Imperfect Surrogate Users: Understanding Performance Implications of Augmentative and Alternative Communication Systems through Bounded Rationality, Human Error, and Interruption Modeling

BOYIN YANG, University of Cambridge, United Kingdom
PER OLA KRISTENSSON, University of Cambridge, United Kingdom

Nonspeaking individuals with motor disabilities frequently rely on augmentative and alternative communication (AAC) systems that allow users to communicate through a text entry interface coupled with a speech synthesizer. Such systems are notoriously difficult to evaluate with end-users. However, recent research has proposed envelope analysis as a method to estimate text entry rates and keystroke savings by simulating the interaction of an expert surrogate user entering sentences on a conceptual word-predictive text entry system. While only a part of the evaluation process of an AAC system, this method enables AAC designers to benefit from quantitative insights early on in the design process. This paper extends prior work by (1) demonstrating how to incorporate natural language generation, such as sentence generation, in such analyses; (2) presenting a model of an *imperfect surrogate user* that incorporates bounded rationality, human error, and interruptions to provide a more realistic simulation of text entry behavior; and (3) demonstrating how to estimate model parameters by observing users' actual typing behavior. We validate the model with data collected from eight participants using an AAC system on a touchscreen.

CCS Concepts: • **Human-centered computing → Text input**; **HCI theory, concepts and models**; **Accessibility design and evaluation methods**.

Additional Key Words and Phrases: text entry design, predictive text entry, interactive system, computational interaction, computational modeling, user modeling

ACM Reference Format:
Boyin Yang and Per Ola Kristensson. 2023. Imperfect Surrogate Users: Understanding Performance Implications of Augmentative and Alternative Communication Systems through Bounded Rationality, Human Error, and Interruption Modeling. *Proc. ACM Hum.-Comput. Interact.* 7, MHCI, Article 213 (September 2023), 33 pages. https://doi.org/10.1145/3604260

1 INTRODUCTION

Nonspeaking individuals with motor disabilities are heavily reliant on augmentative and alternative communication (AAC) systems to communicate. Such systems provide nonspeaking users with means to communicate via a speech synthesizer. Predictive text entry AAC systems provide access techniques, such as eye gaze, dwell mouse click, touchscreen, and so on, and provide text predictions in the form of word, phrase, and sentence predictions. These features enable literate AAC users to potentially increase their text entry rates.

However, evaluating AAC systems with actual users poses a challenge since the user group is highly heterogeneous, with individual access needs, technical solutions, and personal support

Authors' addresses: Boyin Yang, University of Cambridge, United Kingdom, by266@cam.ac.uk; Per Ola Kristensson, University of Cambridge, United Kingdom, pok21@cam.ac.uk.

2573-0142/2023/9-ART213
https://doi.org/10.1145/3604260

infrastructure. This makes it difficult to adopt best practices in user interface design, such as iterative refinements of interface features, co-design, or studying long-term effects through deployment studies [21]. As a consequence, prior research has suggested the merits of adopting design engineering methods to *complement* current AAC design practice (e.g. [21]). Briefly, this involves identifying a conceptual design in the form of a parameterized function model [23], which can be subsequently analyzed using envelope analysis, which simulates system performance and thus allows designers to understand possible entry and error rates, keystroke savings, and so on, that may be conceptually possible using a particular system design. This allows AAC designers to understand likely performance implications at an early stage in the design process, serving as complementary design know-how in tandem with traditional user-centered AAC design approaches. Prior work using this approach has studied performance envelopes of context-aware sentence retrieval [21] and word prediction [23].

Recent works have begun to use natural language generation (NLG) technologies to enable AAC systems to generate entire sentences with reasonably acceptable results [6, 44]. State-of-the-art technologies, such as ChatGPT [34] and GPT-4 [35], demonstrate the potential for NLG-assisted AAC systems to further increase text entry rates. However, these approaches have not been integrated into functional AAC systems with graphical user interfaces (GUIs), and such systems are difficult to design and assess since their performance is governed by a very large number of parameters. Some of these parameters relate to the user interface, such as the number of word and sentence suggestions, some relate to the underpinning models themselves, and some are latent and relate to the strategies users adopt to optimize their performance, such as typing a few letters and then looking at a word prediction.

Kristensson and Müllners [23] highlight the substantial impact of text entry strategy on the text entry rate and keystroke savings for word prediction text entry systems. This prior work serves as the foundation for the current research. The present study aims to extend upon these findings by investigating the impact of text entry strategy on predictive systems that include both word and sentence prediction functions. The inclusion of a sentence prediction function brings additional interaction points, thereby adding an extra layer of complexity to the system. Therefore, it is essential to understand how it influences the overall efficiency and effectiveness of the predictive system.

Prior envelope analysis studies [21, 23] have estimated the upper bound of text entry rate and keystroke savings based on the assumption of "perfect" surrogate user models that can perform text entry tasks precisely, that is, they simulate error-free expert performance. As such, they do not accurately reflect actual user behavior, which limits the benefits of using a design engineering approach to guide early text entry system design.

This paper contributes to the development of design engineering methods to complement AAC design practice by making three contributions:

- We extend envelope analyses to NLG-based AAC, including Large Language Model (LLM) sentence generation, such as GPT.
- We present a model of an *imperfect surrogate user* that unlike prior work [21, 23] models users that are not behaving error-free or optimal. The model does this by incorporating three components: a bounded rationality model, a human error model, and an interruption model. These additional models capture the fact that users' rationality is bounded, users make mistakes, and users can be interrupted in their typing tasks.

- A limitation of prior work [21, 23] was that model parameters for envelope analysis had to be estimated by the designer. We present a method for estimating such parameters for actual user behavior at runtime and use this method to validate our model with eight users.

1.1 Paper Structure

This paper has two main objectives: (1) to extend prior work [23] of a conceptual design of word predictive text entry system to a text entry system with word and sentence prediction functions; and (2) to propose and apply the *imperfect surrogate user model* to perform envelope analysis and estimate parameters from actual user data at runtime. To achieve these objectives, the rest of this paper is structured as follows. First, we review prior work in AAC system design, bounded rationality, human error, and interruptions for text entry modeling. Second, we present a function structure model for the design of a word and sentence predictive text entry system for AAC. We calculate the upper bounds on error-free and optimal expert text entry rate and keystroke savings for both able-bodied and AAC users based on parameters obtained from the literature. Next, we introduce the *imperfect surrogate user model* and use it to carry out envelope analysis to understand the potential performance impact of incorporating human performance factors along with text entry strategies on text entry rates and keystroke savings. We then explain how to estimate parameters for the model by observing actual user behavior. We use this method to validate the imperfect surrogate user model with data collected from eight participants using an AAC system on a touchscreen tablet PC. Finally, we discuss the implications of this work and conclude.

2 RELATED WORK

The literature has long considered approaches for evaluating text entry methods (e.g. [59]), such as expanded rehearsal interval training [58], representative stimulus sentences [28, 50], stimulus sentence presentation styles [24], and composition tasks [14, 51]. It has also been recognized that for text entry methods to be successful, they need to consider wider issues beyond merely improving entry and error rates (e.g. [19, 20]). However, unlike other text entry domains, AAC also brings its own unique design issues.

2.1 Challenges in AAC Systems Design

The study of AAC has always been challenging. In general, the demand for research-driven technological development is enormous, especially in obtaining insights from the processes underpinning basic cognitive, motor, sensory-perceptual, and linguistic functions and utilizing them to maximize human-computer interaction efficiency through the implementation of AAC devices and methods [22, 25]. Moreover, the lack of researchers, engineers, and technical developers [30] results in a large number of unanswered questions and technical problems [55], especially when AI technologies, such as NLG models [6, 44], are involved as design materials.

User experience studies of AAC systems often employ qualitative empirical approaches, such as field studies, which can take many weeks to several months to produce outputs [4, 29]. Other methods of evaluating user experience involve questionnaires [33]. However, such post-hoc evaluation methods may fail to capture immediate feedback on user experience or aspects not listed in the questionnaire. For example, Black et al. [4] point out that users do not always select the correct prediction once it appears on the system, however, the researchers fail to understand users' intention behind this action. Besides, it can be challenging for AAC users to think aloud while using the system. Video analysis can capture the entire interaction process [53] and may provide insights into the intention behind user actions. However, this approach is time-consuming, requiring researchers to analyze the video frame-by-frame. Hence, efficient methods to understand and identify AAC user performance for iterative improvement of NLG-based AAC systems are still lacking.

2.2 Computational Models for Text Entry

The idea of viewing interaction through the lens of a computational model is not new but has recently been invigorated through the establishment of computational interaction [36] and the development of new mathematical tools to model interaction, such as Bayesian methods [54]. The concept of computational models for text entry, commonly used in the design of general text entry systems, can also assist in the design of AAC systems without the extensive involvement of AAC users. These text entry models typically focus on two main directions. The first direction is related to Fitts' law (or FFitts law [3] for touchscreen-based research), which has been extensively researched for non-predictive text entry, modeling user typing speed on different keyboard layouts using different typing methods, such as two-thumb text entry on mini-QWERTY mechanical keyboards [9], stroke-based OPTI II soft keyboards [40] and stylus-based QWERTY soft keyboards [48]. These models quantitatively simulate the time cost of each click or gesture stroke movement from one key to another, taking into account the distance of each movement and the interaction methods.

The second direction of text entry modeling focuses on predictive text entry features, which heavily involve decision-making processes. Instead of focusing on calculating the time cost of the finger moving between keys via Fitts' law based models that can be impacted by system layouts and typing methods, these studies [21, 23] investigate how predictive text features can increase or decrease the text entry rate and keystroke savings at a function level.

It is particularly important to investigate complex systems at a *functional level*, understanding how multiple functions and interaction points in a complex system mutually impact user performance and lead to different system efficiency and effectiveness. For example, research questions could be what is the best time for a user to check word predictions, and when should a user give up on word predictions [23]. Every keystroke takes time from users, which is particularly important to consider in the case of AAC users. The trade-off here is that, although correct predictions can save valuable keystrokes for users, having the user checking predictions generated from too little user input may cost the user extra time, as further user input is required to generate the user's expected predictions. Hence the goal is to type the correct text with minimum effort using prediction functions. This is a typical task analysis (TA) issue, which is at the heart of this paper.

Design researchers have been building user models for TA, such as KLM (Keystroke Level Model) [7], MHP (Model of Human Processor) [57], and GOMS (Goals, Operations, Methods, and Selection rules) [8], using psychological theories and simulation modeling since at least the 1980s. These interaction models investigate how users reason and make decisions when using complex interfaces, with the intention to allow different design elements or design configurations to be tested prior to the development of a working system or before carrying out user studies [27, 41, 43].

In this vein, Kristensson and Müllners [23] propose a computational model at a functional level, including three parameters about text entry strategy, to simulate and analyze the impact of text entry strategies on text entry rate and keystroke savings in rational and error-free settings. This model enables an explanation of the *mechanism* for why word prediction is typically not useful for an able-bodied user.

However, these studies share a common limitation in that they assume text entry is error-free and that the surrogate user is an expert, which may not necessarily reflect reality.

2.3 Human Performance Factors in Text Entry

Empirical studies in bounded rationality [12, 17], human error [11, 13, 24], and interruptions [5] have shown that such human factors concerns have a negative impact on a user's text entry rate.

The concept of bounded rationality is derived from behavioral economics and public policy for decision-making [46]. The main assertion of bounded rationality is that people, limited by time, knowledge, and resources, make satisfactory decisions instead of maximizing utility [45]. Specifically, Quinn and Zhai [38] note that text entry suggestions come with a cognitive cost, while Sarcar et al. [42] adopt a computational rationality model to develop an ability-based optimization text entry system for smartphones.

Interruptions frequently occur in daily life. Pielot et al. [37] report that on average, participants received 63.5 mobile notifications per day, such as messages and emails. Borst et al. [5] propose an interruption model of memory-for-problem-states verified with text entry experiments, which integrates three factors: interruption duration, interrupting-task complexity, and moment of interruption.

Human error is one of the most common human factors concerns in text entry and is more thoroughly studied than other factors. For instance, the autocorrect function is designed to reduce the impact of human errors. Further, as for example Banovic et al. [1] point out, making typing errors when entering text is inevitable, and correcting errors is time-consuming. As a result, typists may slow down their speed to reduce typing errors. Accordingly, Banovic et al. [1] propose a computational model to estimate the effects of risk aversion to errors on expert typing speeds for QWERTY mobile touchscreen keyboards with or without autocorrect [2].

In summary, prior research has demonstrated that bounded rationality, human error, and interruptions are three significant human factors concerns that adversely affect text entry performance. Computational models have been proposed for general human-computer interaction tasks, with each model focusing on a specific human factor. Moreover, specific computational models for text entry have been proposed to estimate the upper bound of expert text entry rates under error conditions. However, there is currently no computational model that integrates all three factors to estimate the performance of non-expert users for word and sentence prediction systems, which is particularly important for AAC design.

3 FUNCTION STRUCTURE MODEL: TEXT ENTRY STRATEGIES FOR PREDICTIVE AAC SYSTEMS

The function structure model allows designers to understand system functions and data flows between functions. This can then be used to derive a human-computer interaction flowchart for envelope analysis [23]. We adopt this model to illustrate the function descriptions of a predictive AAC text entry system. To be more specific, the overall function `Generate Sentence` is decomposed into six main functions: `Type Key`, `Predict Current Word`, `Predict Next Word`, `Select Word Prediction`, `Predict Sentence`, and `Select Sentence Prediction`. These functions are connected by signal flows represented by text with different fonts and different types of lines (see Figure 1).

The signal flows provided by the system or the user are categorized into four different types. First, the user input-related signal flows, including **Key Press**, **Word Selection**, and **Sentence Selection**, are the user's physical actions. Second, Word Hypotheses, Sentence Hypotheses, and Observation are the user's mental actions. Third, *Language Context* is the text prediction-related internal information defined and generated by the system, such as language models and machine learning algorithms. Fourth, ***Word*** and ***Sentence*** are system-predicted text selected by the user.

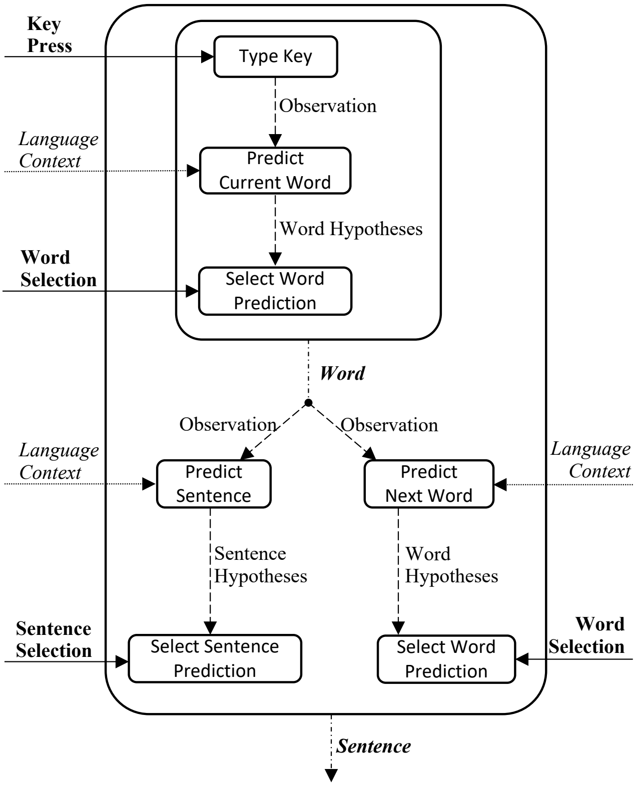


Fig. 1. Function structure model for NLG-based AAC text entry systems with word and sentence prediction functions. The fonts indicate different element types. `Sans serif` text in rounded rectangles indicates the main functions; **Bold** text aligned with solid lines indicates users' physical actions; Normal text aligned with dash lines represents users' mental actions; *Italic* text aligned with dot lines represents the system internal information; ***Italic and bold*** text aligned with dash-dot lines represents the system outputs.

Based on this categorization, performance analysis of this joint user-system gives rise to three user-simulating parameters and three system-simulating parameters, respectively. The user-simulating parameters define the time cost of the user's physical actions and mental actions:

Key Typing Time — T_{key} `Type Key` actions include entering letters (**Key Press**), selecting word prediction (**Word Selection**), and selecting sentence prediction (**Sentence Selection**). This parameter is determined by the time duration between two contiguous keystrokes entered by the user without any involvement of considerable mental processing time. Although this time duration can be estimated by Fitts' law [26] based on the layout of the keyboard, to simplify this model, we assume the time cost for every keystroke is identical.

Reaction Time for Word Predictions — T_{react_w} This parameter is a reflection of `Select Word Prediction`. It is a substantial time cost for mental action that allows the user to read through the word prediction list (*Observation*) and determine whether or not to select a prediction (*Word Hypotheses*). We assume the time cost is identical every time the user checks the list.

Reaction Time for Sentence Predictions — T_{react_s} Similarly, this parameter is a representation of `Select Sentence Prediction`, estimating the mental action time cost for processing

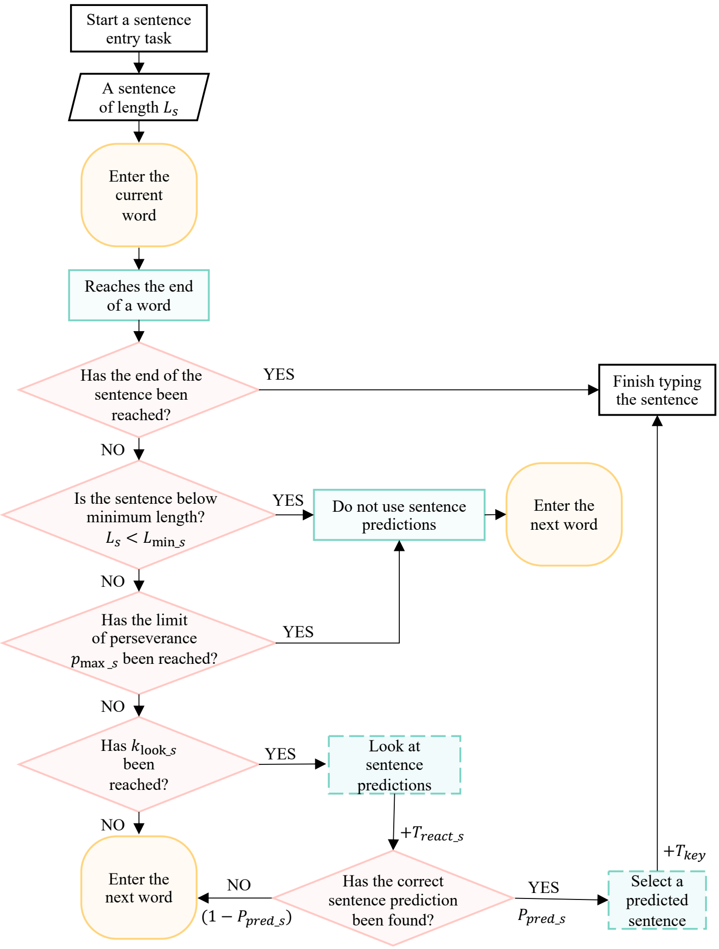

Start a sentence entry task
A sentence of length L_s
Enter the current word
Reaches the end of a word
Has the end of the sentence been reached?
YES
Finish typing the sentence
NO
Is the sentence below minimum length? $L_s < L_{\min\_s}$
YES
Do not use sentence predictions
Enter the next word
NO
Has the limit of perseverance $p_{\max\_s}$ been reached?
YES
NO
Has k_{look_s} been reached?
YES
Look at sentence predictions
$+T_{react\_s}$
NO
Enter the next word
NO
$(1 - P_{pred\_s})$
Has the correct sentence prediction been found?
YES
P_{pred_s}
Select a predicted sentence
$+T_{key}$

Fig. 2. The flowchart for the Overall Text Entry Strategy on an NLG system equipped with word prediction and sentence prediction functions. The white rectangles indicate the start and the end of a complete sentence entry task. The white parallelogram is the targeted text. The teal rectangles with solid lines show the specific status, and the teal rectangles with dashed lines are specific user actions. The pink rhombuses denote strategy decisions. The yellow rounded squares represent repeated user operations, whose details are illustrated in Figure 3 and Figure 4.

a sentence prediction (*Observation* and *Sentence Hypotheses*). As sentence length impacts reaction time, we estimate this parameter by multiplying the sentence length in words, L_s, by the reaction time for word predictions (i.e., $T_{react\_s} = L_s \cdot T_{react\_w}$). Alternatively, this parameter can also be estimated empirically via real users.

In addition, the system simulates outcomes based on *Language Context*, which defines the likelihood of prediction functions obtaining the correct text when the surrogate user inputs new text via a single keystroke (e.g. a letter, a predicted word, or a predicted sentence) in a simulated text entry task. This likelihood is the proportion of queries that yield correct predictions with a range of between 0 and 1 [23]:

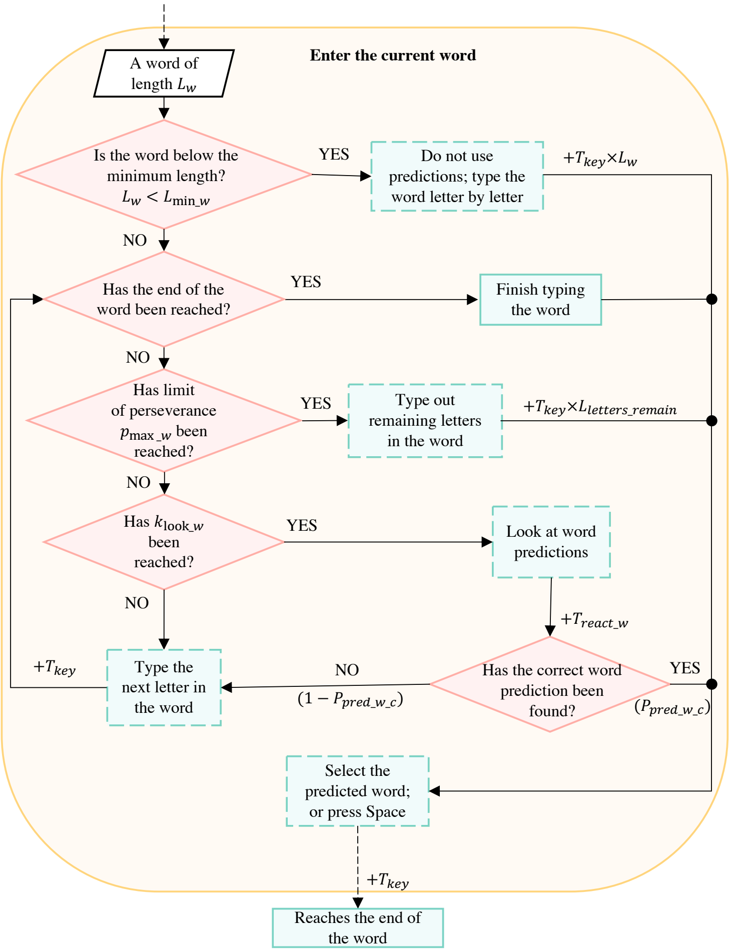

Enter the current word
A word of length L_w
Is the word below the minimum length? $L_w < L_{\min\_w}$
YES
Do not use predictions; type the word letter by letter
$+T_{key} \times L_w$
NO
Has the end of the word been reached?
YES
Finish typing the word
NO
Has limit of perseverance $p_{\max\_w}$ been reached?
YES
Type out remaining letters in the word
$+T_{key} \times L_{letters\_remain}$
NO
Has k_{look_w} been reached?
YES
Look at word predictions
$+T_{react\_w}$
NO
$+T_{key}$
Type the next letter in the word
NO
$(1 - P_{pred\_w\_c})$
Has the correct word prediction been found?
YES
$(P_{pred\_w\_c})$
Select the predicted word; or press Space
$+T_{key}$
Reaches the end of the word

Fig. 3. The flowchart for Entering the Current Word Strategy Model. It is a module in the overall text entry strategy model. The big yellow rounded square corresponds to the yellow rounded square with correlated text in Figure 2. The white parallelogram is the targeted text. The teal rectangles with solid lines show specific statuses, and those with dashed lines are specific user actions. The pink rhombuses denote strategy decisions.

$$P_{pred} = \frac{N_{success}}{N_{success} + N_{fail}} \tag{1}$$

As described in the function structure model, there are two interaction points at which prediction functions can boost text entry rate: (1) when typing a word, the system predicts the currently typed word; and (2) when a word is completed, the system predicts the next word and the entire sentence. We parameterize the current word, next word, and sentence prediction functions to accommodate various language models through three parameters:

Current Word Prediction Accuracy — $P_{pred\_w\_c}$ The current word prediction function provides a list of current entering word guesses based on the typed letters and context information, displayed as a list of words on the system. This design aims to boost the user to finish an

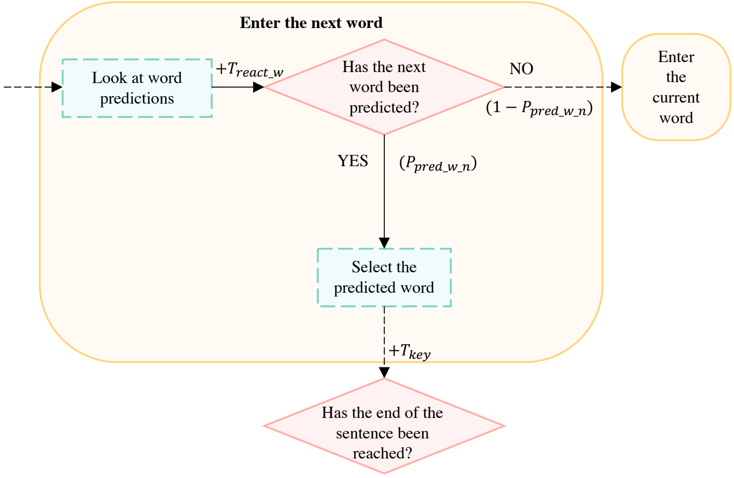


Fig. 4. The flowchart of Entering the Next Word Strategy Model. It is a module in the overall text entry strategy model. The big yellow rounded square corresponds to the yellow rounded square with correlated text in Figure 2. The teal rectangles with dashed lines are specific user actions. The pink rhombuses are strategy decisions.

expected word by entering the minimum number of letters. To simplify this model, we assign a value between 0 and 1 to reflect prediction accuracy.

Next Word Prediction Accuracy — $P_{pred\_w\_n}$ The next word prediction function predicts the next word based on previous entries and context information, assisting the user to quickly form a sentence. Similarly, we assign a value between 0 to 1 to estimate the accuracy. This function may appear indistinguishable from the current word prediction function in the interface as it also shows a list of predicted words on the system, however, the underlying algorithms are different. Thus we separate word prediction accuracy into two parameters.

Sentence Prediction Accuracy — P_{pred_s} Information retrieval-based sentence generation and large language models-based (LLMs-based) sentence generation are two mainstream sentence prediction approaches that have different attributes. The former retrieves text from a limited data set and produces sentence suggestions, while the latter, such as ChatGPT [34] and GPT-4 [35], produces sentence predictions based on prompts. However, the acceptance of predicted words and generated sentences under the scope of LLMs remains an unanswered question, as individuals may have different levels of acceptance of AI-suggested text, which leads to different P_{pred_s} values. In addition, the *imperfect surrogate user* focuses on simulating and identifying non-expert users' performance affected by human factors concerns. Therefore, the mathematical simulation of specific word and sentence prediction functions is out of the scope of this research. Accordingly, to simplify this model, we assign a value between 0 tand 1 to estimate the accuracy, which we derive empirically from an existing AAC text entry system [56]. We discuss this procedure in Section 3.2.

3.1 Text Entry Strategy Modeling

There are three parallel interaction points in interacting with a text prediction system: (1) letter-by-letter typing; (2) selecting a word prediction; and (3) selecting a sentence prediction. The main goal of studying text entry strategy is to minimize the cost of a poor guess (checking the prediction list but end up without a satisficing result) and maximize entry rate (quickly finish the sentence by saving keystrokes). The strategy is defined by whether, and when, referring to the word predictions and the sentence predictions. In other words, the strategy determines how users arrange their physical and mental actions.

Operator	Description and Remarks	Time (sec) or Rate (%)
T_{key}	Keystroke time	
	Spinal cord injury users with good typing skills without prediction systems	0.60 sec^a
	Disabled user on AZERTY keyboard	2.08 sec^b
T_{react}	Reaction time for word prediction	
	Spinal cord injury users unfamiliar with word prediction systems	1.20 sec^a
R_{error}	Human error rate	
	Disabled user on AZERTY keyboard	10.28 $\%^b$

Table 1. Available AAC user parameter reference values from the literature [a][18], [b][52].

Kristensson and Müllners [23] demonstrate that, in word predictive text entry systems, such strategies have a significant impact on text entry rate and keystroke savings. They propose three text entry strategy parameters for word prediction:

***Minimum Word Length* — L_{min_w}** The minimum word length strategy restricts the use of predictions to only words above a certain length, L_{min_w}. The idea behind this parameter is to only refer to predictions for longer words to save keystrokes.

***Type-then-Look for Word Predictions* — k_{look_w}** The prediction success rate increases when typing a new letter in a word. This parameter defines the number of letters that need to be typed before looking at the word prediction list. Holding off the use of the word prediction function in the initial letters' entry increases the reliability of predictions and reduces the time for checking the prediction list.

***Perseverance for Word Predictions* — p_{max_w}** The system is unlikely to produce accurate predictions for every word, so the user is unlikely to pursue word predictions indefinitely. This strategy parameter assumes the user checks the prediction every time a new letter is typed. If the correct prediction is not obtained by the nth letter, the prediction is abandoned. This parameter defines the number of letters that are typed before stopping to check the word prediction list.

Similarly, for the sentence predictive function, we define three new corresponding parameters:

***Minimum Sentence Length* — L_{min_s}** A correct prediction for a long sentence can produce large keystroke savings. This parameter limits the use of predictions to only sentences above a certain length, L_{min_s}, in words.

***Type-then-Look for Sentence Predictions* — k_{look_s}** Consistently typing words increases the reliability of sentence prediction. This parameter defines the number of words that need to be typed before looking at sentence predictions.

***Perseverance for Sentence Predictions* — p_{max_s}** Checking long sentence predictions often takes a longer time than checking short word predictions. Hence, users may discard sentence predictions when a certain number of words has been typed. This parameter defines this cut-off strategy.

Figure 2 illustrates the overall sentence entry strategy on the NLG text entry system equipped with word and sentence prediction functions. Repeated steps are summarized and modularized for simplicity and clarity, and are presented as yellow rounded squares in the graph. The details of these modules are shown in Entering the Current Word Model (see Figure 3), and Entering the Next Word Model (see Figure 4) respectively.

3.2 Parameter Allocation for Surrogate Users and Predictive AAC Systems

Envelope analysis essentially simulates the user and system interaction and calculates the performance within an envelope of parameterized conditions. Therefore, the choice of these parameters can affect the results of estimations. In this study, the diversity of system prediction accuracy based on various language models is out of the scope of this paper. Instead, we aim to investigate the impact of text entry strategy on system efficiency by using fixed parameters for both surrogate users and conceptual AAC systems. This approach strikes a balance between over-parameterization and simplicity and is based on an "uninformative prior" to avoid the need for elaborate distributional assumptions that may be challenging to justify.

To regulate the prediction accuracy of the system (i.e., $P_{pred\_w\_c}$, $P_{pred\_w\_n}$, P_{pred_s}), we utilize an existing AAC system that integrates word and sentence prediction functions [56], along with a publicly available fictional AAC-like communications dataset [49]. From this corpus, we randomly select 100 sentences, which we use for the following envelope analyses and real user text entry analyses. These sampled sentences vary in length from one to ten words, with an average length of 5.13 words. To calculate the prediction accuracy utilizing equation 3, we type each sample sentence on the AAC system and log predicted words and sentences, resulting in $P_{pred\_w\_c} = 0.71$, $P_{pred\_w\_n} = 0.58$, $P_{pred\_s} = 0.44$. We illustrate the impact of each model on net entry rates and changes in keystrokes by envelope analyses for each condition. The text entry rate is measured in words per minute (wpm) and uses the convention that one word is five characters long, including spaces.

We create an example AAC surrogate user ($T_{key} = 0.60sec$, $T_{react\_w} = 1.20sec$, $T_{react\_s} = 1.20 \times 5.13 = 6.16sec$, where 5.13 is the average sentence length in the dataset) using available AAC user parameter values from the literature, listed in Table 1.

As a comparison, we create another able-bodied surrogate user by adopting the parameter from the previous study [23]: $T_{key} = 0.26sec$ (based on [3]), $T_{react\_w} = 0.45sec$ (based on [10]), $T_{react\_s} = 0.45 \times 5.13 = 2.31sec$, where 5.13 is the average sentence length in the dataset.

3.3 Envelope Analyses and Surrogate Users via KLM

We explore the viable efficacy, in terms of communication rate, of a wide range of text entry strategies on this NLG system through quantitative envelope analyses. The fundamental mechanism of this analysis is simulating the user performance on the computational system model and calculating the time cost and keystrokes of a given task via the keystroke level model (KLM) [7], which includes three operators: physical, cognitive and system:

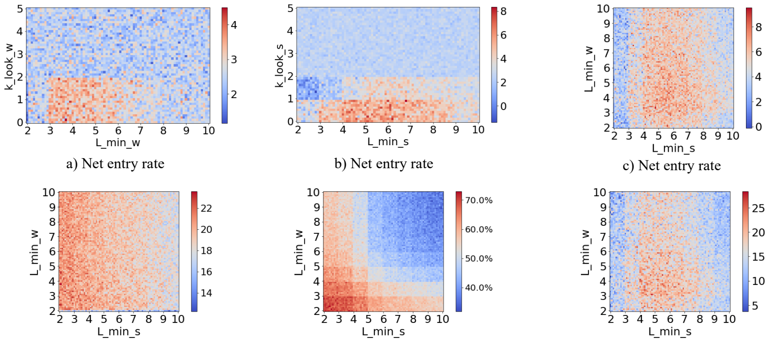
$$T_{execute} = T_{physical} + T_{cognitive} + T_{system} \tag{2}$$

We ignore T_{system} in this analysis since modern predictive text entry systems have a nearly instantaneous responses when producing predictions. Then, according to the text entry strategy flowcharts (see Figure 2, 3, and 4), we estimate the time cost of a task, T_{total}, as follows:

$$T_{total} = \sum T_{key} + \sum T_{react\_w} + \sum T_{react\_s} \tag{3}$$

which is influenced by the entry strategy parameters L_{min_w}, k_{look_w}, p_{max_w}, L_{min_s}, k_{look_s}, and p_{max_s}.

Kristensson and Müllners [23] reveal that text entry strategies on a word predictive system significantly impact entry rate. They also indicate that keystroke savings do not guarantee savings



a) Net entry rate

b) Net entry rate

c) Net entry rate

d) Standard deviation of net entry rate

e) Keystroke saving

f) Net entry rate: able-bodied

Image	**a)**	**b)**	**c)**	**d)**	**e)**	**f)**
Prediction Method	word prediction	sentence prediction	word and sentence predictions			
System Parameters	$P_{pred\_w\_c} = 0.71$, $P_{pred\_w\_n} = 0.58$, and $P_{pred\_s} = 0.44$					
User Parameters	**AAC surrogate user:** $T_{key} = 0.60$ sec, $T_{react\_w} = 1.20$ sec, and $T_{react\_s} = 6.16$ *sec*					**Able-bodied surrogate user:** $T_{key} = 0.26$ sec, $T_{react\_w} = 0.45$ sec, and $T_{react\_s} = 2.31$ sec
Strategy Parameters	**Value**					
L_{min_w}	2-10	-	2-10			
k_{look_w}	0-5	-	0			
p_{max_w}	2	-	2			
L_{min_s}	-	2-10	2-10			
k_{look_s}	-	0-5	0			
p_{max_s}	-	2	2			

Fig. 5. System evaluation with different prediction approaches. In Figure 5a–e, we use an AAC users' parameters, while in Figure 5f, we use an able-bodied users' parameters for comparison. In Figure 5c–f (word and sentence predictions as prediction method), the values of the fixed text entry strategy parameters are selected by envelope analyses by comparing each pair of parameters with the net entry rate. $k_{look\_w} = 0$, $p_{max\_w} = 2$, $k_{look\_s} = 0$, and $p_{max\_s} = 2$ yield the maximum net entry rate. We use the same approach for selecting the fixed values for Figure 5a and 5b. Figure 5a shows that when using only the word prediction method, checking predictions for words with two to six letters in the first two letters' inputs leads to a higher net entry rate. Figure 5b suggests that, when using only the sentence prediction method, checking sentence predictions in the first two words' inputs for sentences with four to seven words is optimal. Figure 5c shows that for AAC users, the optimal strategy is to check word predictions for words with lengths of three to six and sentences with lengths of four to seven, yielding net entry rates in the range of 8 to 9 wpm. Figure 5d shows that frequently checking the sentence prediction reduces the stability of the results observed in Figure 5c. Figure 5e reveals that increasing reliance on word and sentence predictions leads to higher keystroke savings. Finally, Figure 5f shows that for able-bodied users, the optimal strategy is to check predictions for words with lengths between three and five and sentences with word lengths between four and six, yielding net entry rates in the range of 25 to 28 wpm.

in time. By carefully selecting the range of strategy parameters, a positive net entry rate can be achieved, such that the net entry rate is the entry rate difference between assisted text entry and straightforward letter-by-letter typing. A positive net entry rate indicates that the predictive system improves typing performance. Accordingly, on the basis of the prior study [23], the motivation behind this analysis is to identify *which* strategies for an AAC text entry system equipped with word and sentence prediction functions can possibly improve text entry rates and keystroke savings of letter-by-letter typing, and to understand whether the sentence prediction function has a positive impact on entry rate.

We examine three conditions: (i) use word predictions only; (ii) use sentence predictions only; and (iii) use mixed predictions.

Simulations of the full parameter set are conducted for different combinations of the text entry strategy parameters: L_{min_w} ranging from 2 to 10, k_{look_w} from 0 to 5, p_{max_w} from 2 to 10, L_{min_s} from 2 to 10, k_{look_s} from 0 to 10, and p_{max_s} from 2 to 10. This combination covers a wide range of possible text entry strategies. The envelope analyses reveal the following discoveries, which are new findings in relation to prior work [23]:

Extensively using word predictions after typing a few letters increases text entry rate. In condition (i), when only word prediction is involved, the system is considered equivalent to a word predictive system. It reproduces a similar result to a previous study in using the able-bodied surrogate user [23], such that word perseverance p_{max_w} has a limited influence on entry rate when type-then-look $k_{look\_w} < 2$. This is because with 71% current word prediction accuracy and 58% next word prediction accuracy (i.e., $P_{pred\_w\_c} = 0.71$ and $P_{pred\_w\_n} = 0.58$ as listed in section 3.2), statistically, 92% words can be accurately predicted within the first two letters if the word is at the start of a sentence (i.e., $0.71 + (1 - 0.71) \times 0.71 = 0.92$) and 96% of words can be accurately predicted within the first two letters if the word is not in the beginning (i.e., $0.58 + (1 - 0.58) \times 0.71 + (1 - 0.58) \times (1 - 0.71) \times 0.71 = 0.96$). However, when $k_{look\_w} > 3$, $p_{max\_w} < 4$ yields a higher net entry rate, which is not observed when using the able-bodied surrogate user. This is because the marginal benefit from selecting expected predictions decreases when the word approaches completion (after four letters are typed in this case). Regular checking of predictions consumes more time for AAC users (i.e., a larger T_{react}), resulting in a faster decline compared to able-bodied users. Hence, we fix $p_{max\_w} = 2$ and alter minimum word length L_{min_w} and type-then-look for word predictions k_{look_w} to observe their effects on net entry rate. As shown in Figure 5a, the entry rate strongly depends on the choice of these two parameters. The red hot colors with net entry rates above average indicate when $k_{look\_w} < 2$ and L_{min_w} is 3–5, the net entry rate reaches its maximum.

Sentence prediction strategy greatly impacts text entry rate. In condition (ii), the analysis of sentence prediction shows that sentence perseverance p_{max_s} has little influence on the entry rate when type-then-look $k_{look\_s} < 3$, as the 82% of sentences are correctly predicted within the first three words (i.e., $0.44 + (1 - 0.44) \times 0.44 + (1 - 0.44)^2 \times 0.44 = 0.82$). However, when type-then-look $k_{look\_s} > 3$, sentence perseverance $p_{max\_s} < 4$ yields a higher net entry rate. This is not observed with the able-bodied surrogate user either. We conjecture this is for the same reason as for word prediction. Hence, we set $p_{max\_s} = 2$ to investigate the impact of L_{min_s} and k_{look_s} on net entry rate. Figure 5b shows that when only the sentence prediction function is available, when L_{min_s} is 4–6 and $k_{look\_s} < 1$, this yields a high positive net entry rate.

Word prediction and sentence prediction together improve text entry rate. The AAC system allows users to adopt both word predictions and sentence predictions in tandem. To understand whether the combined use of both prediction functions can still positively impact the entry rate, in condition (iii), we alter the usage frequency of both the prediction functions via changing L_{min_w} and L_{min_s}. Figure 5c shows that the combination of these two functions yields a higher value range of net entry rate (from -0.01 to 9.6) than these two individual functions' results shown in Figure 5a and Figure 5b (word prediction results from 0.02 to 4.5 wpm and sentence prediction from -1.4 to 8.6 wpm). In addition, frequently using the word and sentence prediction functions after typing the initial few letters/words produces high net entry rates.

Extensively using predictions decreases performance consistency. Figure 5d shows the net entry rate standard deviation. The red hot color indicates a high standard deviation, and the cool blue color represents a low standard deviation. A low value is preferred as it indicates a more consistent level of performance. Extensive use of sentence prediction (i.e., small L_{min_s}) yields a higher standard deviation.

Keystroke savings do not necessarily translate into positive net entry rates. Figure 5e shows that text entry strategies that make extensive use of predictions (i.e., small L_{min_w} and L_{min_s}) maximize the keystroke savings. However, this strategy yields a low net entry rate, as shown in Figure 5c. In contrast, the strategies that yield medium keystroke savings have a higher net entry rate (compare Figure 5c and 5e), which indicates that the keystroke savings metric is not linearly correlated to net entry rate.

It is meaningful to design for individual users. By comparing Figure 5c and 5f, we observe that by adopting the same text entry strategy, the AAC surrogate user yields a much lower net text entry rate than the able-bodied surrogate user. This emphasizes that individual differences can lead to very different performance outcomes. We also find that the optimal strategies that yield the best net entry rate are actually different due to the difference between the AAC surrogate user and the able-bodied surrogate user in terms of typing speed and reaction time.

4 IMPERFECT SURROGATE USER MODEL KLM-BEI

This section introduces the *imperfect surrogate user model* KLM-BEI: a keystroke-level model augmented by modeling bounded rationality, human errors, and interruptions. It is an extension of the conventional task analysis model KLM. While conventional KLM is useful as a "cheap and cheerful" initial estimation model, it is nonetheless a very simple model that can only approximately predict the time cost of a task decomposed by unit tasks in a perfect context with no interruptions of the task, using a single method, and assuming error-free expert performance, etc. [41].

However, actual AAC users are not always capable of achieving a goal in an optimal way as, inevitably, errors and interruptions occur which affect performance. To account for this, we introduce the *imperfect surrogate user model* KLM-BEI to address these inherent uncertainties presented in the task analysis. The model regards the user, the interactive system, and the environment as a joint system that involves decision-making in the presence of action execution failures and interruptions in the environment.

An overview of the model is illustrated in Figure 6. The system interaction simulator includes three user action stages in turn: the decision-making action stage that correlates to the text entry strategy (i.e. to select a word prediction or a sentence prediction or to type letter-by-letter) where the system checks bounded rationality (i.e. to select the correct prediction or ignore), the keystroke action stage where the system checks human error (i.e. to type the key correctly or not), and actual keystroke action execution where the system calculates the time cost of the action using a model reminiscent of KLM. Further, interruptions are monitored throughout the whole interaction process and the time cost is also calculated by the model. It is worth mentioning that, in Figure 2, 3, and 4, the teal rectangles with dashed lines are user actions that require human performance factor checks.

We now describe the rest of the key components of this model: (1) the bounded rationality model; (2) the human error model; and (3) the interruption model. To ensure consistency, we illustrate

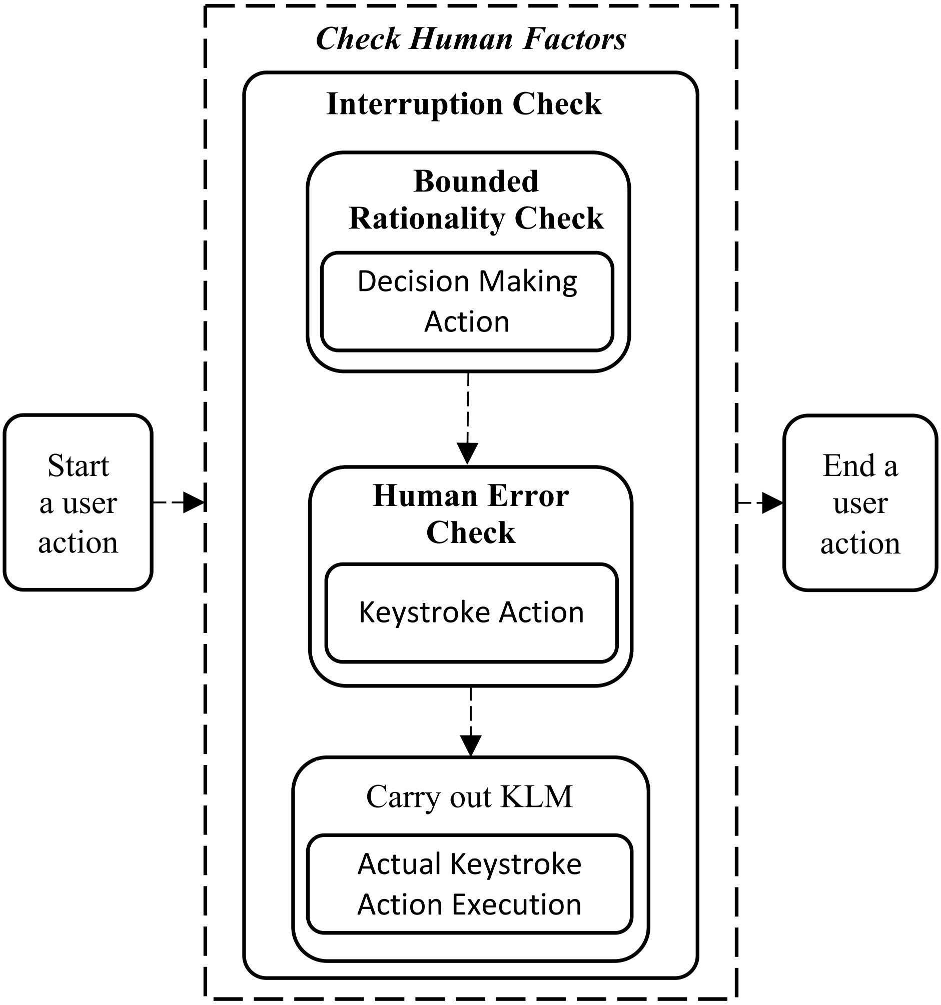


Fig. 6. An overview of the *imperfect surrogate user model* KLM-BEI. This model incorporates uncertainties into the KLM model, including bounded rationality, human error, and interruption. These specific models will be introduced in Figure 7, 9, and 10, respectively. The three modules checking human performance are highlighted by dashed-line boxes, which are aligned with user actions represented as teal rectangles with dashed lines in Figures 2, 3, and 4. The correlated flowchart indicating simulation steps shows in Figure 13.

the impact of each model on net entry rates and changes in keystrokes by envelope analyses for each condition and adopt the same selected sentences for simulation and the same system settings introduced in Section 3.2.

4.1 Bounded Rationality Model

Models of goal-directed planning that take the expenses of computation into account are described as models incorporating *bounded rationality*, a term coined by Herbert Simon [47] in behavioral economics. People frequently engage in satisficing strategies where they follow a plan that is

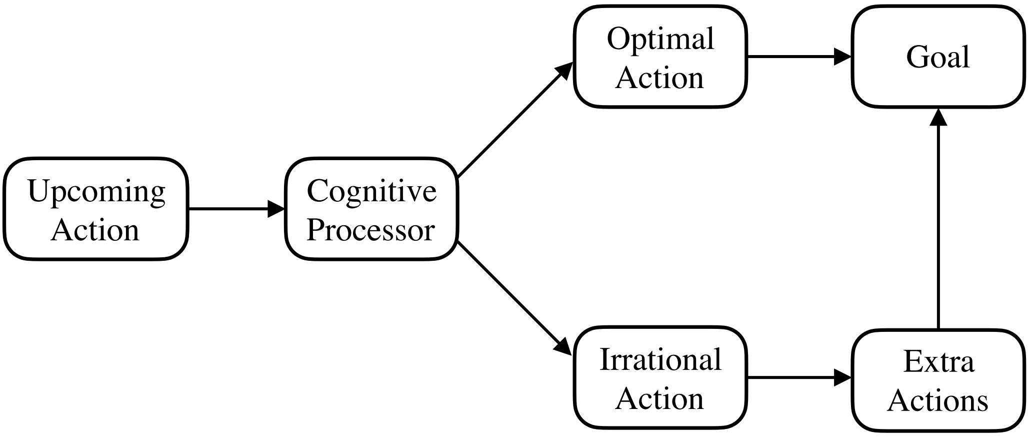


Fig. 7. An overview of the bounded rationality model.

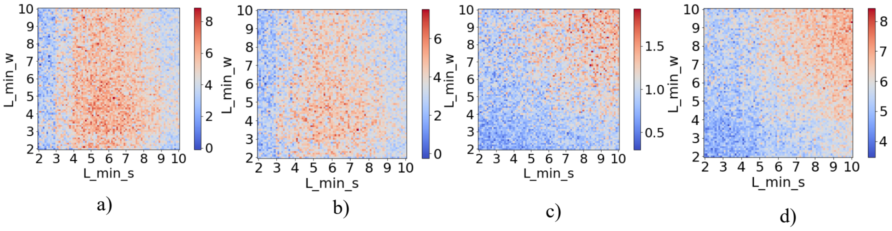
satisfactory, rather than optimal, within some constraints. We introduce a bounded rationality model for envelope analysis to illustrate human decisions in a simple interaction task, such as entering texts in a predictive system.

As shown in Figure 7, the simple goal-oriented task starts with an upcoming action. Then the cognitive processor decides whether the surrogate user executes either an optimal (i.e. rational) or an irrational action. This process is determined by the parameter *rational rate* ($R_{rational}$):

$$R_{rational} = 1 - \frac{N_{irrational}}{N_{errorfree}} \tag{4}$$

where $N_{irrational}$ is the number of *irrational actions* and $N_{errorfree}$ is *all actions* under an error-free condition. A higher $R_{rational}$ represents a higher rationality level of actions. An optimal action brings the user to the goal directly, while an irrational action requires extra actions. For example, assume the goal is to type the word 'beautiful' in the predictive text entry system and the user has typed the letter 'b', and accordingly, the system presents the word prediction suggestion 'beautiful'. A *rational action* is to choose the word prediction suggestion to complete this entry, whereas an *irrational action* is to type the next letter 'e', which still pursues the goal but demands several extra keystrokes to complete the word. The *extra actions* are those actions followed by an irrational action until the goal is achieved, regardless of whether the user chooses to engage with a word prediction suggestion or to type letter-by-letter in subsequent keystrokes.

The comparison among Figure 8a–d shows that different bounded rationality levels can impact the text entry rate due to the extra keystrokes. A higher rationality level (i.e., a higher $R_{rational}$ value) leads to more efficient usage of the system utility and fewer extra keystrokes. Specifically, Figure 5c ($R_{rational}$ = 100%), Figure 8a ($R_{rational}$ = 90%), and Figure 8b ($R_{rational}$ = 50%) show three different distributions of net entry rate with respect to word and sentence entry strategies. In general, surrogate users with lower rationality levels tend to produce lower text entry efficiencies. In addition, the strategy parameter configurations that produce the fastest text entry rate can



a) b) c) d)

Image	a)	b)	c)	d)
Description	Net entry rate with 90% rational operations	Net entry rate with 50% rational operations	Extra keystrokes with 90% rational operations	Extra keystrokes with 50% rational operations
System Parameters	$P_{pred\_w\_c} = 0.71$, $P_{pred\_w\_n} = 0.58$, and $P_{pred\_s} = 0.44$			
User Parameters	**AAC surrogate user:** $T_{key} = 0.60$ sec, $T_{react\_w} = 1.20$ sec, and $T_{react\_s} = 6.61$ sec			
Human Performance Factors	**Value**			
$R_{rational}$	90%	50%	90%	50%
Strategy Parameters	**Value**			
L_{min_w}	2-10	2-10	2-10	2-10
k_{look_w}	0	0	0	0
p_{max_w}	2	2	2	2
L_{min_s}	2-10	2-10	2-10	2-10
k_{look_s}	0	0	0	0
p_{max_s}	2	2	2	2

Fig. 8. The impact of bounded rationality and text entry strategies on typing efficiency in terms of net text entry rate and extra keystrokes. Figure 8a shows that when 90% of the actions are rational, the optimal strategy is to check predictions for words with lengths of three to five and sentences with lengths of four to six, resulting in a net entry rate range between 7 and 9 wpm. Figure 8b shows that when 50% of the actions are rational, the optimal strategy is to check predictions for words with lengths of three to four and sentences with word lengths of four to seven, yielding a net entry rate range between 4 to 7 wpm. Figure 8c shows that when 90% of the actions are rational, increasing the frequency of word and sentence prediction usage leads to higher extra keystrokes, with a maximum of two extra keystrokes. Finally, Figure 8d shows a similar trend but with a higher maximum number of eight extra keystrokes.

change under different rationality conditions. In other words, a user's optimal strategy may change depending on the user's level of rationality.

4.2 Human Error Model

Erroneous behavior is an inherent part of human performance [41]. Although there are many different ways to categorize error types such as slips, knowledge-based mistakes, rule-based mistakes, etc. [16, 32, 39], these errors share the same attributes: task execution deviates from the goal. In this study, we assume errors can be spotted during the sentence entry task, and a set of correcting actions are executed once the error is observed. As illustrated in Figure 9, the task starts with an upcoming action. The motor processor decides whether the surrogate user executes either an expected or unexpected action. An expected action results in accomplishing the goal, while

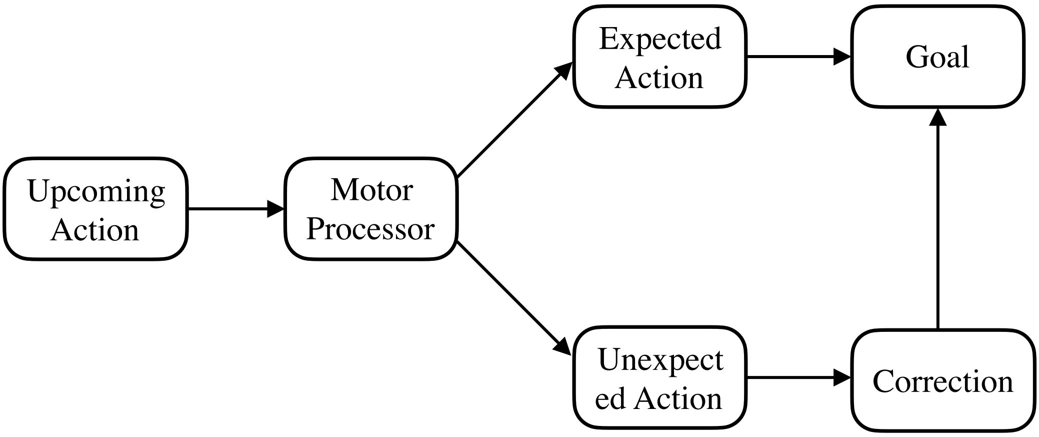


Fig. 9. An overview of the human error model.

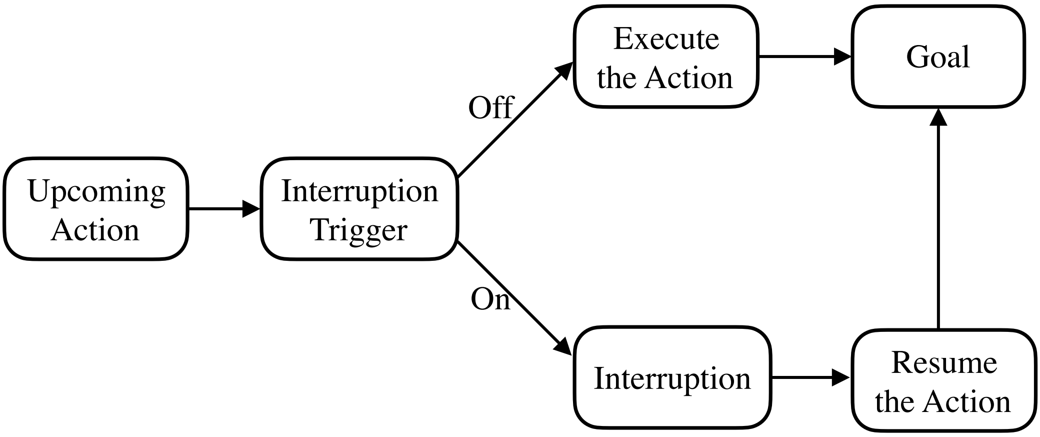
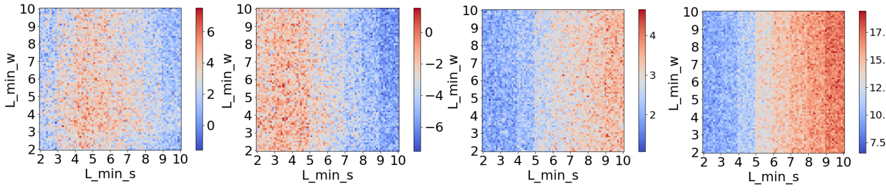


Fig. 10. An overview of the interruption model.

an unexpected action requires a set of corrective steps, which requires additional actions. In a predictive text entry system, unexpected actions include typing unexpected letters or selecting an unwanted word or sentence suggestion.



a) b) c) d)

Image	a)	b)	c)	d)
Description	Net entry rate with 10% erroneous operations	Net entry rate with 50% erroneous operations	Corrective keystrokes with 10% erroneous operations	Corrective keystrokes with 50% erroneous operations
System Parameters	$P_{pred\_w\_c} = 0.71$, $P_{pred\_w\_n} = 0.58$, and $P_{pred\_s} = 0.44$			
User Parameters	**AAC surrogate user:** $T_{key} = 0.60$ sec, $T_{react\_w} = 1.20$ sec, and $T_{react\_s} = 6.61$ sec			
Human Performance Factors	**Value**			
R_{error}	10%	50%	10%	50%
Strategy Parameters	**Value**			
L_{min_w}	2-10	2-10	2-10	2-10
k_{look_w}	0	0	0	0
p_{max_w}	2	2	2	2
L_{min_s}	2-10	2-10	2-10	2-10
k_{look_s}	0	0	0	0
p_{max_s}	2	2	2	2

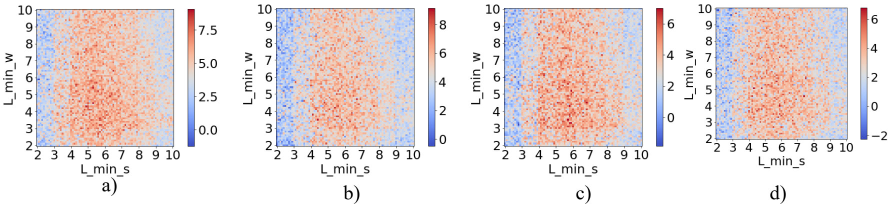
Fig. 11. The impact of human error and text entry strategies on typing efficiency (the net text entry rate and the corrective keystrokes). Figure 10a shows that when 10% of the actions are erroneous, the optimal strategy is to check predictions for words with lengths of three to four and sentences with lengths of four to five, resulting in a net entry rate range between 5 and 7 wpm. Figure 10b shows that when 50% of the actions are erroneous, the optimal strategy is to check predictions for sentences with word lengths less than five, yielding a net entry rate between -1 to 1 wpm, while word prediction has limited impact in this case. Figure 10c shows that when 10% of the actions are erroneous, increasing the frequency of sentence prediction usage leads to more corrective keystrokes, with a maximum of five corrective keystrokes. Finally, Figure 10d shows a similar trend but with a higher maximum number of 19 corrective keystrokes.

We define the *human error rate* (R_{error}) to interpret the possibility of an action being erroneous:

$$R_{error} = \frac{N_{unexpected}}{N_{total}} \tag{5}$$

where $N_{unexpected}$ is the keystroke number of *unexpected actions* and N_{total} is the overall keystrokes. A lower R_{error} represents higher user expertise in using the system.

We investigate the impact of human error on typing efficiency via two sets of parameters with different R_{error}. Comparing Figure 5c ($R_{error} = 0$) and Figure 11a ($R_{error} = 10\%$), it is obvious that human error greatly constrains the net entry rate, even with only 10% erroneous operations. The comparison between Figure 11a ($R_{error} = 10\%$) and Figure 11b ($R_{error} = 50\%$) shows that a high error rate can not only lead to a very different optimal typing strategy (i.e., the hot red area indicating the relatively high entry rate changes the distribution) but also dramatically decrease



a) b) c) d)

Image	a)	b)	c)	d)
Description	Net entry rate with 10% interruption rate and a 5-second interruption event	Net entry rate with 50% interruption rate and a 5-second interruption event	Net entry rate with 10% interruption rate and a 50-second interruption event	Net entry rate with 50% interruption rate and a 50-second interruption event
System Parameters	$P_{pred\_w\_c} = 0.71$, $P_{pred\_w\_n} = 0.58$, and $P_{pred\_s} = 0.44$			
User Parameters	**AAC surrogate user:** $T_{key} = 0.60$ sec, $T_{react\_w} = 1.20$ sec, and $T_{react\_s} = 6.61$ sec			
Human Performance Factors	**Value**			
T_{event}	5 seconds		50 seconds	
$R_{interrupt}$	10%	50%	10%	50%
Strategy Parameters	**Value**			
L_{min_w}	2-10	2-10	2-10	2-10
k_{look_w}	0	0	0	0
p_{max_w}	2	2	2	2
L_{min_s}	2-10	2-10	2-10	2-10
k_{look_s}	0	0	0	0
p_{max_s}	2	2	2	2

Fig. 12. The impact of interruption and text entry strategies on the net text entry rate. Figure 12a shows that with a 10% interruption rate and a five-second interruption event, the optimal strategy is to check predictions for words with lengths of three to seven and sentences with lengths of four to six, resulting in a net entry rate range between 6 and 8 wpm. Figure 12b shows that when we have a 50% interruption rate and a five-second interruption event, the optimal strategy is to check predictions for words with lengths of three to six and sentences with word lengths of four to six, yielding a net entry rate range between 5 and 8 wpm. Figure 12c shows that with a 10% interruption rate and a 50-second interruption event, the optimal strategy is to check predictions for words with lengths of three to seven and sentences with word lengths of four to six, resulting in a net entry rate range between 5 and 7 wpm. Figure 12d shows that when we have a 50% interruption rate and a 50-second interruption event, the optimal strategy is to check predictions for words with lengths of three to six and sentences with word lengths of four to six, yielding a net entry rate range between 4 and 6 wpm.

the text entry rate as the maximum net entry rate in Figure 11b is 1.7 wpm, less than 7.7 wpm in Figure 11a. Figure 11c–d reveals that the reason behind this phenomenon is that the high human error rate leads to more corrective keystrokes in general. In addition, the highly frequent use of sentence prediction functions (i.e., a smaller L_{min_s} value) minimizes the corrective keystrokes, while the use of word prediction functions has a limited impact on the corrective keystrokes.

4.3 Interruption Model

Interruption is one of the important human performance factors that forces users to pause their tasks. We propose a simple model allowing systems to auto-detect interruption events. Figure 10 illustrates this concept, starting with an upcoming action. Thereafter an interruption event that suspends this action may be triggered, determined by the *interruption trigger*. If the interruption trigger is off then there are no interruptions, and the user executes the action normally. If the interruption trigger is on then the user requires extra time to resume the action after they react to the interrupting event. This resumption time ($T_{interrupt}$) is considered the cost of the interruption. In addition, we assume that the interruption happens at most once in one sentence entering process for simplicity. *Interruption rate* ($R_{interrupt}$), ranging from 0 to 100%, is used to determine the possibility of activating the trigger, which is defined by the equation:

$$R_{interrupt} = \begin{cases} \frac{N_{interrupt}}{N_{sentence}} & \text{If no interruption has happened} \\ & \text{during current sentence entry} \\ & \text{If there has been an interruption} \\ 0 & \text{during the current sentence entry} \end{cases} \tag{6}$$

where $N_{interrupt}$ is the number of interruptions in all sentences and $N_{sentence}$ is the number of sentences.

Hodgetts and Jones. [15] indicate that a longer interruption requires a higher cost for retrieving goal memory, leading to a slower resumption. The motivation to investigate the interruption is that a well-designed user interface is supposed to assist this memory retrieval process and reduce the cognitive load. In other words, the task resumption cost can be regarded as one of the indicators for evaluating an interaction design. Monk et al. [31] find a logarithmic model that fits the correlation of the interrupting event time and the resumption time when the interruption duration is within 60 seconds:

$$T_{interrupt} = 0.189 \times \log(T_{event}) + 1.03 \tag{7}$$

where T_{event} is less than 60 seconds, and the constants may vary to fit different settings better.

As shown in Figure 12, we set four conditions with different interruption frequencies and interruption durations to understand their impact on the net text entry rate.

The distributions of net entry rates in the four sub-figures share a similar pattern to the result from an ideal setting shown in Figure 5c ($R_{interrupt} = 0$). High net entry rates gather at a strategy range of relatively high frequent use of the word and sentence predictions ($L_{min\_w} > 2$ and L_{min_s} is 4–8). However, the range of the net entry rate differs in different settings. The comparisons between Figure 12a, 12b and Figure 12c, 12d show that a higher interruption rate (higher $R_{interrupt}$) leads to a lower net entry rate. In addition, a longer interruption event tends to have a more severe impact on entry rate when the interruption rate is high.

4.4 The KLM-BEI Model

The KLM-BEI model is a wrapped-up model of the three fore-mentioned human performance factors: bounded rationality, human error, and interruption. Based on Figure 6, 7, 9, and 10, Figure 13 illustrates the workflow of this combined model that continuously checks the three aspects during the whole interaction process. To understand the resistance of the system with a fixed text entry strategy (i.e., to what extent the system can keep improving text entry rate when taking human performance factors into account), we allocate a set of optimal text entry strategy parameters and alter the rational rate ($R_{rational}$) and the error rate (R_{error}). However, note that we fix the interruption parameters ($R_{interupt}$ = 10% and T_{event} = $5sec$) and optimal text entry strategy parameters ($L_{min\_w} = 4$, $k_{look\_w} = 0$, $p_{max\_w} = 2$, $L_{min\_s} = 5$, $k_{look\_s} = 0$ and $p_{max\_s} = 2$).

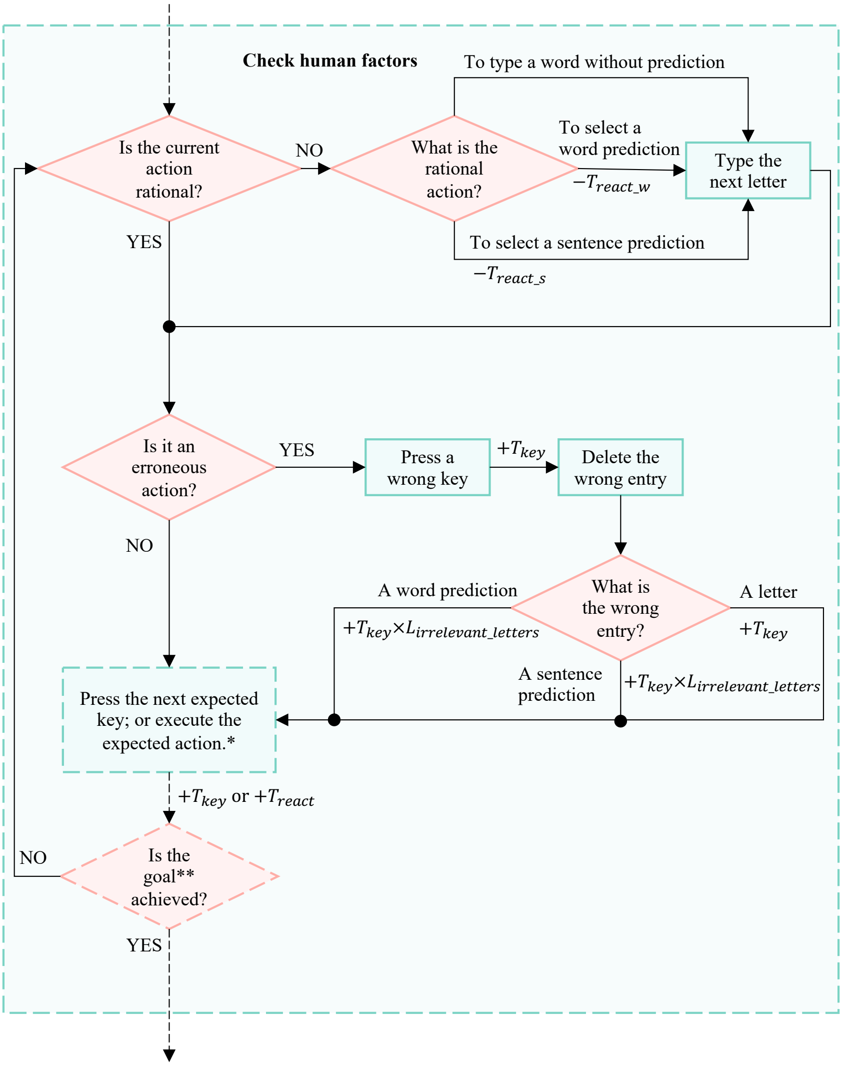

Check human factors
Is the current action rational?
NO
YES
What is the rational action?
To type a word without prediction
To select a word prediction
$-T_{react\_w}$
To select a sentence prediction
$-T_{react\_s}$
Type the next letter
Is it an erroneous action?
YES
NO
Press a wrong key
$+T_{key}$
Delete the wrong entry
What is the wrong entry?
A word prediction
$+T_{key}\times L_{irrelevant\_letters}$
A letter
$+T_{key}$
A sentence prediction
$+T_{key}\times L_{irrelevant\_letters}$
Press the next expected key; or execute the expected action.*
$+T_{key}$ or $+T_{react}$
Is the goal** achieved?
NO
YES

Fig. 13. The flowchart of Check Human Performance Factors Model. (*): This could be a corrective action, which means the actual executed action is based on the type of correction the user is carrying out. (**): The goal refers to the goal in the bounded rationality model (Figure 7) and human error model (Figure 9).

Figure 14a shows that, as expected, the highest net entry rate is achieved with a maximum rational rate ($R_{rational}$ = 100%) and a minimum human error rate (R_{error} = 0). The steep zero-crossing line in this figure indicates that R_{error} exerts more influence on the net entry rate than $R_{rational}$, as when R_{error} > 60%, the net entry rate cannot possibly have a positive value. On the one hand, this implies that regardless of the efficiency of the prediction function, once the human error rate remains at a high level, it is very difficult to increase the entry rate through the text prediction algorithms. Instead, a better design direction would be to improve the user experience to reduce the human error rate. On the other hand, it also reveals that bounded rationality has a relatively lower impact on the entry rate than human error. Within the full range of $R_{rational}$, the net entry rate

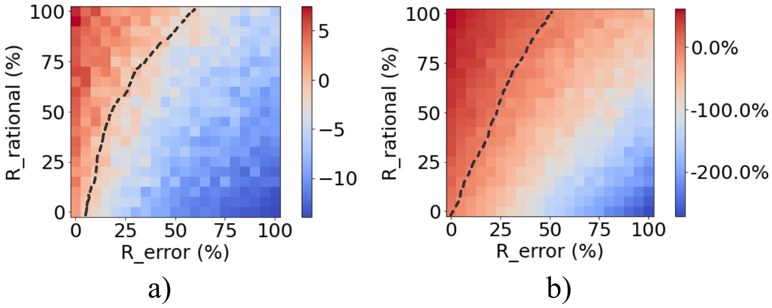
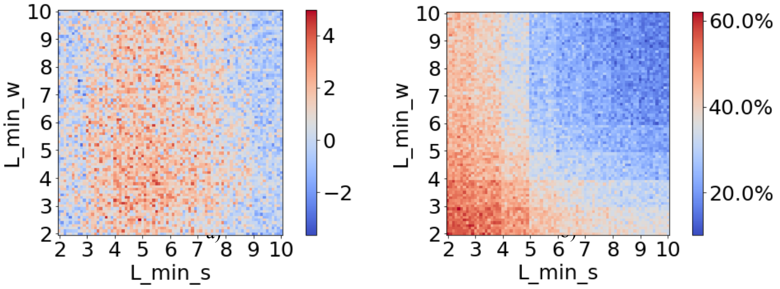


Image	a)	b)
Description	Net entry rate of optimal text entry strategy with respect to the full range of $R_{rational}$ and R_{error}	Keystroke saving of optimal text entry strategy with respect to the full range of $R_{rational}$ and R_{error}
System Parameters	$P_{pred\_w\_c} = 0.71$, $P_{pred\_w\_n} = 0.58$, and $P_{pred\_s} = 0.44$	
User Parameters	**AAC surrogate user:** $T_{key} = 0.60$ sec, $T_{react\_w} = 1.20$ sec, and $T_{react\_s} = 6.16\ sec$ and $T_{interrupt} = 0.189 \times \log(T_{event}) + 1.03$ sec	
Strategy Parameters	$L_{min\_w} = 4$, $k_{look\_w} = 0$, $p_{max\_w} = 2$, $L_{min\_s} = 5$, $k_{look\_s} = 0$, $p_{max\_s} = 2$	
Human Performance Factors	**Value**	
T_{event}	5 seconds	5 seconds
$R_{interrupt}$	10%	10%
$R_{rational}$	0-100%	0-100%
R_{error}	0-100%	0-100%

Fig. 14. Human performance factors evaluation with optimal text entry strategy. The black dashed line makes the zero-crossing, above which predictions provide a performance gain.

can achieve a positive value if the human error rate is well managed. Figure 14b observes a similar result that the keystroke savings can only be positive when $R_{error} < 40\%$. Further, with a higher rationality rate, the system has a higher tolerance to human errors in terms of saving keystrokes.

Having first simulated an imperfect surrogate user by fixing the human performance factors we now investigate the impact of using a different text entry strategy on performance. Specifically, we investigate the text entry rate and keystroke savings. The table in Figure 15 shows the parameter

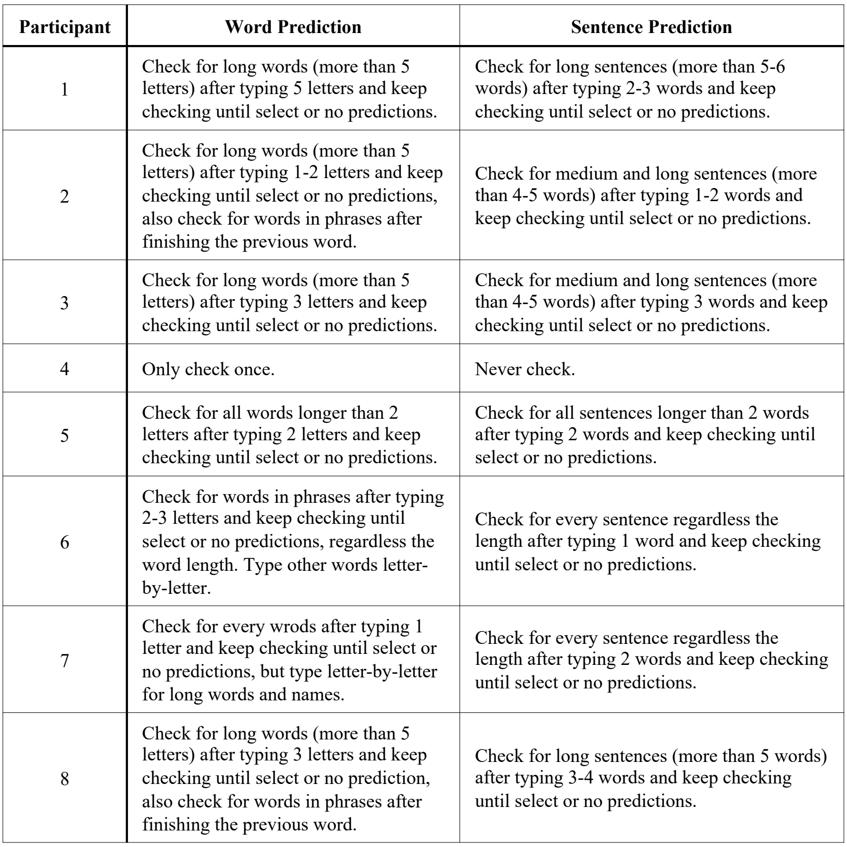


a) b)

Image	a)	b)
Description	Net entry rate of the imperfect surrogate user	Keystroke saving of the imperfect surrogate user
System Parameters	$P_{pred\_w\_c} = 0.71$, $P_{pred\_w\_n} = 0.58$, and $P_{pred\_s} = 0.44$	
User Parameters	**AAC surrogate user:** $T_{key} = 0.60$ sec, $T_{react\_w} = 1.20$ sec, and $T_{react\_s} = 6.61$ sec	
Human Performance Factors	$R_{rational} = 90\%$, $R_{error} = 10\%$, $R_{interrupt} = 50\%$, and $T_{event} = 5\ sec$	
Strategy Parameters	**Value**	
L_{min_w}	2-10	2-10
k_{look_w}	0	0
p_{max_w}	2	2
L_{min_s}	2-10	2-10
k_{look_s}	0	0
p_{max_s}	2	2

Fig. 15. Imperfect user simulation with fixed human performance factors. Figure 15a shows that when considering all the impacts of listed human performance factors, the optimal strategy is to check predictions for words with lengths of three to four and sentences with word lengths of four to five, resulting in a net entry rate range between 3 and 5 wpm. Figure 12b shows that in this setting, relying more on word and sentence predictions leads to higher keystroke savings, with a maximum keystroke saving of 61%.

configurations for the envelope analysis. The plots suggest that human performance factors have a



Participant	Word Prediction	Sentence Prediction
1	Check for long words (more than 5 letters) after typing 5 letters and keep checking until select or no predictions.	Check for long sentences (more than 5-6 words) after typing 2-3 words and keep checking until select or no predictions.
2	Check for long words (more than 5 letters) after typing 1-2 letters and keep checking until select or no predictions, also check for words in phrases after finishing the previous word.	Check for medium and long sentences (more than 4-5 words) after typing 1-2 words and keep checking until select or no predictions.
3	Check for long words (more than 5 letters) after typing 3 letters and keep checking until select or no predictions.	Check for medium and long sentences (more than 4-5 words) after typing 3 words and keep checking until select or no predictions.
4	Only check once.	Never check.
5	Check for all words longer than 2 letters after typing 2 letters and keep checking until select or no predictions.	Check for all sentences longer than 2 words after typing 2 words and keep checking until select or no predictions.
6	Check for words in phrases after typing 2-3 letters and keep checking until select or no predictions, regardless the word length. Type other words letter-by-letter.	Check for every sentence regardless the length after typing 1 word and keep checking until select or no predictions.
7	Check for every wrods after typing 1 letter and keep checking until select or no predictions, but type letter-by-letter for long words and names.	Check for every sentence regardless the length after typing 2 words and keep checking until select or no predictions.
8	Check for long words (more than 5 letters) after typing 3 letters and keep checking until select or no prediction, also check for words in phrases after finishing the previous word.	Check for long sentences (more than 5 words) after typing 3-4 words and keep checking until select or no predictions.

Table 2. Descriptions of the eight participants' text entry strategies. These overall strategies are extracted based on our interviews with them and from inspecting log files. In the text entry tasks the participants tended to exhibit a consistent overall performance, but at a sentence level, they may have adopted flexible strategies to suit their needs.

significant impact on performance. Compared to the perfect user (oracle) simulation (see Figure 5c and 5e), the simulation of the imperfect surrogate user (see Figure 15a and 15b) demonstrates a notably lower maximum net entry rate and keystroke savings and a smaller range of text entry strategy parameters that produce positive net entry rates.

5 RUNTIME ESTIMATION OF PARAMETERS AND VALIDITY

A natural question to ask is how to validate the overall model. However, since the model is generative by definition, it creates all conceivable operating points given a particular parameter configuration [21, 23]. Thus, assuming the parameter interactions are valid, if the parameters are accurate, then the correct corresponding possible operating points will be generated.

Therefore, a more meaningful and practical question is how to *estimate* such parameter values by observing the runtime behavior of the joint human-computer system during use. Providing such a function allows designers to rapidly estimate appropriate parameter values, which can then subsequently be used in envelope analysis.

Participant	1	2	3	4	5	6	7	8
T_{key} (sec)	0.25	0.32	0.27	0.34	0.26	0.27	0.35	0.51
T_{react_w} (sec)	0.75	0.77	0.98	n/a	0.88	0.65	1.00	0.61
T_{react_s} (sec)	2.18	1.18	0.87	n/a	1.61	n/a	1.73	1.21
R_{error} (%)	9.1	2.9	6.0	8.5	17.8	2.0	2.3	5.3
$R_{irrational}$ (%)	86.3	87.8	88.8	86.5	86.8	91.5	89.1	90.3
$R_{interrupt}$ (%)	102.1	107.0	88.2	75.3	126.1	96.9	122.8	98.0
$T_{interrupt}$ (sec)	3.92	3.20	3.84	3.65	4.48	3.80	4.11	4.00
Entry Rate (WPM)	27.14	24.74	23.69	21.49	20.57	12.75	16.80	15.70
Keystroke Saving (%)	13.0	1.2	-8.5	-30.4	-56.7	-103.1	-71.6	-91.5

Table 3. Real user parameters extracted from the eight participants using KLM-BEI.

In contrast to prior work [21, 23], we have developed a runtime parameter estimation function in our system. The user freely types sentences, and by observing the behavior of the joint system, our system automatically estimates parameter values. When the user has typed a complete sentence, the goal of the user is assumed to be to arrive at that sentence. Thus, the system can estimate the parameters on a sentence frequency basis.

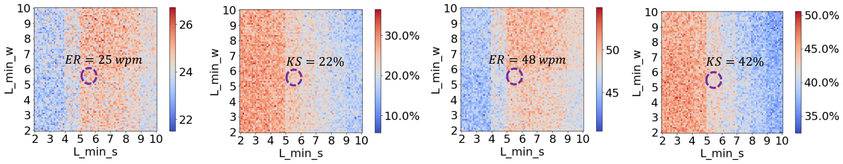
Erroneous actions can be estimated by examining deletion actions. Since they relate to text correction, erroneous actions can be estimated as the actions that the user took to input text that ended up being deleted text.

Irrational actions can be estimated by examining any additional actions taken by the user to type the text that could have been avoided had the user noticed and used any suitable text prediction suggestions that were provided by the system.

We estimate interruption time by assessing whether the user's reaction time is longer than the time we expect the user to need to type the next key. This is achieved by examining the time interval between two subsequent actions. If the interval is longer than the keystroke typing time and the time required to assess text prediction suggestions, then we consider an interruption event to have occurred. We then calculate the interruption time as the time between the start of the user's interruption and the time when they resume their typing task. In envelope analyses, we can estimate the resumption time (i.e., the interruption cost) by using Equation 7, as it, along with the interruption event time, adds up to the total interruption time.

5.1 Validity

We have two goals for validating the KLM-BEI model. First, to validate that the KLM-BEI model can be used to extract parameters that are affected by human performance factors at runtime, such as



a) Entry rate in KLM-BEI

b) Keystroke saving in KLM-BEI

c) Entry rate in KLM

d) Keystroke saving in KLM

Image	a)	b)	c)	d)
System Parameters	$P_{pred\_w\_c} = 0.71$, $P_{pred\_w\_n} = 0.58$, and $P_{pred\_s} = 0.44$			
Real User Parameters	$T_{key} = 0.25$ sec, $T_{react\_w} = 0.75$ sec, and $T_{react\_s} = 2.18$ sec			
Extracted Human Performance Factors	$R_{rational} = 86.3\%$, $R_{error} = 9.1\%$, $R_{interrupt} = 102.0\%$, and $T_{event} = 3.92$ sec			
Extracted Strategy Parameters	$L_{min\_w} = 5$, $k_{look\_w} = 5$, $p_{max\_w} = 6$, $L_{min\_s} = 5$, $k_{look\_s} = 2$, $p_{max\_s} = 6$			
Real User Performance	**Real entry rate:** 27.1 wpm; **Real keystroke saving:** 13.5%			
Varibles for Envelope Analyses	L_{min_w}: 2-10; L_{min_s}: 2-10			

Fig. 16. Entry rate (ER) and keystroke savings (KS) estimations via the *imperfect surrogate user model* KLM-BEI and the conventional model KLM using real human performance factors extracted from Participant 1. Figure 17a shows that when using Participant 1's text entry parameters and human performance factors (i.e., using the KLM-BEI model), the optimal strategy is to check predictions for words with lengths larger than six and sentences with word lengths of five to eight, resulting in a net entry rate range between 25 and 27 wpm. Figure 17b shows that in this setting, relying more on word and sentence predictions leads to higher keystroke savings, with a maximum keystroke saving of 35%. Further, Figure 17c shows that when only considering Participant 1's text entry parameters but ignoring human performance factors (i.e., using the KLM model), the optimal strategy is to check predictions for words with lengths larger than six and sentences with word lengths of five to eight, resulting in a net entry rate range between 50 and 55 wpm. Figure 17d shows that in this setting, relying more on word and sentence predictions leads to higher keystroke savings, with a maximum keystroke saving of 51%. Purple circles reflect the estimated entry rate and keystroke savings in different models with respect to the overall text entry strategy adopted by Participant 1, which shows that KLM-BEI can better reflect reality than KLM.

$R_{rational}$, R_{error}, $R_{interrupt}$, and $T_{interrupt}$. Second, to validate that by adopting the proposed KLM-BEI model, envelope analyses can produce more accurate estimations than a previous approach [23] for individual users with the aid of light touch data collection. To achieve these goals, we recruited eight participants by convenience sampling. The participants were literate able-bodied users aged 20–35 and had at least three years of experience in using text entry systems on touchscreen devices. The reason for not recruiting AAC users is that the experiment aimed to validate that the KLM-BEI model can be used for understanding user performance which assists the system design, rather than directly analyzing user performance for a specific system design. Accordingly, participants were asked to type 100 sentences on an AAC text entry system assisted with word and sentence prediction functions [56] installed on a touchscreen tablet PC (Dell XPS 13 2-in-1 tablet with 13" 3:2 3K (2880x1920) touchscreen).

The text entry performance was logged and analyzed via the KLM-BEI model built into the AAC system. The recording is a text file in which the time of each user action, the pressed key, the predicted words and sentences, and the displayed text was logged for calculating text entry rate and keystroke savings. In addition, rational rate ($R_{rational}$), error rate (R_{error}), interruption rate ($R_{interrupt}$), and interruption time (T_{event}) were also extracted from this log.

The same sentences described in Section 3.2 were reused, which were randomly selected from a crowdsourced AAC-like communications dataset [49]. Ten additional sentences were randomly selected from the dataset and provided to the participants to allow them to familiarize themselves with the word prediction and the sentence prediction functions of the system.

Participants were told to type freely during the text entry task (that is, not given instructions about text entry strategies for using predictions) and invited to a short interview about the text entry strategy the participant adopted after the task. Table 2 shows descriptions of the text entry strategies the eight participants adopted.

We observed three main types of text entry strategies. The first type used a mix of word and sentence predictions. For example, Participant 2, 3, and 8 relied extensively on the predictions for long words and/or words in phrases, and medium and long sentences. Further, Participant 5 and 7 actively used word and sentence predictions for almost every word and sentence. The second type mainly used sentence predictions. For example, Participant 1 used predictions for long sentences and only checked word prediction when they realized they were typing a long word. Participant 6 strongly depended on sentence prediction but only checked word predictions when they felt the system could accurately predict them. The third type did not use any prediction functions, such as Participant 4 who was almost completely reliant on letter-by-letter entry.

Table 3 shows that the KLM-BEI model can identify parameters that are affected by three types of human performance factors and extract user parameters based on the interaction log. These parameters are then applied back to the proposed KLM-BEI model and the conventional KLM model for envelope analyses. The results reveal a substantial improvement in that the new model's envelope analyses yield a more accurate text entry performance estimation in text entry rate and keystroke savings than using the conventional model. Figure 16 is an example of using parameters extracted from Participant 1 for envelope analyses. The correlated actual text entry strategy of Participant 1 is highlighted in purple circles in Figure 16a–d. By comparing Figure 16a and c, and Figure 16b and d with the real user text entry rate and keystroke savings, the envelope analyses using KLM-BEI demonstrate a substantial improvement of estimations (for example, entry rate is 25 wpm and the keystroke savings is 22%) that are closer to actual measurements (for example, entry rate is 27 wpm and the keystroke savings is 14%). The prior analysis overestimated the entry rate and keystroke savings (for example, 48 wpm for entry rate and 42% for keystroke savings). The other participants' resulting simulations using real user parameters share similar improvements. That is, the KLM-BEI-assisted envelope analysis yielded more accurate estimations.

6 DISCUSSION AND CONCLUSIONS

The rapid development of large language models (LLMs), such as ChatGPT, brings great opportunities to predictive AAC text entry system design for users with motor disabilities. However, such systems also introduce many complexities that make it difficult for designers to know *a priori* how to set parameters at appropriate values, such as the number of word and sentence suggestions, and understand what the requirements are on various subsystems, such as the accuracy required for word auto-complete. As system complexity increases, it is not viable to solely rely on the traditional use of text entry experiments, as such experiments can only test a few operating points. Further, some parameters that govern the joint human-system outcomes (entry rates, error rates, keystroke savings, and so on) are latent in the sense that they are directly connected to user strategies in, for example, leveraging word and sentence suggestions. Since we cannot directly control user strategies in experiments, we need to simulate various outcomes to assess which operating points our NLG-based AAC text entry systems may realize.

This paper contributes to this line of work by presenting the imperfect surrogate user model, KLM-BEI, which we use to generate performance envelopes across a wide range of parameters and strategies, and to extract user parameters from actual text entry. We extend prior work in using design engineering methods for text entry design [21, 23] by (1) analyzing an AAC text entry system assisted by word and sentence prediction functions; (2) incorporating human performance factors into the computational model to allow for analysis of imperfect user behavior (bounded rationality, human error, and interruption modeling); and (3) demonstrating a method for estimating parameter values for the model at runtime by analyzing user behavior.

We hope this work will stimulate further research. In particular, we see five particularly promising avenues of future work: (1) to explore alternative system parameters and user parameters, such as different prediction parameters and timings; (2) to study if this model could also be useful to inform mobile text entry design for able-bodied users; (3) to develop design tools that allow text entry designers to easily explore a wide range of text entry designs using the model in this paper and future work; (4) to investigate model refinements that take into account text entry systems that adapt to users' text entry activities; and (5) to examine the efficacy of the model in text entry design for particularly challenging areas of text entry, such as mid-air text entry in virtual and augmented reality and augmentative and alternative communication.

7 OPEN SCIENCE

The source code for the KLM-BEI simulation is available as supplementary material.

REFERENCES

[1] 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* (Denver, Colorado, USA) *(CHI '17)*. Association for Computing Machinery, New York, NY, USA, 4229–4241. https://doi.org/10.1145/3025453.3025695

[2] Nikola Banovic, Ticha Sethapakdi, Yasasvi Hari, Anind K. Dey, and Jennifer Mankoff. 2019. The Limits of Expert Text Entry Speed on Mobile Keyboards with Autocorrect. In *Proceedings of the 21st International Conference on Human-Computer Interaction with Mobile Devices and Services* (Taipei, Taiwan) *(MobileHCI '19)*. Association for Computing Machinery, New York, NY, USA, Article 15, 12 pages. https://doi.org/10.1145/3338286.3340126

[3] Xiaojun Bi, Yang Li, and Shumin Zhai. 2013. FFitts Law: Modeling Finger Touch with Fitts' Law. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Paris, France) *(CHI '13)*. Association for Computing Machinery, New York, NY, USA, 1363–1372. https://doi.org/10.1145/2470654.2466180

[4] Rolf Black, Joseph Reddington, Ehud Reiter, Nava Tintarev, and Annalu Waller. 2010. Using NLG and sensors to support personal narrative for children with complex communication needs. In *Proceedings of the NAACL HLT 2010 Workshop on Speech and Language Processing for Assistive Technologies*. Association for Computational Linguistics, Los Angeles, California, 1–9. https://aclanthology.org/W10-1301

[5] Jelmer P. Borst, Niels A. Taatgen, and Hedderik van Rijn. 2015. What Makes Interruptions Disruptive? A Process-Model Account of the Effects of the Problem State Bottleneck on Task Interruption and Resumption. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems* (Seoul, Republic of Korea) *(CHI '15)*. Association for Computing Machinery, New York, NY, USA, 2971–2980. https://doi.org/10.1145/2702123.2702156

[6] Shanqing Cai, Subhashini Venugopalan, Katrin Tomanek, Ajit Narayanan, Meredith Morris, and Michael Brenner. 2022. Context-Aware Abbreviation Expansion Using Large Language Models. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, Seattle, United States, 1261–1275. https://doi.org/10.18653/v1/2022.naacl-main.91

[7] Stuart K Card, Thomas P Moran, and Allen Newell. 1980. The keystroke-level model for user performance time with interactive systems. *Commun. ACM* 23, 7 (1980), 396–410. https://dl.acm.org/doi/pdf/10.1145/358886.358895

[8] Stuart K Card, Thomas P Moran, and Allen Newell. 1983. *The psychology of human-computer interaction.* Crc Press, USA. https://doi.org/10.1201/9780203736166

[9] Edward Clarkson, Kent Lyons, James Clawson, and Thad Starner. 2007. Revisiting and Validating a Model of Two-Thumb Text Entry. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. Association for Computing Machinery, New York, NY, USA, 163–166. https://doi.org/10.1145/1240624.1240650

[10] Andy Cockburn, Carl Gutwin, and Saul Greenberg. 2007. A Predictive Model of Menu Performance. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (San Jose, California, USA) *(CHI '07)*. Association for Computing Machinery, New York, NY, USA, 627–636. https://doi.org/10.1145/1240624.1240723

[11] Herbert A. Colle and Keith J. Hiszem. 2004. Standing at a kiosk: Effects of key size and spacing on touch screen numeric keypad performance and user preference. *Ergonomics* 47, 13 (2004), 1406–1423. https://doi.org/10.1080/00140130410001724228

[12] Nick C. Ellis. 2006. Language Acquisition as Rational Contingency Learning. *Applied Linguistics* 27, 1 (03 2006), 1–24. https://doi.org/10.1093/applin/ami038 arXiv:https://academic.oup.com/applij/article-pdf/27/1/1/446011/ami038.pdf

[13] Jacqui Fashimpaur, Kenrick Kin, and Matt Longest. 2020. Pinchtype: Text entry for virtual and augmented reality using comfortable thumb to fingertip pinches. In *Extended Abstracts of the 2020 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–7. https://doi.org/10.1145/3334480.3382888

[14] Dylan Gaines, Per Ola Kristensson, and Keith Vertanen. 2021. Enhancing the composition task in text entry studies: Eliciting difficult text and improving error rate calculation. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–8.

[15] Helen M Hodgetts and Dylan M Jones. 2006. Interruption of the Tower of London task: support for a goal-activation approach. *Journal of Experimental Psychology: General* 135, 1 (2006), 103. https://doi.org/10.1037/0096-3445.135.1.103

[16] Erik Hollnagel. 1998. *Cognitive reliability and error analysis method (CREAM)*. Elsevier, Amsterdam, Netherlands.

[17] Jussi Jokinen, Aditya Acharya, Mohammad Uzair, Xinhui Jiang, and Antti Oulasvirta. 2021. Touchscreen Typing As Optimal Supervisory Control. 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 720, 14 pages. https://doi.org/10.1145/3411764.3445483

[18] Heidi Horstmann Koester and Simon P Levine. 1994. Modeling the speed of text entry with a word prediction interface. *IEEE transactions on rehabilitation engineering* 2, 3 (1994), 177–187. https://doi.org/10.1037/h0056940

[19] Per Ola Kristensson. 2009. Five challenges for intelligent text entry methods. *AI Magazine* 30, 4 (2009), 85–85.

[20] Per Ola Kristensson. 2015. Next-generation text entry. *Computer* 48, 07 (2015), 84–87.

[21] Per Ola Kristensson, James Lilley, Rolf Black, and Annalu Waller. 2020. A design engineering approach for quantitatively exploring context-aware sentence retrieval for nonspeaking individuals with motor disabilities. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–11. https://doi.org/10.1145/3313831.3376525

[22] Per Ola Kristensson, Morten Mjelde, and Keith Vertanen. 2023. Understanding Adoption Barriers to Dwell-Free Eye-Typing: Design Implications from a Qualitative Deployment Study and Computational Simulations. In *Proceedings of the 28th International Conference on Intelligent User Interfaces*. Association for Computing Machinery, New York, NY, USA, 607–620.

[23] Per Ola Kristensson and Thomas Müllners. 2021. Design and Analysis of Intelligent Text Entry Systems with Function Structure Models and Envelope Analysis. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*. Association for Computing Machinery, New York, NY, USA, 1–12. https://doi.org/10.1145/3411764.3445566

[24] Per Ola Kristensson and Keith Vertanen. 2012. Performance comparisons of phrase sets and presentation styles for text entry evaluations. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces*. Association for Computing Machinery, New York, NY, USA, 29–32. https://doi.org/10.1145/2166966.2166972

[25] Janice Light, David McNaughton, David Beukelman, Susan Koch Fager, Melanie Fried-Oken, Thomas Jakobs, and Erik Jakobs. 2019. Challenges and opportunities in augmentative and alternative communication: Research and technology development to enhance communication and participation for individuals with complex communication needs. *Augmentative and Alternative Communication* 35, 1 (2019), 1–12. https://doi.org/10.1080/07434618.2018.1556732

[26] I Scott MacKenzie. 1992. Fitts' law as a research and design tool in human-computer interaction. *Human-computer interaction* 7, 1 (1992), 91–139. https://doi.org/10.1207/s15327051hci0701_3

[27] I. Scott MacKenzie, Tatu Kauppinen, and Miika Silfverberg. 2001. Accuracy Measures for Evaluating Computer Pointing Devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Seattle, Washington, USA) *(CHI '01)*. Association for Computing Machinery, New York, NY, USA, 9–16. https://doi.org/10.1145/365024.365028

[28] I Scott MacKenzie and R William Soukoreff. 2003. Phrase sets for evaluating text entry techniques. In *CHI'03 extended abstracts on Human factors in computing systems*. Association for Computing Machinery, New York, NY, USA, 754–755.

[29] Karissa Marble-Flint, Holli Steiner, and Ashly Elliott2 Megan Stein. 2021. Implementing Words Their Way with an Adolescent Who Uses AAC: A Case Study. *London International Conference on Education* 12, 1 (2021), 3480–3485. https://doi.org/10.20533/licej.2040.2589.2021.0458

[30] David McNaughton, Janice Light, David R Beukelman, Chris Klein, Dana Nieder, and Godfrey Nazareth. 2019. Building capacity in AAC: A person-centred approach to supporting participation by people with complex communication needs. *Augmentative and Alternative Communication* 35, 1 (2019), 56–68. https://doi.org/10.1080/07434618.2018.1556731

[31] Christopher A Monk, J Gregory Trafton, and Deborah A Boehm-Davis. 2008. The effect of interruption duration and demand on resuming suspended goals. *Journal of experimental psychology: Applied* 14, 4 (2008), 299. https://doi.org/10.1037/a0014402

[32] Donald A Norman. 1981. Categorization of action slips. *Psychological review* 88, 1 (1981), 1.

[33] M Norré. 2020. Evaluation of a Word Prediction System in an Augmentative and Alternative Communication for Disabled People. *International Information and Engineering Technology Association* 81, 1-4 (2020), 49–54. https://doi.org/10.18280/mmc_c.811-409

[34] OpenAI. 2022. *Introducing ChatGPT*. OpenAI. Retrieved Aprol 6, 2023 from https://openai.com/blog/chatgpt

[35] OpenAI. 2023. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[36] Antti Oulasvirta, Per Ola Kristensson, Xiaojun Bi, and Andrew Howes. 2018. *Computational Interaction*. Oxford University Press, Oxford, United Kingdom. https://doi.org/10.1093/oso/9780198799603.001.0001

[37] Martin Pielot, Karen Church, and Rodrigo de Oliveira. 2014. An in-situ study of mobile phone notifications. In *Proceedings of the 16th international conference on Human-computer interaction with mobile devices & servicesSeptember*. Association for Computing Machinery, New York, NY, USA, 233–242. https://doi.org/10.1145/2628363.2628364

[38] Philip Quinn and Shumin Zhai. 2016. A Cost-Benefit Study of Text Entry Suggestion Interaction. 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, 83–88. https://doi.org/10.1145/2858036.2858305

[39] James Reason. 1990. *Human error*. Cambridge university press, UK.

[40] Jochen Rick. 2010. Performance Optimizations of Virtual Keyboards for Stroke-Based Text Entry on a Touch-Based Tabletop. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology* (New York, New York, USA) *(UIST '10)*. Association for Computing Machinery, New York, NY, USA, 77–86. https://doi.org/10.1145/1866029.1866043

[41] Frank E Ritter, Gordon D Baxter, and Elizabeth F Churchill. 2014. *Foundations for designing user-centered systems*. Springer, London.

[42] Sayan Sarcar, Jussi P.P. Jokinen, Antti Oulasvirta, Zhenxin Wang, Chaklam Silpasuwanchai, and Xiangshi Ren. 2018. Ability-Based Optimization of Touchscreen Interactions. *IEEE Pervasive Computing* 17, 1 (2018), 15–26. https://doi.org/10.1109/MPRV.2018.011591058

[43] Andrew Sears, Julie A. Jacko, Josey Chu, and Francisco Moro. 2001. The role of visual search in the design of effective soft keyboards. *Behaviour & Information Technology* 20, 3 (2001), 159–166. https://doi.org/10.1080/01449290110049790 arXiv:https://doi.org/10.1080/01449290110049790

[44] Junxiao Shen, Boyin Yang, John J Dudley, and Per Ola Kristensson. 2022. KWickChat: A Multi-Turn Dialogue System for AAC Using Context-Aware Sentence Generation by Bag-of-Keywords. In *27th International Conference on Intelligent User Interfaces*. Association for Computing Machinery, New York, NY, USA, 853–867. https://doi.org/10.1145/3490099.3511145

[45] Herbert A. Simon. 1956. Rational choice and the structure of the environment. *Psychological Review* 63, 2 (1956), 129–138. https://doi.org/10.1037/h0042769

[46] Herbert A Simon. 1984. Models of bounded rationality, volume 1: economic analysis and public policy. *MIT Press Books* 6, 3 (1984), 308–308.

[47] Herbert A. Simon. 1990. *Bounded Rationality*. Palgrave Macmillan UK, London, 15–18. https://doi.org/10.1007/978-1-349-20568-4_5

[48] R. Willliam Soukoreff and I. Scoot MacKenzie. 1995. Theoretical upper and lower bounds on typing speed using a stylus and a soft keyboard. *Behaviour & Information Technology* 14, 6 (1995), 370–379. https://doi.org/10.1080/01449299508914656 arXiv:https://doi.org/10.1080/01449299508914656

[49] Keith Vertanen and Per Ola Kristensson. 2011. The imagination of crowds: conversational AAC language modeling using crowdsourcing and large data sources. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, USA, 700–711. https://aclanthology.org/D11-1065

[50] Keith Vertanen and Per Ola Kristensson. 2011. A versatile dataset for text entry evaluations based on genuine mobile emails. In *Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*. Association for Computing Machinery, New York, NY, USA, 295–298.

[51] Keith Vertanen and Per Ola Kristensson. 2014. Complementing text entry evaluations with a composition task. *ACM Transactions on Computer-Human Interaction (TOCHI)* 21, 2 (2014), 1–33.

[52] Nadine Vigouroux, Frédéric Vella, Philippe Truillet, and Mathieu Raynal. 2004. Evaluation of AAC for text input by two groups of subjects: able-bodied subjects and disabled motor subjects. In *8th ERCIM Workshop*. User Interface for All, Vienna, Austria, 28–29. https://hal.science/hal-03627614

[53] Annalu Waller. 2019. Telling tales: unlocking the potential of AAC technologies. *International journal of language & communication disorders* 54, 2 (2019), 159–169. https://doi.org/10.1111/1460-6984.12449

[54] J.H. Williamson, A. Oulasvirta, P.O. Kristensson, and N. Banovic. 2022. *Bayesian Methods for Interaction and Design*. Cambridge University Press, Cambridge, United Kingdom. https://books.google.co.uk/books?id=voKAEAAAQBAJ

[55] B Yang and PO Kristensson. 2023. Tinkerable Augmentative and Alternative Communication for Users and Researchers. In *Design for Sustainable Inclusion*, Joy Goodman-Deane, Hua Dong, Ann Heylighen, Jonathan Lazar, and John Clarkson (Eds.). Springer International Publishing, Cham, 137–145.

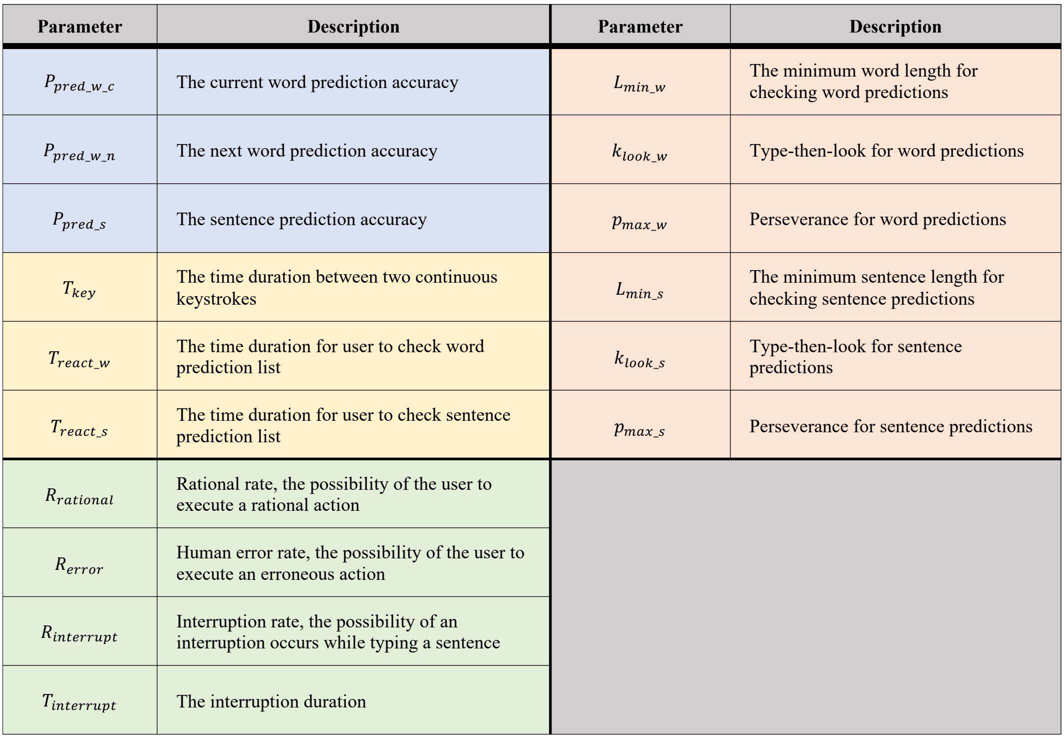
[56] Boyin Yang and Per Ola Kristensson. 2023. A Demonstration of a Tinkerable Augmentative and Alternative Communication Keyboard. In *Companion Proceedings of the 28th International Conference on Intelligent User Interfaces* (Sydney, NSW, Australia) *(IUI '23 Companion)*. Association for Computing Machinery, New York, NY, USA, 138–140. https://doi.org/10.1145/3581754.3584153

[57] Richard M Young, Thomas RG Green, and Tony Simon. 1989. Programmable user models for predictive evaluation of interface designs. *ACM SIGCHI Bulletin* 20, SI (1989), 15–19. https://doi.org/10.1145/67450.67453

[58] Shumin Zhai and Per-Ola Kristensson. 2003. Shorthand Writing on Stylus Keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems* (Ft. Lauderdale, Florida, USA) *(CHI '03)*. Association for Computing Machinery, New York, NY, USA, 97–104. https://doi.org/10.1145/642611.642630

[59] Shumin Zhai, Per-Ola Kristensson, and Barton A Smith. 2005. In search of effective text input interfaces for off the desktop computing. *Interacting with computers* 17, 3 (2005), 229–250.

A PARAMETERS AND DESCRIPTIONS



Parameter	Description	Parameter	Description
$P_{pred\_w\_c}$	The current word prediction accuracy	L_{min_w}	The minimum word length for checking word predictions
$P_{pred\_w\_n}$	The next word prediction accuracy	k_{look_w}	Type-then-look for word predictions
P_{pred_s}	The sentence prediction accuracy	p_{max_w}	Perseverance for word predictions
T_{key}	The time duration between two continuous keystrokes	L_{min_s}	The minimum sentence length for checking sentence predictions
T_{react_w}	The time duration for user to check word prediction list	k_{look_s}	Type-then-look for sentence predictions
T_{react_s}	The time duration for user to check sentence prediction list	p_{max_s}	Perseverance for sentence predictions
$R_{rational}$	Rational rate, the possibility of the user to execute a rational action		
R_{error}	Human error rate, the possibility of the user to execute an erroneous action		
$R_{interrupt}$	Interruption rate, the possibility of an interruption occurs while typing a sentence		
$T_{interrupt}$	The interruption duration		

Table 4. A list of 16 parameters used for envelope analyses in this paper. The parameters shaded in light blue are system parameters defining prediction accuracy. The ones shaded in light yellow are user parameters for the time duration of typing and checking predictions. The ones shaded in light orange are text entry strategy parameters. These three types of parameters are used for both KLM and KLM-BEI. The remaining parameters shaded by light green are used for KLM-BEI only, which are human factors-related, indicating the possibility that each type of human factor may occur during the interaction.

Received January 2023; revised May 2023; accepted June 2023