to minimize the effect of errors. However, participants still made several types of errors.

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press

Errors are mainly grouped into three categories: skilled-based, rule-based, and knowledge-based (Brown, 2004; Ritter et al., 2014; May et al., 2019). The most common errors that are observed in participants' behavior for the KST in the Excel study are slips that are part of skilled-based errors. The KST is self-explanatory, and participants are aware of planned actions, but they execute incorrect actions such as missing keypresses. Besides, repetitive actions cause lapse errors that are due to the inability to correctly remember the stage of planned actions, such as skipping a step in the instruction and also entering the wrong key.

Violations are intentional mistakes that can be grouped into three sets of routing, situational, and exceptional. It is important to note in some studies, violations are also considered as a type of error (Lawton, 1998).

ACT-R models generally simulate error-free expert behavior. In the Dismal model, the number of mouse clicks during correct task performance is known (125 clicks). Any clicks more than 125 represent mistakes. However, Kim's (2008) analysis showed that this definition of mistake (number of clicks) did not correlate with performance time, meaning that the fastest performers also made more mistakes (clicks more than 125).

We evaluated participants' performance by comparing their final output with a correctly completed spreadsheet (as shown in Fig. 1). The score of a cell is 1 if the cell values in two documents are equal. Therefore, the perfect score is 80 when all the cells' values are equal in the two documents because 80 cells exist in the KST table. Figure 6 presents the participants' scores on a scale of 1–80. Eighty-three percent of the participants have a score higher than 70. But, Figure 6 shows that errors remain in the majority of the results.

In the Excel study, participants made two types of errors: (a) multiple unintentional actions in the form of slips and lapses such as typos and (b) multiple intentional actions in the form of violations (deliberately doing the wrong thing), such as using CTRL +Z (undo) when asked not to use Excel shortcut keys in the study instructions and which was not taught.

Error detection and correction

There have been limited attempts to understand and simulate error detection and correction in cognitive models and architectures. A sample study investigates error recovery in touch screen environments (Goodwin et al., 2016). Their work suggests identifying strategies that occur naturally during error recovery and evaluates their occurrences in low-, medium-, and high-error environments. Also, this work proposed that errors mainly happen while searching for a target, shifting attention away from the target, and choosing the following action.

In the Excel study, we observed that participants used the primary detection mechanism, which is self-monitoring. We chose to explore participant 19's behavior as an example in detail to investigate error detection and correction. This participant's score is near 80, and this participant made skill-based errors in some subtasks that affected the final score but did not make any rule-based or knowledge-based errors. This choice is arbitrary, and we believe that other participants could have been chosen and led to similar results.

Participant 19 made 25 slips. Twenty-four slips were detected and corrected using backspaces. Only one slip, typing "*" instead of ":", was not detected and therefore was never corrected by the participant. This participant spent 17.6 s on detecting and correcting errors, which is about 3% of the time spent on completing the task.

The Excel environment assisted participants in detecting errors by highlighting the selected cells in formulas, pop-up error notifications, and formula format notifications. We disabled the autocorrection and filling in Excel, so participants needed to correct their errors manually.

Modeling error detection and correction

We developed an Error model that produces errors. The Error model is an Eyes and Hands model that generates errors and error corrections. The Error model injects participant 19's errors and

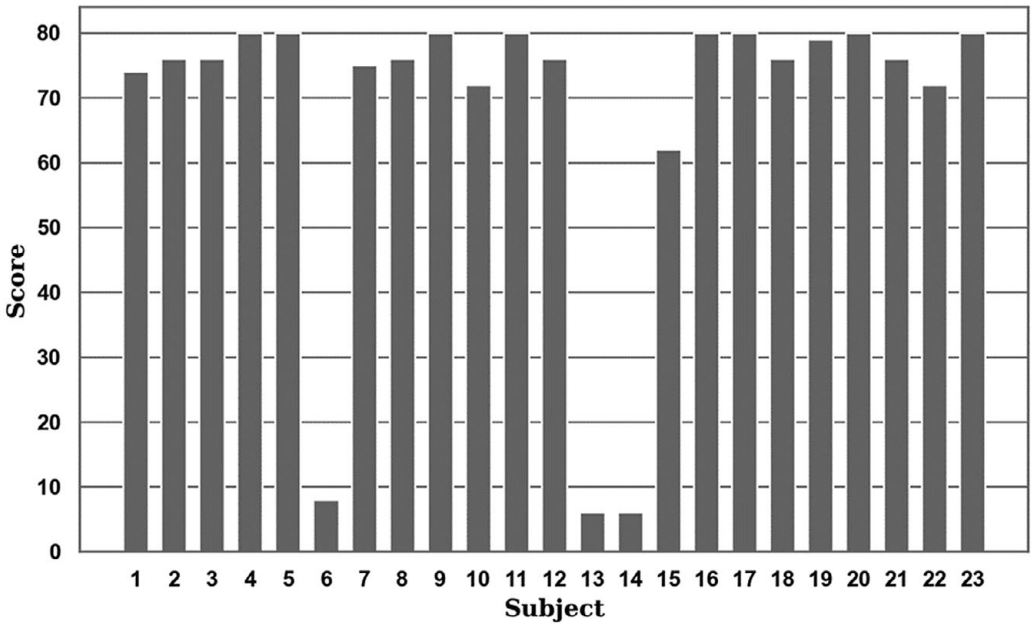

Figure 6. Number of cells (out of 80) correctly filled in. The average score is 67.39, and the SD is 24.39. Participants 6, 13, and 14, respectively, completed the task in 595.61, 581.66, and 1,075.14 s. Participant 14 skipped subtask 8.

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press

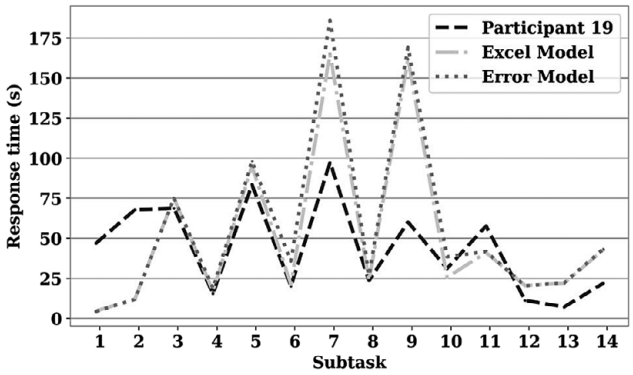

Figure 7. The Error model predicts more time for performing subtasks than the Excel model (without errors). This can be expected, given the new additional steps of error correction.

corrects them in the same locations and order the participant did. But this built-in error-generating and error-correcting model also can be extended to inject errors randomly. New declarative chunks are added for new pressing key actions in the Error model. The Error model has 777 declarative knowledge chunks, which is 45 chunks more than the Excel model, including extra chunks (memory items) to implement backspaces as a correction strategy.

The Error model's predictions are compared with both participant 19's behavior and the Excel model's predictions. Figure 7 shows this comparison. The Error model and Excel model's performances are summarized in Table 3. Both models correlate very well but have a very high MSE and RMSE. The Error model does not correlate well with the time predictions, but it includes an important aspect of behavior. Further refinements are required that will be likely to make it better besides including errors and correcting errors and can make more accurate time predictions.

Where does the time go?

These results raise a new RQ (RQ3) because the model's correlation with performance time has not been improved despite adding errors. The Error model does not better fit the timing data but now predicts errors and their correction. In some subtasks, the Error model predicts more time than the Excel model, such as in subtask 6 that the Error model detects and corrects seven slips. Thus, just adding users' errors will not improve the model's time predictions. Investigating the proportion of time spent and predicted can answer where the time goes.

Figure 8 shows the model's prediction time and participant 19. Each subtask time is divided into three main categories: vision knowledge, motor knowledge of mouse movements, and motor knowledge of keypress. Subtasks 1, 2, 7, and 9 in Figure 8 illustrate the noticeable differences between the Error model predictions and participant 19.

The recorded video and eye movement data of participant 19 show that for subtasks 1 and 2 (as shown in Table 1), more time was spent searching for visual objects. Therefore, the Error model underestimates the needed time for these two subtasks. In contrast, for subtasks 7 and 9, participant 19 could estimate the location of the following visual object (e.g., the cell below) without searching. The ACT-R visual module shifts attention at a constant rate for all objects. To include the additional search time, the retrieval time of visual objects for subtasks 1 and 2 are increased from the default .05 s to 5, 42, and 43 s for the three difficult-finding visual objects to fit the human data. The retrieval time remains the same for other subtasks. The adjusted Error model reflects the new retrieval time. Table 4 demonstrates the adjusted Error model improves performance (correlation and error rate).

Table 3. Error model (N = 5) and Excel model (N = 10) compared with participant 19's performance by subtask

	Correlation	MSE	RMSE
Error model	.678	1,877.734	43.33
Excel model	.689	1,482.701	40.25

ACT-R estimates more time for subtasks 3–9 and 12–14's completion time than the time participant 19 spent on these subtasks. There is a noticeable difference between ACT-R estimated times and the time spent by the user for actions within each subtask. Actions are movements that their aggregation completes the whole subtask, such as shifting attention, moving a mouse, click, and pressing a keystroke. The time spent by participant 19 to switch among actions is significantly shorter than what ACT-R predicts (i.e., the retrieval time of chunks needs to be adjusted). The retrieval time in ACT-R is calculated by the following formula:

$$RT = Fe^{-(f*A_i)}, \quad (1)$$

Where parameter F is the *latency factor parameter* (represented by *:lf* in ACT-R). In the original KST experiment (Kim et al., 2007, 2008), this parameter was set to .31. The latency factor in the modified Error model is set to .2, which significantly improves the estimated time for subtasks 5, 12, and, 14. There is a minor decrease in the accuracy of the estimated time for subtasks 10 and 11. However, the improvement in MSE and r^2 suggests an overall improvement in the model.

Discussion and conclusions

We have demonstrated how to test interfaces without modifying them using a model that interacts with them. We used an artificial eye and a simulated hand to develop Eyes and Hands models. JSegMan was used to develop Eyes and Hands models (Tehranchi and Ritter, 2018a). It performs a sample task, KST, in two spreadsheet tools: Dismal and Excel. The KST contains 14 non-iterative subtasks, each of which has multisteps and is complicated and long enough to be modified to support investigations of different skills.

The revised KST task allowed us to examine two sets of knowledge and skills: procedural/ declarative and cognitive/ perceptual-motor skills. We used published data on the Dismal interface and conducted a new study with Excel. The Excel model was compared with the previous Dismal model. The Excel model predicted human performance well, r = .84 and MSE = 792.6. The predictions were supported with human data from performing the KST with an uninstrumented spreadsheet (Excel). Both theoretically and experimentally, performing the KST in the Dismal environment takes more time than performing the KST in the Excel environment.

This result represents scientific and engineering contributions. The scientific contribution is that performance on a whole task on two interfaces was modeled without modifying the interfaces. Nearly, all models of interaction use scaffolding to see the task and to interact with the task environment. These models use a task modified for them, and interaction is based on a function call from

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press

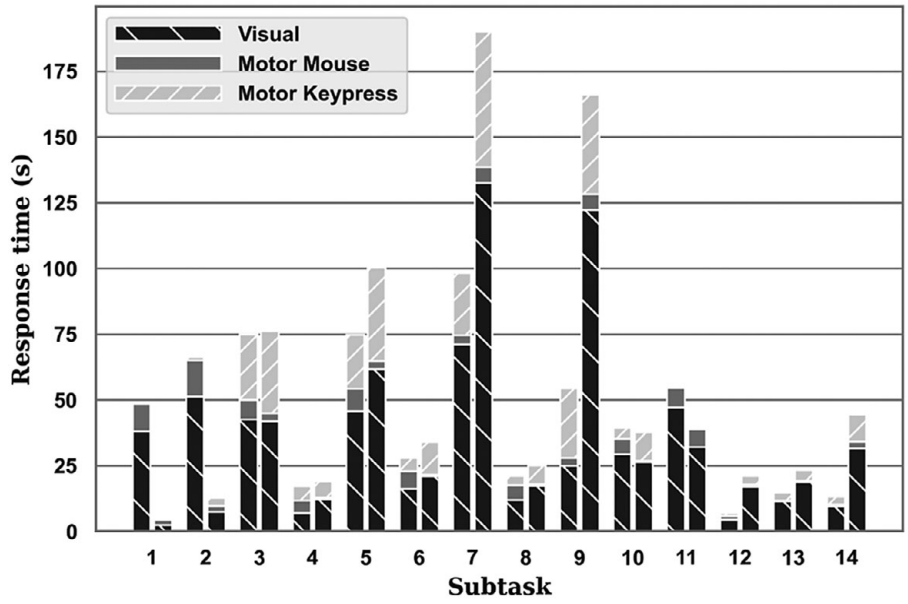

Figure 8. The response time breakdown of the Kim spreadsheet task for participant 19 (left bars) and the Error model (right bars) into three main categories: Vision, Motor Mouse (includes moving the cursor and clicking), and Motor Keypress (typing).

Table 4. Error model (*N* = 5) and adjusted Error model (*N* = 5) compared with participant 19

	Correlation	MSE	RMSE
Error model	.678	1,877.734	43.33
Adjusted error model	.840	343.377	18.53

them to the modified interface. The presented models did the whole task without modifying the task environment. The engineering contribution is that performance on interfaces doing a nontrivial 14 subtasks can be predicted. The predictions can be used to test and validate interfaces. Pew and Mavor's (2007) National Research Council report called for such user models to be created to develop, test, and design interfaces. Future work could extend the interface design in this way.

Next, the existence and impact of errors on and in performance have often been forgotten and not included in nearly all user models. In this study, we also collected new data on errors for forgetting keystrokes and missing keys to investigate errors, error type, error detection, and error correction strategies. The new error knowledge was added to the model to improve predictions and show that it is possible to start to model errors.

Participant 19 spent 3% of the task time correcting errors. This can be contrasted with Landauer's (1987) 35% of keystrokes. Clearly, there will be differences between tasks and individuals. Further work can be performed using this approach to examine errors in interface usage. Also, the Error model overestimates the typing time. For instance, participant 19 spent less time pressing a keystroke than the next keypress object that the Error model estimates to retrieve. Finger movements are included in ACT-R that are unnecessary or do not represent the user's keypress behavior. We presented an analysis of the types of errors that occurred in one participant and presented one of the first models that can generate errors.

Finally, we then adjust that model to understand where the time goes. Further analysis showed that visual skills such as visual search take longer than cognitive models currently predict. Our analysis suggests that visual search and shifting attention need to be separated and have different retrieval times. The model introduced different aspects of the relationship between time and knowledge and raised other RQs.

There are limitations to this work. The number of errors analyzed is not large; it is mostly a proof of concept. The types of errors are not very large and should be expanded by analyzing further data and expanding the mechanism for error generation and correction. Also, the demographics of the participants used to test this approach are somewhat narrow, particularly in age and gender. Future work should include a wider range of participants.

Future work will involve applying this approach to a more extensive set of interfaces. The application can serve science with more accurate models of interaction, learning, and memory because the models can use the same interface. This work provides an engineering advantage (less model to implement) and a scientific advantage (the model sees more like what humans see). The model of vision individually can be reused and supports science and engineering in this regard. This model of vision can also be extended to include the ability to learn to recognize more visual patterns.

Including interaction with uninstrumented interfaces and errors opens up a new world for predicting human behavior and applications to improve interactive design and apply intelligent agents as models of humans. In the future, we may be able to test interfaces automatically. Also, error generation and correction need not be excluded as an aspect of behavior and predictions for design. Eyes and Hands models will also help support generating automatic quantitative, qualitative, and subjective predictions of interface use. A model that interacts with an interface can start to model the time and time course of use. The time course can be compared with expectations and also moderated by the stress and novelty of the interface or task. The generation of errors provides another way to measure design performance.

Data availability statement. Models are available at https://github.com/HCAI-Lab.

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press

Funding statement. This work was supported by ONR (Grant No. N00014-15-1-2275) and The Pennsylvania State University.

Competing interest. The authors do not have any competing interests to report.

References

Alvarenga MAB and Frutuoso e Melo PF (2019) A review of the cognitive basis for human reliability analysis. *Progress in Nuclear Energy* **117**, 103050.

Anderson JR (2007) *How Can the Human Mind Exist in the Physical Universe?* New York: Oxford University Press.

Anderson JR, Betts S, Bothell D, Hope R, and Lebiere C (2019) Learning rapid and precise skills. *Psychological Review*, **126**(5), 727–760.

Bishop B and McDaid K (2008) An empirical study of end-user behaviour in spreadsheet error detection & correction. arXiv preprint arXiv:0802.3479.

Bradski GB (2004) The OpenCV Library. *Dr. Dobb's Journal of Software Tools* **120**, 122–125.

Brown AB (2004) Oops! Coping with human error in IT systems. *Queue* **2**(8), 34–41.

Byrne MD (2001) ACT-R/PM and menu selection: applying a cognitive architecture to HCI. *International Journal of Human–Computer Interaction* **55**(1), 41–84.

Byrne MD and Anderson JR (1998) Perception and action. In Anderson JR and Lebiere C (eds.), *The Atomic Components of Thought*. Mahwah, NJ: Erlbaum, pp. 167–200.

Byrne MD, Wood SD, Sukaviriya P, Foley JD and Kieras DE (1994) Automating interface evaluation. In *Proceedings of the CHI'94Conference on Human Factors in Computer Systems*, 232–237. New York, NY: ACM.

Card SK, Moran T and Newell A (1983) *The Psychology of Human–Computer Interaction*. Hillsdale, NJ: Erlbaum.

Chalhoub G and Sarkar A (2022) "It's freedom to put things where my mind wants": understanding and improving the user experience of structuring data in spreadsheets. Paper presented at the Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems.

Chen C, Reniers G and Khakzad N (2019) Integrating safety and security resources to protect chemical industrial parks from man-made domino effects: a dynamic graph approach. *Reliability Engineering & System Safety* **191**, 106470.

Chen C, Reniers G and Khakzad N (2020) A thorough classification and discussion of approaches for modeling and managing domino effects in the process industries. *Safety Science* **125**, 104618.

Cunha J, Dan M, Erwig M, Fedorin D and Grejuc A (2018) Explaining spreadsheets with spreadsheets (short paper). *ACM SIGPLAN Notices* **53** (9), 161–167.

Di Pasquale V, Miranda S and Neumann WP (2020) Ageing and human-system errors in manufacturing: a scoping review. *International Journal of Production Research* **58**(15), 4716–4740.

Elkind JI, Card SK, Hochberg J and Huey BM (1989) *Human Performance Models for Computer-Aided Engineering*. Washington, DC: National Academy Press.

Fincham JM, Tenison CS, and Anderson JR (2022). Combining EEG and a cognitive model to infer the time course of game play. In *Proceedings of ICCM, The 20th International Conference on Cognitive Modeling* (pp. 88–93).

Fisher M, Rothermel G, Brown D, Cao M, Cook C and Burnett M (2006) Integrating automated test generation into the WYSIWYT spreadsheet testing methodology. *ACM Transactions on Software Engineering and Methodology (TOSEM)* **15**(2), 150–194.

Gonzalez C, Lerch JF and Lebiere C (2003) Instance-based learning in dynamic decision making. *Cognitive Science* **27**(4), 591–635.

Goodwin PR, St. Amant R and Rohit A (2016) Towards error recovery microstrategies in a touch screen environment. Paper presented at the Proceedings of the 14th International Conference on Cognitive Modeling (ICCM 2016), Penn State, University Park, PA.

Halbrügge M (2013) ACT-CV: bridging the gap between cognitive models and the outer world. *Grundlagen und Anwendungen der Mensch-Maschine-Interaktion* **10**, 205–210.

Hope RM, Schoelles MJ and Gray WD (2014) Simplifying the interaction between cognitive models and task environments with the JSON Network Interface. *Behavior Research Methods* **46**(4), 1007–1012.

Huang Y, Xu C, Jiang Y, Wang H and Li D (2020) WARDER: towards effective spreadsheet defect detection by validity-based cell cluster refinements. *Journal of Systems and Software* **167**, 110615.

Kankuzi B and Sajaniemi J (2013) An empirical study of spreadsheet authors' mental models in explaining and debugging tasks. Paper presented at the 2013 IEEE Symposium on Visual Languages and Human Centric Computing.

Kankuzi B and Sajaniemi J (2014) Visualizing the problem domain for spreadsheet users: a mental model perspective. Paper presented at the 2014 IEEE Symposium on Visual Languages and Human-Centric Computing (VL/HCC).

Kieras DE and Meyer DE (1997) An overview of the EPIC architecture for cognition and performance with application to human–computer interaction. *Human–Computer Interaction* **12**, 391–438.

Kim JW (2008) *Procedural Skills: From Learning to Forgetting. Department of Industrial and Manufacturing Engineering*. University Park, PA: The Pennsylvania State University.

Kim JW, Koubek RJ and Ritter FE (2007) Investigation of procedural skills degradation from different modalities. In *Proceedings of the 8th International Conference on Cognitive Modeling*, 2552013;260. Oxford, UK: Taylor & Francis/Psychology Press.

Kim JW and Ritter FE (2015) Learning, forgetting, and relearning for keystroke- and mouse-driven tasks: relearning is important. *Human–Computer Interaction* **30**(1), 1–33.

Klaproth O, Halbrügge M and Russwinkel N (2019) ACT-R model for cognitive assistance in handling flight deck alerts. Paper presented at the Proceedings of the 17th International Conference on Cognitive Modeling (ICCM 2019), Montreal, Canada.

Laird JE (2012) *The Soar cognitive architecture*. Cambridge, MA: MIT Press.

Landauer TK (1987) Relations between cognitive psychology and computer systems design. In Preece J and Keller L (eds.), *Human–Computer Interaction*. Englewood Cliffs, NJ: Prentice-Hall, pp. 141–159.

Lankford C (2000) Gazetracker: software designed to facilitate eye movement analysis. Paper presented at the Proceedings of the 2000 Symposium on Eye Tracking Research & Applications.

Lawton R (1998) Not working to rule: understanding procedural violations at work. *Safety Science* **28**(2), 77–95.

Lohse GL (1997) Models of graphical perception. In Helander M, Landauer TK and Prabhu P (eds.), *Handbook of Human–Computer Interaction*. Amsterdam: Elsevier Science B. V., pp. 107–135.

May NC, Batiz EC and Martinez RM (2019) Assessment of leadership behavior in occupational health and safety. *Work* **63**(3), 405–413.

Myers BA (1995) User interface software tools. *ACM Transactions on Computer–Human Interaction* **2**(1), 64–103.

Newell A (1990) *Unified Theories of Cognition*. Cambridge, MA: Harvard University Press.

Nor AKM, Pedapati SR and Muhammad M (2021) Reliability engineering applications in electronic, software, nuclear and aerospace industries: a 20-year review (2000–2020). *Ain Shams Engineering Journal* **12**(3), 3009–3019.

Paik J, Kim JW, Ritter FE and Reitter D (2015) Predicting user performance and learning in human–computer interaction with the Herbal compiler. *ACM Transactions on Computer–Human Interaction* **22**(5), 25.

Pan X, Lin Y and He C (2017) A review of cognitive models in human reliability analysis. *Quality and Reliability Engineering International* **33**(7), 1299–1316.

Panko R (2016) "What we don't know about spreadsheet errors today: The facts, why we don't believe them, and what we need to do", pp. 1–15, Proceedings of the EuSpRIG 2015 Conference "Spreadsheet Risk Management" ISBN: 978-1-905404-52-0, arXiv preprintarXiv:1602.02601.

Panko RR and Aurigemma S (2010) Revising the Panko–Halverson taxonomy of spreadsheet errors. *Decision Support Systems* **49**(2), 235–244.

Pew RW (2007) Some history of human performance modeling. In Gray W (ed.), *Integrated Models of Cognitive Systems*. New York: Oxford University Press, pp. 29–44.

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press

Pew RW and Mavor AS (eds.) (2007) *Human–System Integration in the System Development Process: A New Look*. Washington, DC: National Academy Press.

Philippart M (2018) Human reliability analysis methods and tools. In *Space Safety and Human Performance*. Amsterdam: Elsevier, pp. 501–568.

Powell SG, **Baker KR and Lawson B** (2008) A critical review of the literature on spreadsheet errors. *Decision Support Systems* **46**(1), 128–138.

Reason J (1990) *Human Error*. Cambridge: Cambridge University Press.

Ritter FE (2019) Modeling human cognitive behavior for system design. In Scataglini S and Paul G (eds.), *DHM and Posturography*. London: Academic Press, pp. 517–525, Chapter 537.

Ritter FE, **Baxter GD and Churchill EF** (2014) *Foundations for Designing User-Centered Systems: What System Designers Need to Know about People*. London: Springer.

Ritter FE, **Baxter GD**, **Jones G and Young RM** (2000) Supporting cognitive models as users. *ACM Transactions on Computer–Human Interaction* **7**(2), 141–173.

Ritter FE, **Baxter GD**, **Jones G and Young RM** (2001) User interface evaluation: how cognitive models can help. In Carroll J (ed.), *Human–Computer Interaction in the New Millenium*. Reading, MA: Addison-Wesley, pp. 125–147.

Ritter FE, **Kukreja U and St. Amant R** (2007) Including a model of visual processing with a cognitive architecture to model a simple teleoperation task. *Journal of Cognitive Engineering and Decision Making* **1**(2), 121–147.

Ritter FE, **Tehranchi F and Oury JD** (2018) ACT-R: a cognitive architecture for modeling cognition. *Wiley Interdisciplinary Reviews: Cognitive Science* **10**(3), e1488.

Ritter FE and Wood AB (2005) Dismal: a spreadsheet for sequential data analysis and HCI experimentation. *Behavior Research Methods* **37**(1), 71–81.

Salthouse TA (1986) Perceptual, cognitive, and motoric aspects of transcription typing. *Psychological Bulletin* **3**(3), 303–319.

Salvucci DD (2009) Rapid prototyping and evaluation of in-vehicle interfaces. *ACM Transactions on Computer–Human Interaction* **16**(2), 9.

Salvucci DD (2013) Integration and reuse in cognitive skill acquisition. *Cognitive Science* **37**(5), 829–860.

Shahab MA, **Iqbal MU**, **Srinivasan B and Srinivasan R** (2022a) HMM-based models of control room operator's cognition during process abnormalities. 1. Formalism and model identification. *Journal of Loss Prevention in the Process Industries* **76**, 104748.

Shahab MA, **Iqbal MU**, **Srinivasan B and Srinivasan R** (2022b) HMM-based models of control room operator's cognition during process abnormalities. 2. Application to operator training. *Journal of Loss Prevention in the Process Industries* **76**, 104749.

St Amant R, **Riedl MO**, **Ritter FE and Reifers A** (2005). Image processing in cognitive models with SegMan. In *Proceedings of HCI International '05*, Volume 4 - Theories Models and Processes in HCI. Paper # 1869. Mahwah, NJ: Erlbaum.

St. Amant R, **Horton TE and Ritter FE** (2007) Model-based evaluation of expert cell phone menu interaction. *ACM Transactions on Computer–Human Interaction* **14**(1), Article no. 1, 24 pp.

Tao J, **Qiu D**, **Yang F and Duan Z** (2020) A bibliometric analysis of human reliability research. *Journal of Cleaner Production* **260**, 121041.

Tehranchi F and Ritter FE (2017) An eyes and hands model forcognitive architectures to interact with user interfaces. In *MAICS, The 28th Modern Artificial Intelligence and Cognitive Science Conference*, 15–20. Fort Wayne, IN: Purdue University.

Tehranchi F and Ritter FE (2018a) Modeling visual search in interactive graphic interfaces: Adding visual pattern matching algorithms to ACT-R. In *Proceedings of the 16th International Conference on Cognitive Modeling (ICCM 2018)*. 162–167. Madison, WI.

Tehranchi F and Ritter FE (2018b) Using Java to provide cognitive models with a universal way to interact with graphic interfaces. Paper presented at the Proceedings of the International Conference on Social Computing, Behavioral–Cultural Modeling and Prediction and Behavior Representation in Modeling and Simulation, Washington DC (paper 50).

Wallach DP, **Fackert S and Albach V** (2019) Predictive prototyping for real-world applications: A model-based evaluation approach based on the ACT-R cognitive architecture. In *DIS '19: Proceedings of the 2019 on Designing Interactive Systems Conference*, 1495–1502.

Zhan Y, **Tadikamalla PR**, **Craft JA**, **Lu J**, **Yuan J**, **Pei Z and Li S** (2019) Human reliability study on the door operation from the view of deep machine learning. *Future Generation Computer Systems* **99**, 143–153.

Zhao Y and Smidts C (2021a) CMS-BN: a cognitive modeling and simulation environment for human performance assessment, Part 1 – methodology. *Reliability Engineering & System Safety* **213**, 107776.

Zhao Y and Smidts C (2021b) CMS-BN: a cognitive modeling and simulation environment for human performance assessment, Part 2 – application. *Reliability Engineering & System Safety* **213**, 107775.

https://doi.org/10.1017/S089006042300015X Published online by Cambridge University Press