The Open Rehabilitation Journal, **2011**, *4*, **1-12**

Modeling One-Switch Row-Column Scanning with Errors and Error Correction Methods

Richard C Simpson*[,1,2], Robert Mankowski[1] and Heidi H. Koester[3]

[1]*Department of Rehabilitation Science and Technology, University of Pittsburgh, Pittsburgh, PA, USA*

[2]*VA Pittsburgh Healthcare System, Pittsburgh, PA, USA*

[3]*Koester Performance Research, Ann Arbor, MI, USA*

Abstract: Single switch scanning has lots of different configuration options. One way to choose the most appropriate configuration for a client is to use a model to predict performance under different configurations. Most existing models expect error-free performance, however, and none integrates all the types of errors that can occur and the variety of error-correction methods that are available. A model is presented which predicts user performance for single-switch row-column scanning with errors and error-correction methods. The model is used to draw conclusions about the utility of different error-correction methods.

Keywords: Augmentative communication, user modeling, assistive technology.

1. INTRODUCTION

Single-switch scanning is used by individuals with severe motor and communicative disabilities as a method for entering text and data into computers and augmentative communication (AAC) devices. The impairments may stem from a variety of medical conditions, such as cerebral palsy, traumatic brain injury, muscular dystrophy, and/or neuromuscular diseases such as multiple sclerosis and amyotrophic lateral sclerosis, affecting hundreds of thousands of people in the U.S. Single switch scanning is an extremely slow method of text entry. A very fast user may achieve 8 words per minute [1-4], while rates of 1 word per minute and lower are common [5-7]. These slow rates create a major obstacle to users' communication and personal achievement. Despite its limitations, however, row-column scanning fills an important niche within technology access methods by providing an affordable alternative for individuals with limited movement and vocal abilities. Hence, despite increasing interest in speech recognition, eye-tracking, and direct-brain interfaces for accessing assistive technology, there remain valid reasons for seeking to enhance performance using row-column scanning.

2. CONFIGURATION OPTIONS

Product developers have implemented numerous features and configuration options to allow for customization of scanning software, with the goal of increasing text entry rate (TER) on an individualized basis. Some of the configuration options available in current one-switch scanning systems are shown in Table **1**. Most of the configuration options are self-explanatory, but a few deserve more attention.

Items in a scanning matrix can be either *static* or *dynamic*. Static items always produce the same output. Examples of static keyboard items are:

- Text entry (e.g., letters, numbers, punctuation)
- Editing (e.g., backspace, delete, arrow keys)
- Modifier keys (e.g., shift, control, alt)
- Error correction (e.g., stop scanning, continue scanning)
- Navigation (e.g., a key that opens up a new keyboard layout, reverse scan direction)

Dynamic items, on the other hand, change based on the current context. Examples of dynamic keyboard items are character prediction and word prediction.

The *scanning mode* dictates how the system responds to the switch. In *automatic* scanning, a switch press indicates a selection. In *inverse* scanning, the user activates the switch until the desired target is highlighted, and then releases the switch to make a selection. In *step* scanning, a long switch activation advances the highlight to the next item and a short switch activation indicates a selection.

The *scanning pattern* determines the order in which items are highlighted. In *linear* scanning, each item is highlighted in turn, until an item is selected. Linear scanning is the slowest method of scanning. In *grouped* scanning, progressively smaller groups of items are highlighted until a selection is made. The most common implementation of grouped scanning is *row-column* scanning, in which each row within the matrix is sequentially highlighted until the user selects the row containing the desired item by activating the switch. The columns within the selected row are then scanned until the target item is highlighted and can be selected by activating the switch a second time.

---

*Address correspondence to this author at the Bakery Square, Suite 401, Rm 15; 6425 Penn Ave, University of Pittsburgh, PA, USA; Tel: 412-383-6593; Fax: 412-624-6501; E-mail: ris20@pitt.edu

In most systems, once the user makes a selection, scanning resumes at the topmost group (i.e., the first row for row-column scanning). Some systems provide alternatives however, such as resuming scanning at the group where the selection took place (i.e., at the row where the selection occurred) or at the item that was selected (i.e., at the column that was previously selected).

When a user selects the wrong group, there must be a mechanism for correcting the error. The options available on commercial products are:

1. A fixed *loop count* that defines the number of times the items within a group are scanned before scanning recommences
2. A *stop scanning* item (usually at the beginning of the row)
3. Activating the switch for an extended time
4. Selecting an (incorrect) item within the row

When the user selects the correct group, but fails to make a selection within that group, there must be some way to cause the system to scan through the group again. The options available on commercial products are:

1. A fixed loop count
2. A *continue scanning* item at the end of the row that can be selected to re-initiate scanning through the row

Finally, some systems allow the user to register multiple switch activations when a switch is activated for an extended period of time.

3. PROBLEM STATEMENT

Proper configuration of the features available within scanning systems can make a major difference. For example, different scanning patterns and modes are most appropriate for individual clients [8, 9]. From a face validity standpoint, the timing parameters are a key factor in TER, particularly the scan rate, and this has been confirmed empirically [2]. If a person is capable of using a scan rate of 1.0 seconds, say, but their system is set to 2.0 seconds, their TER will be only half of what it could be. A case study by Koester (1990) [10] demonstrated how modifications to both item layout and scan rate yielded a TER enhancement of 321% for one individual. For experienced users of single-switch scanning, who have been proactive in adjusting their system configurations, gains may be more modest. For example, the five individuals in Bhattacharya’s study [13] showed differences of 20 to 25% when using different configurations.

A real barrier to progress is the lack of an effective and efficient method for tailoring a configuration to a particular user. The current standard of care is for clinicians and users to arrive at appropriate settings by trial and error. This makes it difficult if not impossible to effectively define an optimal configuration. Often, so much time is spent just identifying a reliable switch site and a basic scan layout appropriate for the user’s needs that very little time is left to properly adjust the remaining options. Proper setting of the software parameters by trial and error would likely require many hours, if not a full day or more. This is simply not a practical solution. The result is that many end up using a system under its default configuration.

Table 1. Configuration Options Found in 16 Commercially-Available Scanning Interfaces [5]

Setting	Supported by %	Explanation
Scan Rate	100%	The amount of time an item is available for selection (i.e., highlighted)
Recovery Delay	50%	An additional delay added to the first row or column to provide time for the user to recover from a previous switch activation. Different values may be used for rows and columns in some systems.
Loop Count	81%	Determines how many times the system will scan through the columns within a row before resuming between rows
Reverse Scan	19%	The ability to reverse the direction of scanning through a row
Stop Scanning	38%	The ability to stop scanning a row by selecting an item at the beginning or end of each row
Re-Scan	19%	The ability to re-scan the row by selecting an item at the beginning or end of each row
Automatic/Manual Scan Initiation	88%	Determines whether the user must press a switch to initiate scanning, or if scanning is automatic (and continuous). This setting dictates whether two or three switch presses are required to make a selection.
Switch Repeat	50%	Allows user to hold the switch down to register multiple switch activations.
Repeat Delay	50%	How long the switch must be held down to register the second activation.
Repeat Rate	44%	The length of time between switch activations after the second activation is registered.
Acceptance Delay	69%	The length of time a switch must be activated before the activation is registered.
Switch Hold Escape	6%	The length of time a switch must be held before an exit/escape of the current row or column occurs. Scanning restarts at the top of matrix.
Character Prediction	13%	One or more items in the matrix are dynamically updated based on which letters are most likely to be selected next.
Word Completion / Prediction	100%	One or more items in the matrix are dynamically updated based on what word the user is most likely entering or is likely to enter next.

An alternative is to use models of user performance to select the most appropriate settings. Several models of scanning have been published [11-17], but they are not complete. Most focus on only one scan pattern (typically row/column scanning), none consider the range of scanning control and error correction methods available in real products, and none integrate the likelihood of errors into their predictions of TER. In fact, up to 63% of selections can involve some type of error [5, 12, 13]. Without incorporating errors and their consequences, a model can significantly overestimate performance and cannot suggest accurate correction strategies.

4. RELATED RESEARCH

Damper was one of the earliest investigators to model row-column scanning [15]. He developed equations based on the number of scan steps to each matrix item and the frequency with which each item is selected. He assumed error-free performance and did not consider the effects of errors or error correction methods on selection time.

Lesher [16] and Venkatagiri [17] used computer software to simulate text entry, rather than developing equations to calculate text entry rate. The simulations were used to compare different keyboard layouts, scan patterns, and text entry rate enhancement methods (character prediction and word prediction). Both simulations assumed perfect performance by the user.

Abascal studied the effects of errors in a D-dimensional scanning system [11] by calculating the delay introduced by an error and adding that to the time for an error-free selection. Abascal considered errors of *commission* (pressing the switch at the wrong time) but not errors of *omission*. Within errors of commission, Abascal only considered selecting the wrong group, and not the wrong item. The only type of error-correction mechanism he considered was a “stop scanning item.” Abascal assumed a single probability *e*, for the occurrence of all types of errors.

Most recently, Bhattacharya [14] empirically evaluated two models of scanning performance (one for one-switch scanning, one for two-switch scanning) with 6 disabled and 2 able-bodied subjects. The model assumed error-free performance, and the investigators removed erroneous selections from the data log. For one-switch scanning, model error when predicting error-free performance ranged from 3% to 10% for 3 subjects with disabilities and ranged from 17% and 19% for able-bodied subjects. For two-switch scanning, model error for error free performance ranged from 1% to 9% for subjects with disabilities and 11% to 23% for able-bodied subjects. Bhattacharya also developed a model of the occurrence of timing errors (errors of omission) and selection errors (errors of commission) during scanning [12, 13] but has not yet combined his performance model with his error model to predict actual text entry rate.

5. MODELING ROW-COLUMN SCANNING

Our modeling approach is similar to the approach taken by other models [11, 13, 15-17]. The time required to select a given item is the sum of the time required to scan to the item and the time required to press the switch the required number of times. Our model also includes the delay imposed by each type of error and error correction method, along with the likelihood of each error occurring. For example, if the user fails to select the target row the first time it is highlighted, the system will scan through all the rows in the matrix once and then scan through the rows again until it reaches the target row. The delay (D) due to the timing error in this case is the scan rate (T_s) multiplied by the number of rows (r):

$$D = T_s \cdot r$$

Average selection time ($\bar{T}_{ij}$) for the item in row i and column j is the sum of the time for an error free selection (T_{ij}) and the delay (D_x) associated with each type of error (e_x) multiplied by the probability of each error’s occurrence ($P(e_x)$):

$$\bar{T}_{ij} = T_{ij} + \sum_x D_x \cdot P(e_x)$$

The average selection time for each item in the matrix is then weighted by that item’s frequency of use (F_{ij}) to calculate an average selection time ($\bar{T}$):

$$\bar{T} = \sum_i \sum_j \bar{T}_{ij} \cdot F_{ij}$$

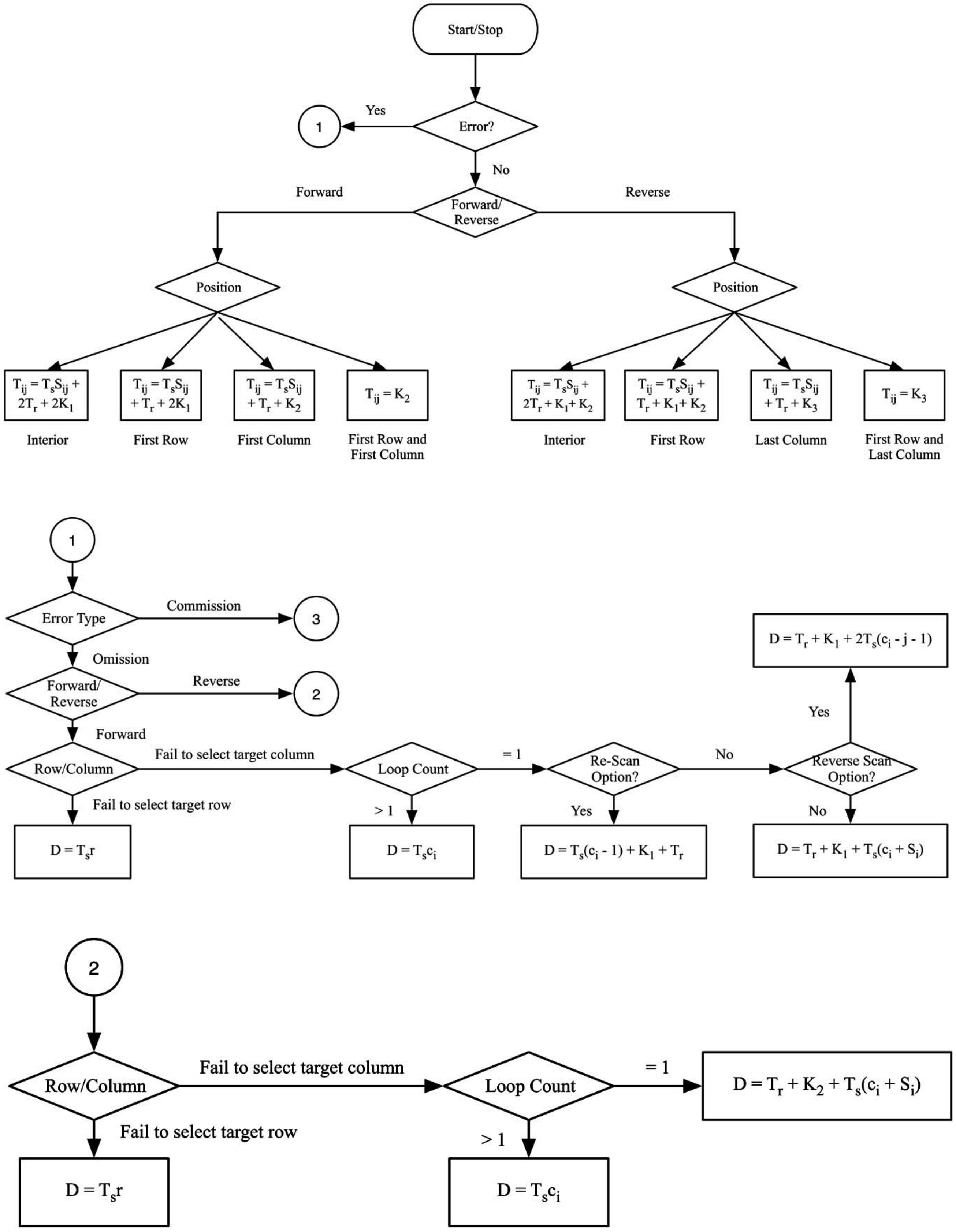
TER in words per minute is a function of the average selection time and the average selections per word ($\bar{W}$):

$$TER = \frac{1}{\bar{T}} \cdot \frac{1}{\bar{W}} \cdot 60$$

A flow chart of the model is shown in Fig. (**1**). The variables associated with the model are described in Table **2**.

Every user model makes assumptions, and ours is no different. Our model assumes:

- The user never makes two mistakes in the same selection attempt (i.e., the user never selects the wrong row and then selects the wrong column; the user never selects two incorrect rows). The model could be expanded to accommodate this by adding in additional probabilities for two error sequences, but it would get very complicated.
- Scanning starts automatically (i.e., the user does not have to activate the switch to initiate scanning). The model could be modified to accommodate this by adding in another switch press.
- When the user aborts scanning through the wrong row, scanning resumes at the top of the matrix. Similarly, when the user selects an item (right or wrong), scanning resumes at the top of the matrix We could implement other behaviors by using a look-up table to determine the number of scan steps to the target item based on the previously selected item.
- When reverse scanning is an option, the user uses it optimally (i.e., always uses reverse scanning when it's the faster selection method and never uses reverse scanning when it's the slower scanning method). We could address this in the model by adding in a probability of using forward and reverse scanning for each item.


Start/Stop
Error?
Yes
1
No
Forward
Forward/ Reverse
Reverse
Position
Position
$T_{ij} = T_sS_{ij} + 2T_r + 2K_1$
Interior
$T_{ij} = T_sS_{ij} + T_r + 2K_1$
First Row
$T_{ij} = T_sS_{ij} + T_r + K_2$
First Column
$T_{ij} = K_2$
First Row and First Column
$T_{ij} = T_sS_{ij} + 2T_r + K_1 + K_2$
Interior
$T_{ij} = T_sS_{ij} + T_r + K_1 + K_2$
First Row
$T_{ij} = T_sS_{ij} + T_r + K_3$
Last Column
$T_{ij} = K_3$
First Row and Last Column
1
Error Type
Commission
3
Omission
Forward/ Reverse
Reverse
2
Forward
Row/Column
Fail to select target column
Fail to select target row
$D = T_sr$
Loop Count
= 1
> 1
$D = T_sc_i$
Re-Scan Option?
Yes
$D = T_s(c_i - 1) + K_1 + T_r$
No
Reverse Scan Option?
Yes
$D = T_r + K_1 + 2T_s(c_i - j - 1)$
No
$D = T_r + K_1 + T_s(c_i + S_i)$
2
Row/Column
Fail to select target column
Fail to select target row
$D = T_sr$
Loop Count
= 1
$D = T_r + K_2 + T_s(c_i + S_i)$
> 1
$D = T_sc_i$

(Fig. 1) contd…..

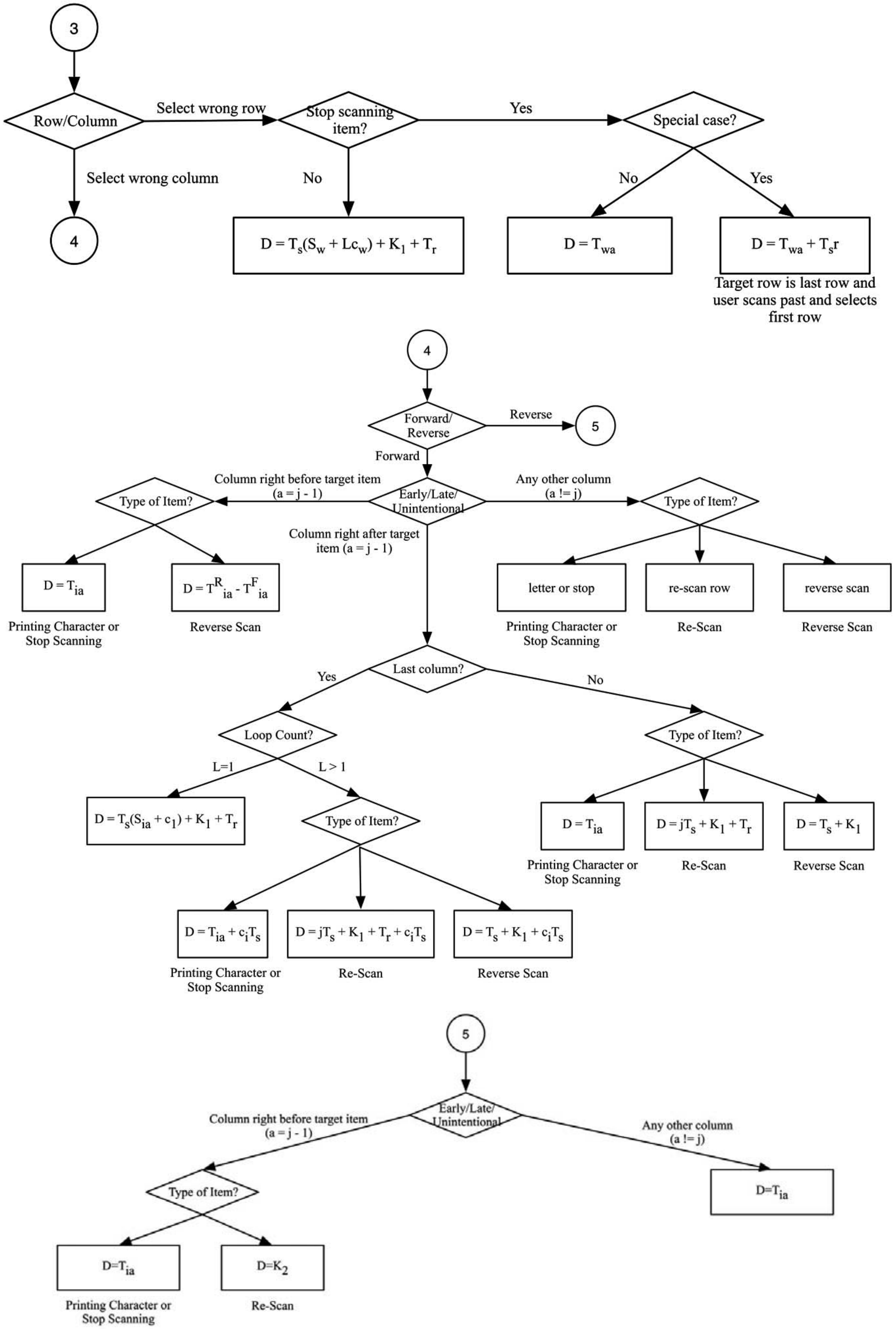


Fig. (1). Graphical representation of row-column scanning model

Table 2. Variables Used in the Model

Quantity	Var	Units
Minimum time to select the item in row i and column j with no error	T_{ij}	seconds
Time to select the item in row i and column j with no error when scanning forward	T^F_{ij}	seconds
Time to select the item in row i and column j with no error when using a reverse-scan item	T^R_{ij}	seconds
Frequency that the item in row i and column j is the final target	F_{ij}	
Scan rate	T_s	seconds/scan period
Scan steps to item in row i and column j	S_{ij}	
Scan steps to row i	S_i	
Recovery delay	T_r	seconds
Single switch press time	K_1	seconds
Double switch press time	K_2	seconds
Triple switch press time	K_3	seconds
X switch presses in a row	K_x	seconds
Time switch must be held down to register an activation	K_H	seconds
Time switch must be held to initiate repeated selection	K_R	seconds
Number of rows in matrix	r	
Number of columns in row i	c_i	
Loop count	L	
Probability of error-free selection	P_c	
Probability of not pressing switch	P_n	
Probability of pressing switch too early	P_e	
Probability of pressing switch too late	P_l	
Probability of pressing switch unintentionally	P_u	
Probability of detecting error	P_d	
Probability of fixing error	P_f	
Average penalty per selection when switch not pressed on target row	$D_{n,row}$	
Average penalty per selection when switch is pressed before target row	$D_{e,row}$	
Average penalty per selection when switch is pressed after target row	$D_{l,row}$	
Average penalty per selection when unintented row is selected	$D_{u,row}$	
Average penalty per selection when switch not pressed on target column	$D_{n,col}$	
Average penalty per selection when switch is pressed before target column	$D_{e,col}$	
Average penalty per selection when switch is pressed after target column	$D_{l,col}$	
Average penalty per selection when unintended column is selected	$D_{u,col}$	
Average number of selections per word	$\bar{W}$	
Average number of seconds to select the item at row *i* and column *j*	$\bar{T}_{ij}$	
Average number of seconds per selection	$\bar{T}$	
Average text entry rate	TER	

- The recovery delay is only applied after a switch press, not every time scanning loops back to the first row or first column. We could address this in the model by adding in an extra recovery delay for these situations.

6. SIMULATING PERFORMANCE WITH ROW/ COLUMN SCANNING KEYBOARDS

To evaluate the models, we performed a series of simulations to determine whether the results of the simulations corresponded to expectations.

6.1. Comparison to the Damper Model

Our first evaluation of our model was a comparison to Damper's work [15], to see if we could reproduce his results. We ran the model with the matrix shown in Fig. (**2**), the letter frequencies from Damper's article, and the assumptions shown in Table **3**.

Sp	E	A	R	D	U	V
T	O	I	L	G	K	
N	S	F	Y	X		
H	C	P	J			
M	W	Q				
B	Z					

Fig. (2). Damper's row-column scanning matrix.

Table 3. Assumptions Made by Damper Model

Press down time	K_d	0.25
Press hold time	K_h	0.00
Press release time	K_u	0.00
Press recovery time	K_r	0.00
Switch Press Time	K_1	0.25
Double-press time	K_2	0.5
Triple-press	K_3	0.75
Scan Rate	T_s	0.5
Recovery Delay	T_r	0.00
Probability of error-free selection	P_c	100%
Probability of not pressing switch	P_n	0%
Probability of pressing switch too early	P_e	0%
Probability of pressing switch too late	P_l	0%
Probability of pressing switch unintentionally	P_u	0%
Selections per word	$\bar{W}$	5.74

Based on the assumptions in Table **3**, our model produced a TER of 6.45 wpm, which is almost identical to the estimate of 6.48 arrived at by Damper [15].

6.2. Simulating Matrices with Different Error Correction Methods

Our next step was to simulate matrices using different combinations of error correction methods. Below are the matrices we used for the simulations.

Sp	E	A	N	D	W	V
T	O	S	L	F	K	
I	H	C	G	J		
R	U	Y	X			
M	P	Q				
B	Z	BkSp				

Fig. (3). Default matrix.

Reverse	Sp	E	A	N	D	W	V
Reverse	T	O	S	L	F	K	
Reverse	I	H	C	G	J		
Reverse	R	U	Y	X			
Reverse	M	P	Q				
Reverse	B	Z	BkSp				

Fig. (4). Matrix with reverse scan item at the start of each row.

Stop	Sp	E	A	N	D	W	V
Stop	T	O	S	L	F	K	
Stop	I	H	C	G	J		
Stop	R	U	Y	X			
Stop	M	P	Q				
Stop	B	Z	BkSp				

Fig. (7). Matrix with a stop scanning item at the front of each row.

Sp	E	A	N	D	W	V	Re-scan
T	O	S	L	F	K	Re-scan	
I	H	C	G	J	Re-scan		
R	U	Y	X	Re-scan			
M	P	Q	Re-scan				
B	Z	BkSp	Re-scan				

Fig. (5). Matrix with re-scan item at the end of each row.

Reverse	Sp	E	A	N	D	W	V	Re-scan
Reverse	T	O	S	L	F	K	Re-scan	
Reverse	I	H	C	G	J	Re-scan		
Reverse	R	U	Y	X	Re-scan			
Reverse	M	P	Q	Re-scan				
Reverse	B	Z	BkSp	Re-scan				

Fig. (6). Matrix with reverse scan item at the beginning of each row and a re-scan item at the end of each row.

Reverse	Sp	E	A	N	D	W	V	Stop
Reverse	T	O	S	L	F	K	Stop	
Reverse	I	H	C	G	J	Stop		
Reverse	R	U	Y	X	Stop			
Reverse	M	P	Q	Stop				
Reverse	B	Z	BkSp	Stop				

Fig. (8). Matrix with reverse scan item at start of each row and stop scan at end.

Stop	Sp	E	A	N	D	W	V	Re-scan
Stop	T	O	S	L	F	K	Re-scan	
Stop	I	H	C	G	J	Re-scan		
Stop	R	U	Y	X	Re-scan			
Stop	M	P	Q	Re-scan				
Stop	B	Z	BkSp	Re-scan				

Fig. (9). Matrix with stop scanning item at front of each row and rescan at end.

We assumed a switch press time of one second (i.e., $K_1 = 1$), which produces a scan rate of 1.54 seconds ($T_s = 1.54$) by the .65 rule. We also assumed that, when the loop count is greater than one ($L > 1$), there would be no need for a re-scan item. Finally, we assumed that probabilities were the same for rows and columns. In other words, the probability of selecting an item early, late or unintentionally was the same regardless of whether the item was a row or a column.

The following items were varied across simulations:

- Matrix items. Matrices were designed with a combination of stop scan, reverse scan, and re-scan items. A matrix with none of these items was considered the "Default" matrix.
- Loop count. The loop count was either 1 or 2.
- Recovery Delay. The recovery delay was either 0.0 or 0.5
- Error of Omission. P_n was set to 0%, 10%, 20%, or 30%. When P_n was greater than 0% all other error probabilities were zero (i.e., $P_u = P_l = P_e = 0$)
- Error of commission (unintentional) P_u was set to 0%, 10%, 20%, or 30%. When P_u was greater than 0% all other error probabilities were zero (i.e., $P_n = P_l = P_e = 0$)
- Error of commission (right after target item). P_l was set to 0%, 10%, 20%, or 30%. When P_l was greater than 0% all other error probabilities were zero (i.e., $P_n = P_u = P_e = 0$)
- Error of commission (right before target item). P_e was set to 0%, 10%, 20%, or 30%. When P_e was greater than 0% all other error probabilities were zero (i.e., $P_n = P_u = P_l = 0$)

A stop scanning item should only be useful when the user has selected the wrong row. Therefore, the stop scanning option should not be useful when the likelihood of an error of omission (Pn) is high, but should be useful when the likelihood of an error of commission (Pu, Pl or Pe) is high. A reverse scanning item should only be useful when the target item is at the end of the row. Since the matrix is arranged by letter frequency, target items should be concentrated towards the beginning of each row, so we would therefore expect reverse scanning to not be helpful for any of the conditions. A re-scan item should only be useful when the loop count is 1, the user has selected the correct row, and then fails to select the correct column. Re-scan should be most useful when the likelihood of an error of omission (Pn) is high, and least useful when the likelihood of an error of commission (Pu, Pl or Pe) is high.

7. RESULTS

As P_n increases, the user is more likely not to press the switch at the correct time (i.e., an error of ommission). As shown in Figs. (**10**, **11**), as P_n increased the stop scanning and reverse scanning options did not help.When the user fails to press the switch when the target row is highlighted, the system must re-scan the rows and none of the column-level error correction methods make a difference.

When the user fails to press the switch when the target column is highlighted, performance is determined by the loop count setting (L). When the loop count was greater than one, the best option was to use the default matrix without stop scanning or reverse scanning. When the loop count was one, the rescan item was helpful, but not much more helpful than the default matrix.

As P_u, P_e or P_l increase, the user is more likely to press the switch at the wrong time. None of the error correction methods will help if the user selects the wrong column. The error corrections only matter if the user selects the wrong row. As shown in Figs. (**12-17**), P_u, P_e and P_l increased, the stop scan item was increasingly helpful.

As shown in Figs. (**10-17**), changing T_r from 0 to 0.5 did not lead to a large difference in performance. Switch press time (T_s) has a much greater influence than T_r, because T_r is only applied after switch presses but T_s is applied at every scan step. Therefore, there’s not nearly as much penalty in using a “generous” recovery delay as there is in using a large T_s.

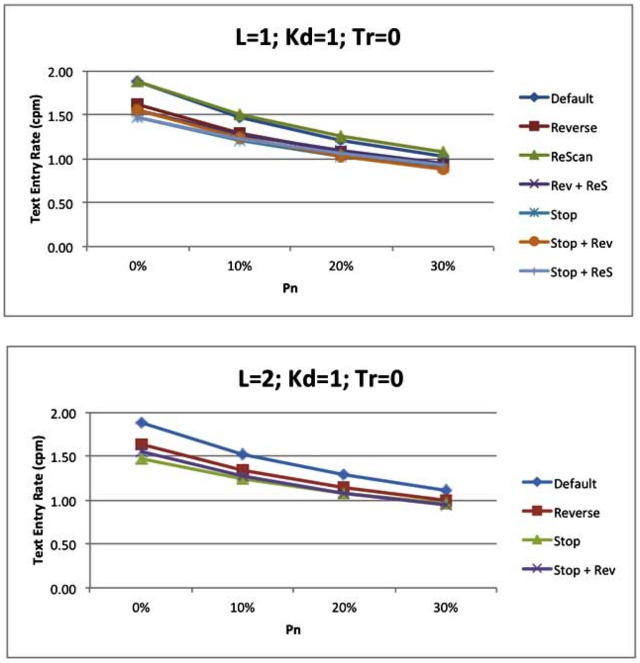


Fig. (10). Effect on Text Entry Rate (measured in characters per minute) of varying P_n, with no recovery delay ($T_r = 0$), $P_u = P_l = P_e = 0$.

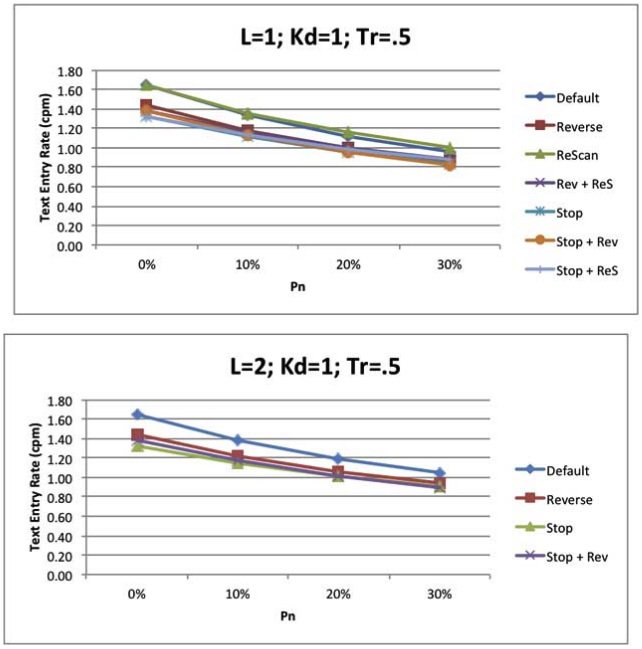


Fig. (11). Effect on Text Entry Rate (measured in characters per minute) of varying P_n, with 0.5 second recovery delay ($T_r = 0.5$), $P_u = P_l = P_e = 0$.

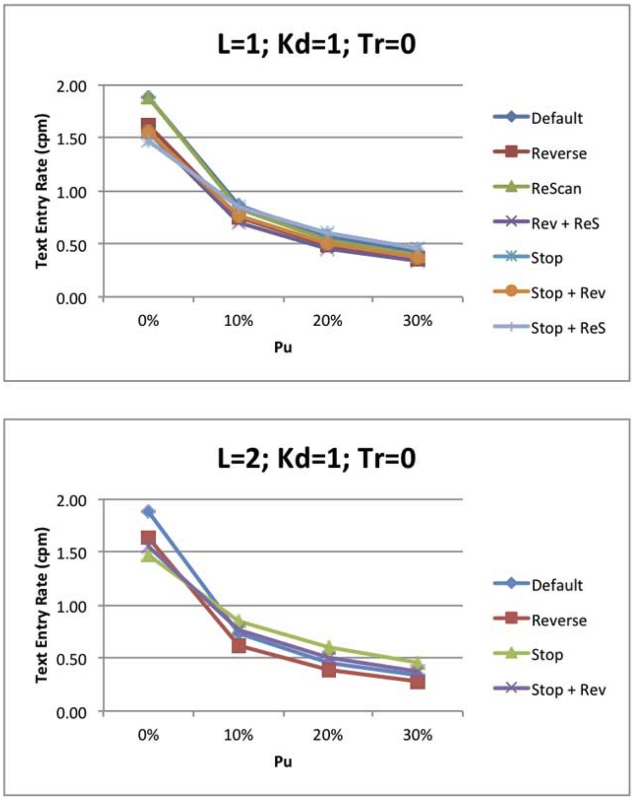


Fig. (12). Effect on Text Entry Rate (measured in characters per minute) of varying P_u, with no recovery delay ($T_r = 0$; $P_n = P_e = P_l = 0$).

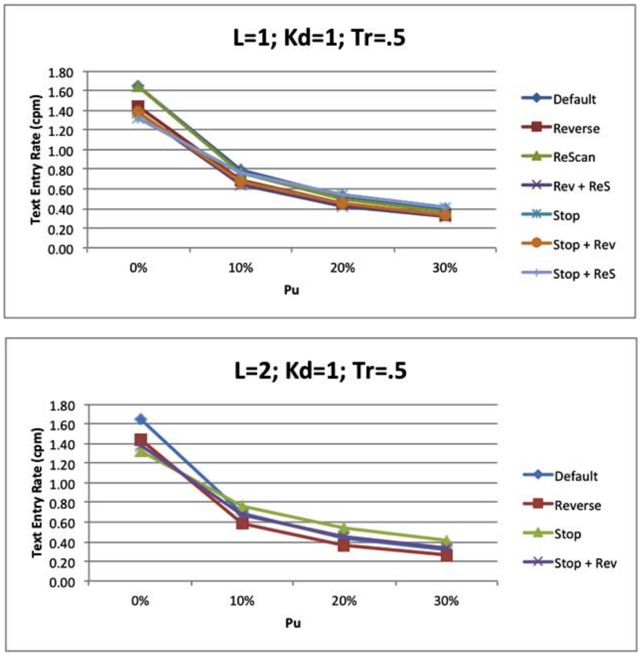


Fig. (13). Effect on Text Entry Rate (measured in characters per minute) of varying P_u, with 0.5 second recovery delay ($T_r = 0.5$; $P_n = P_e = P_l = 0$).

DISCUSSION

The stop scanning option was not expected to be useful when the likelihood of an error of omission (P_n) was high, but be expected to be useful when the likelihood of an error of commission (P_u, P_l or P_e) was high. As shown in Figs. (**10**, **11**), the conditions that included a stop scanning item did not produce the greatest TER as Pn increased. As shown in Figs. (**12-17**), a stop scan item only provided a distinct advantage when the loop count was two. Our expectations were met when errors of commission and frequent and the loop count was greater than 1. A reverse scanning item wasonly

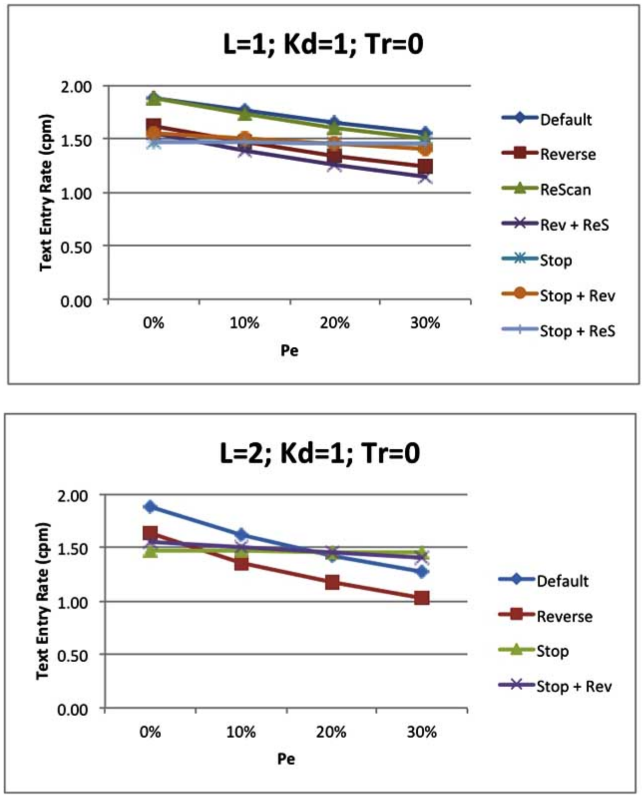


Fig. (14). Effect on Text Entry Rate (measured in characters per minute) of varying P_e, with no recovery delay ($T_r = 0$; $P_n = P_u = P_l = 0$).

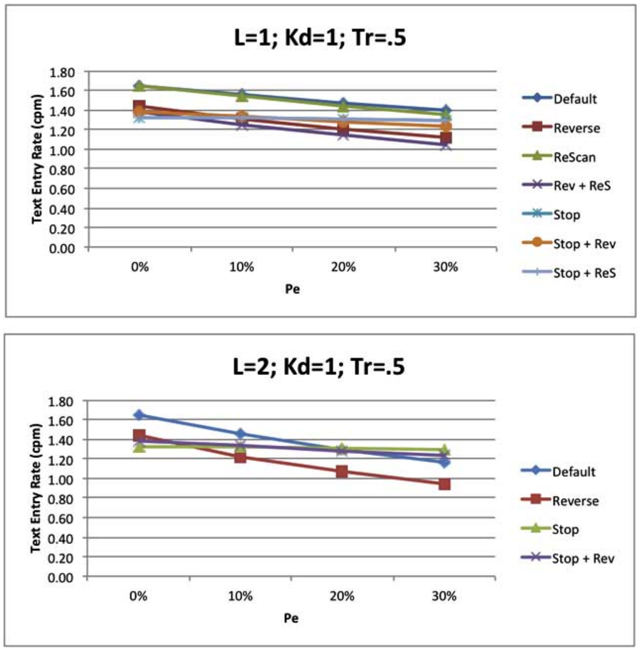


Fig. (15). Effect on Text Entry Rate (measured in characters per minute) of varying P_e, with 0.5 second recovery delay ($T_r = 0.5$; $P_n = P_u = P_l = 0$).

expected to be useful under such limited circumstances that it would provide no advantage, which was found to be true, and a rescan itemwas only expected to be helpful when the likelihood of an error of omission was high and the loop count was 1, was also found to be true.

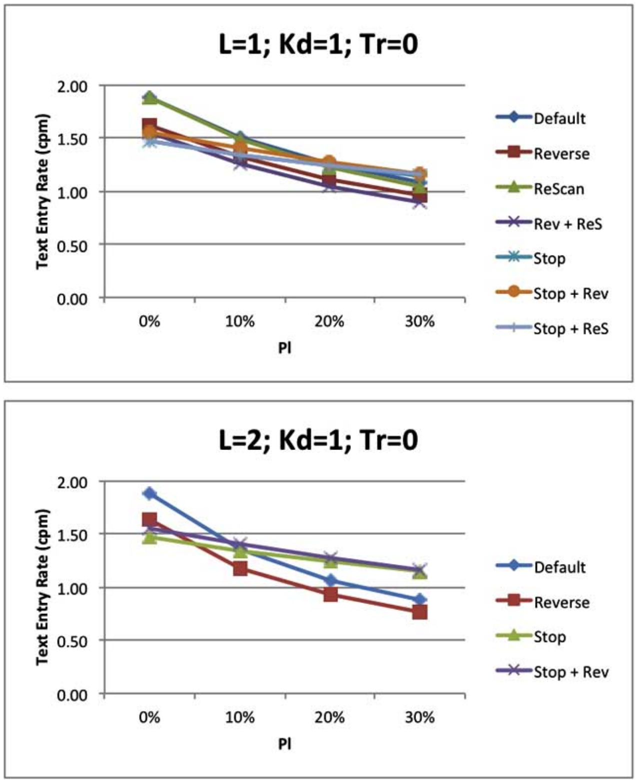


Fig. (16). Effect on Text Entry Rate (measured in characters per minute) of varying P_l, with no recovery delay ($T_r = 0$; $P_n = P_u = P_e = 0$).

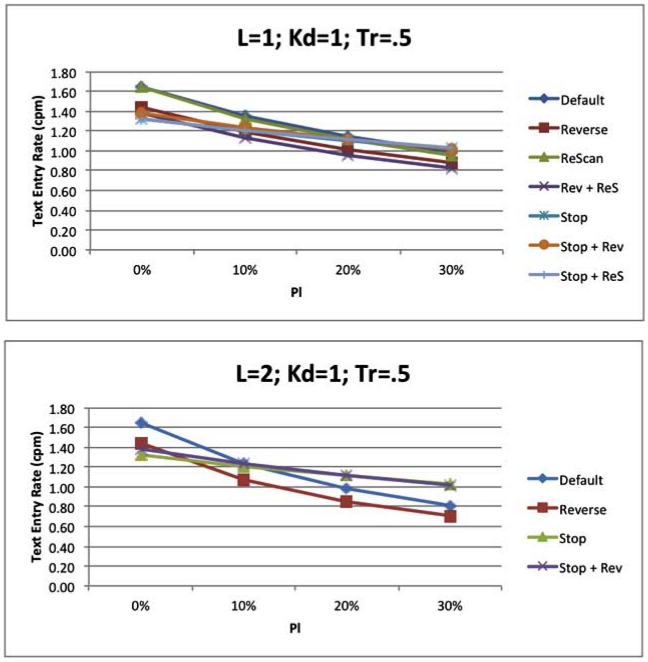


Fig. (17). Effect on Text Entry Rate (measured in characters per minute) of varying P_l, with 0.5 second recovery delay ($T_r = 0.5$; $P_n = P_u = P_e = 0$).

Of greater interest to clinicians is the observation that, even when error frequency was as high as 30%, there was very little difference between the best and worst configurations. In fact, the greatest difference observed between the highest and lowest TER within a single simulation was 0.53 characters per minute (cpm). The default configuration often produced the highest TER.

For all its complexity, there are still many settings and features of scanning interfaces that the model does not yet address:

- Abort/continue scanning method
    - Switch activated for extended time
- Where scanning starts after a selection is made
    - Row where last selection was made
    - Item where last selection was made
- Where scanning starts after column scanning is aborted
    - At the row where column scanning was aborted
    - At the row before column scanning was aborted
    - At the row after column scanning was aborted
- How scanning starts
    - Requires a switch press
- Scan pattern
    - Linear
    - Group-Row-Column
    - Quadrant
- Scan mode
    - Inverse
    - Step
- Repeating selections
    - Register repeat selections if the switch is held down
    - Time switch must be held to initiate repeating
    - Delay between repeated activations
- Types of matrix items
    - Character Prediction
    - Word Prediction
    - Items that open up a new scanning keyboard

The model also does not take cognitive load into consideration. Some scanning features (particularly reverse scan, character prediction and word prediction) require greater cognitive effort on the part of the user, which may lead to a decreased switch press time and increased errors. Anticipating and accommodating this type of effect in the model is also a topic of future work.

CONCLUSIONS

This model is the first to consider alternative error correction methods, and how error correction methods and the likelihood of different errors occurring can impact performance. When errors do occur, the error correction methods become increasingly valuable. The utility of each error correction method depends on the prevalence of different error types. When errors do not occur, there is no advantage to having error correction methods in the matrix. Similarly, when the matrix is arranged by frequency, there is no advantage to having a reverse scan item. These results suggest that the best approach for clinicians is to:

- use a frequency-arranged matrix
- avoid extra "bells and whistles" like stop scanning or reverse scanning items
- keep error rates as low as possible (i.e., focus clinical intervention on working to reduce the probability of errors occurring in the first place).

Our simulations focused on individual types of errors happening in isolation, but real users are likely to make a combination of errors. Our simulations also limited errors to no more than one error per character. Subsequent research (described elsewhere) used the model with real people to predict performance when a combination of errors can occur.

REFERENCES

[1] Koester H, Levine SP. Modeling the speed of text entry with a word prediction interface. IEEE Transact Rehab Eng 1994; 2(3): 177-87.

[2] Lesher GW. Acquisition of scanning skills: The use of an adpative scanning delay algorithm across four scanning displays., in RESNA 2002 Annual Conference, RESNA Press: Minneapolis, MN. 2002: 75-7.

[3] Lesher GW, Rinkus GJ. Leveraging word prediction to improve character prediction in a scanning configuration, in RESNA 2002 Annual Conference, RESNA Press: Minneapolis, MN, 2002.

[4] Simpson RC Koester HH. Adaptive One-Switch Row-Column Scanning. IEEE Trans Rehab Eng 1999; **7**: 464-73.

[5] Mankowski RE. Predicting Communication Rates: Efficacy of a Scanning Model, in Department of Rehabilitation Science and Technology, University of Pittsburgh: Pittsburgh, PA 2009.

[6] Simpson RC, Koester H, and LoPresti EF. Selecting an appropriate scan rate: The '.65 Rule'. Assis Technol 2007; 19(2): 51-60.

[7] Simpson RC, Koester HH, and LoPresti EF. Evaluation of an adaptive row/column scanning system. Technol Disability 2007; 18(3): 127-38.

[8] Angelo J. Comparison of three computer scanning modes as an interface method for persons with cerebral palsy. Am J Occup Ther 1992; 46(3): 217-22.

[9] Anson DK. Alternative Computer Access: A Guide to Selection 1st ed. Philadelphia, PA: F.A. Davis Company 1997; p. 280.

[10] Horstmann HM. Quantitative Modeling in Augmentative Communication -- A Case Study, in RESNA '90 Annual Conference. RESNA Press: Washington, DC 1990.

[11] Abascal J, Gardeazabal L and Garay-Vitoria N. Optimisation of the Selection Set Features for Scanning Text Input, in Computers Helping People with Special Needs, K. Miesenberger Editors, Springer-Verlag: Berlin 2004; pp. 788-95.

[12] Bhattacharya S, Basu A, and Samanta D. Computational modeling of user errors for the design of virtual scanning keyboards. IEEE Trans Neu Sys and Rehabil Eng 2008; 16(4): 400-9.

[13] Bhattacharya S, Samanta D, and Basu A. User errors on scanning keyboards: Empirical study, model and design principles. Interact Comput 2008; 20: 406-18.

[14] Bhattacharya S, Samanta D, and Basu A. Performance models for automatic evaluation of virtual scanning keyboards. IEEE Trans Neu Sys Rehabil Eng 2008; 16(5): pp. 510-9.

[15] Damper RI. Text composition by the physically disabled: A rate prediction model for scanning input. Appl Ergon 1984; 15(4): 289-96.

[16] Lesher GW, Moulton BJ, Higginbotham DJ. Techniques for augmenting scanning communication. Augm Alter Comm 1998; 14(2): 81-101.

[17] Venkatagiri HS. Efficient keyboard layouts for sequential access in augmentative and alternative communication. Augm Alter Comm 1999; 15(2): 126-34.

Received: January 29, 2011 Revised: March 8, 2011 Accepted: March 16, 2011