Effects of Optimizing the Scan-Path on Scanning Keyboards with QWERTY-Layout for English Text

Frode Eika SANDNES[a,b,1] and Fausto Orsi MEDOLA[c]
[a] *Oslo and Akershus University College of Applied Sciences, Oslo, Norway*
[b] *Westerdals Oslo School of Art Communication and Technology, Oslo, Norway*
[c] *UNESP, Bauru, Brazil*

Abstract. Scanning keyboards can be essential tools for individuals with reduced motor function. However, most research addresses layout optimization. Learning new layouts is time-consuming. This study explores the familiar QWERTY layout with alternative scanning paths intended for English text. The results show that carefully designed scan-paths can help QWERTY nearly match optimized layouts in performance.

Keywords. Scanning keyboard, text entry, reduced motor function, assistive technologies.

1. Introduction

Scanning keyboards are commonly used by individuals with reduced motor function. They can be operated with a single switch, unlike physical keyboards where users directly hit the key of the desired character or chording that relies on a smaller set of keys (Sandnes, 2006; Sandnes & Huang, 2006). Scanning input makes use of a virtual keyboard where the characters are highlighted in a certain sequence whereupon the user activates the switch when the desired character is highlighted.

Scanning keyboards trades off physical effort for waiting. That is, the bottleneck is the waiting time incurred by the selector to reach the desired character. In contrast, the waiting time can be reduced at the expense of more physical effort such as with single key tap-code based text input (Sandnes & Medola, 2016), Morse code (Levine, Gauger, Bowers, Khan, 1986) or multiple key menus (Sandnes, Thorkildssen, Arvei & Buverud, 2004).

Research into scanning keyboards has therefore focused on reducing the number of decisions and waiting times (scan lengths) per character. Most studies propose new keyboard layouts based on letter frequencies such that fewer steps are needed for the high frequency characters compared to low frequency characters (MacKenzie & Zhang, 1999).

Yet, several commercial and widely available scanning keyboards such as the On Screen Keyboard (OSK) that comes shipped with the Microsoft operating system employ the QWERTY layout. However, this design in particular is ineffective because it

[1] Corresponding Author.

completely resembles a physical keyboard and infrequent keys such as ESC, SHIFT, etc. are put into the most easy-to-reach positions. Moreover, the Microsoft OSK is a three-level hierarchal design where the user is cognitively burdened with making three decisions per character. The Microsoft design could be drastically improved with just a few simple adjustments.

It has been argued that the introduction of optimized and unfamiliar layouts is less problematic when a young individual learns to use such as system for the first time (Jones, 1998) as novice users have fewer expectations and set habits. However, this study assumes that users who already are familiar with QWERTY will usually find new keyboard layouts disruptive. Many sessions with hard effort are needed to learn new layouts (MacKenzie & Zhang, 1999). The goal is therefore to reuse the familiar QWERTY keyboard (Sandnes & Aubert, 2007) to improve acceptability of the technology and reduce the need for learning. This study thus set out to explore the effect the altering of the scanning path will have on both the overall waiting time and layout unfamiliarity of using the system with English text based on a theoretical methodology. The theoretical findings of this study may be useful for designing effective QWERTY-based scanning keyboards.

2. Background

The literature on scanning keyboards is vast and Polacek has provided an excellent survey with a taxonomy and performance comparisons (Polacek, Sporka & Slavik, 2017). Scanning keyboards are usually controlled using a switch, eye-blink, EEG-signal, joystick or a touch pad (Chiapparino, Stasolla, de Pace & Lancioni, 2011).

A typical scanning keyboard organizes characters into a grid, typically a square matrix. The matrix is then first traversed row by row, and then for a selected row, cell by cell, from left to right. Three-dimensional scanning keyboards have also been proposed (Felzer & Rinderknecht, 2009) and the Windows OSK can be classified as a three-dimensional design since the scans occur across three dimensions, that is, row, column group and cell. However, it is claimed that linear cursor paths are better than hierarchical cursor paths such as binary selection (Zhang, Fang & Francis, 2016). It has been pointed out that scanning keyboards may be preferred over other techniques since they rely on recognition over recall (Jones, 1998).

Several techniques are used in an attempt to improve the efficiency of scanning keyboards, such as go-back buttons at the end of each scanning row (Jones, 1998) where the scanning will return in the opposite direction if the user has missed a character, as opposed to starting from the beginning. In a study of mistakes made by users it was found that it was better to focus on input speed though optimal layouts instead of error correction mechanisms such as reverse scanning buttons, or stop buttons (Simpson, Mankowski & Koester, 2011).

The dwell time is a dominant factor affecting the performance of scanning keyboards. In the Windows OSK the delay can be controlled manually. Novices can therefore start with a slower delay and reduce this as the user gets more experience with the system. Several studies have also attempted to dynamically adjust this delay during text entry according to the user's performance (Simpson & Koester, 1999).

The optimization of keyboard layout has probably received the most attention. Typically, such optimizations start with a matrix of switch counts, and then the elements of the matrix are assigned characters according to the character frequencies (Lesher, Moulton & Higginbotham, 1998). Optimization techniques include decision tree analysis (Higger, Moghadamfalahi, Quivira & Erdogmus, 2016), mixed integer programming (Zhang, Fang & Francis, 2013), Huffman codes (Baljko & Tam, 2006) and inverse Huffman codes (Hamidi & Baljko, 2012). Common to these layouts is that they are information theoretic optimizations. However, these methods do not focus on human characteristics. This work instead fixes the keyboard layout and varies the scanning paths instead based on the assumption that users already have familiarity with the visual spatial layout of the QWERTY configuration.

Other optimizations include the use of letter prediction with n-grams (Jones, 1998), word prediction (Polacek, Sporka & Slavik, 2017), abbreviation expansion (Sandnes, 2015) and ambiguous keyboards requiring dictionaries (Mackenzie & Felzer, 2010; Miró-Borrás, Bernabeu-Soler, Llinares & Igual, 2009). The SAK keyboard comprised four virtual buttons. Experiments showed that only 1.713 scan steps were required per character (Mackenzie & Felzer, 2010). Both commercial scanning keyboards and research have focused on speeding up text entry with text prediction. However, it has been pointed out that word prediction is cognitively demanding compared to the input effort saved through the predictions (Koester & Levine, 1994) and that word prediction provides no practical benefit.

Several studies have discussed the key challenges in scanning keyboard research is the access to disabled individuals and relatively short test times (Simpson & Koester, 1999; Bhattacharya, Samanta & Basu, 2008). Especially with new design layouts longitudinal studies are more appropriate to study the learning effects. However, such longitudinal studies are time-consuming and straining on participants. Therefore, many researchers employ theoretic models to evaluate and compare scanning keyboard designs (Bhattacharya et al., 2008; Francis & Johnson, 2011; MacKenzie, 2012).

One performance measure is the number of steps needed to find a character, multiplied by the frequency of that character (Francis & Johnson, 2011), similar to the scan steps per character measure (MacKenzie, 2012). Moreover, the Manhattan (city-block) distance between characters in a layout that otherwise should be next to each other in a layout have been proposed as a measure (Francis & Johnson, 2011), for example if the QWERTY neighbors A and S are not next to each other the Manhattan distance between the two is used.

QWERTY has received little attention in the scanning keyboard literature. In one study, the top row QWERTYUIOP in the QWERTY layout was divided in two and put into the two first rows of a square matrix, namely QWERT and YUIOP. The other two QWERTY rows were mapped in a similar manner (Steriadis & Constantinou, 2003). Another simple layout has three groups where the three groups correspond to the top, middle and bottom row of the QWERTY layout, respectively (Bhattacharya et al., 2008).

This study departs somewhat from the claims and beliefs in the literature. For instance, Abascal, Gardeazabal and Garay (2004) recommended that layouts should be square and organized according to frequency of use. This study explores whether a familiar layout such as QWERTY can be improved by optimizing the scan path.

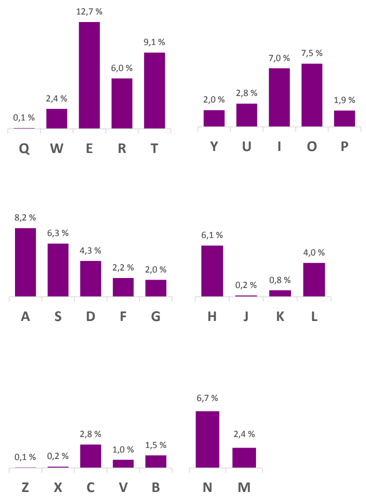

Figure 1. Character frequency on the QWERTY layout.

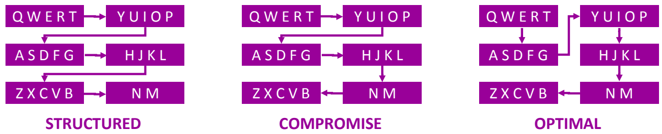

Figure 2. Group scanning paths.

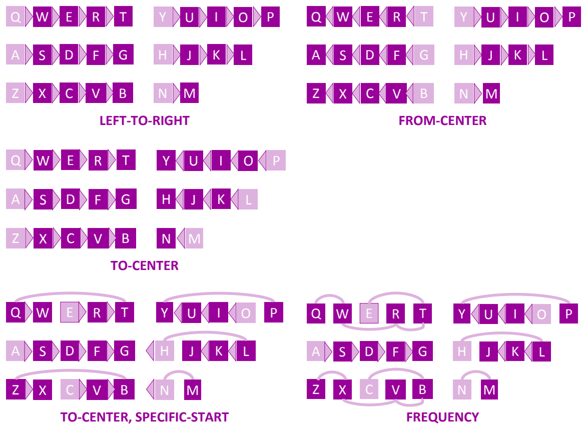

Figure 3. Cell scanning sequences. Bright keys represent scanning start points for the group.

3. Method

3.1. Group sequence

This theoretical study focuses on two-step scan paths where the user firsts select a character group, followed by selecting the specific character/cell within the group. We will therefore first discuss the group sequence, followed by the cell sequence. The QWERTY layout is divided into the left and right hands as used by the two hands. For each hand, the layout is divided into three groups, the top row, middle row and bottom row giving a total of six groups.

To analyze alternative scan paths the relative character frequencies are visualized as they appear on the QWERTY keyboard (see Figure 1). The figure shows that the top left group (QWERT) has the highest total frequency (sum of frequencies for the characters Q, W, E, R and T makes up 30.2% of the total), followed by the middle left group (ASDFG, 22.9%). Similar to this is the right group (YUIOP) with a total frequency of 21.1% followed by the right middle group (HJKL) with a frequency of 11.0%.

The least frequent groups are the bottom right (NM) group with a frequency of 9.2% and finally the bottom left group (ZXCVB) with a frequency of 5.5%. Note that these letter frequencies are based on US English texts.

Based on these observations, the optimal group scan path is QWERT → ASDFG → YUIOP → HJKL → NM → ZXCVB (see Figure 2 OPTIMAL). It is quite interesting that the groups more or less follow a continuous path. However, since the frequencies of the ASDFG and YUIOP groups are quite similar these two are swapped to make a simpler overall scan path (see Figure 2 COMPROMISE). The group scan path is therefore left-to-right for the two first rows and right-to-left for the last row.

In other words, a small reduction is scan path length is traded for a cognitively more intuitive order. This ordering is slightly different to the sequences presented in previous studies which follows more regular structures such as STRUCTURED in Figure 2.

3.2. Cell sequence

According to the literature, the most common cell sequence order is from left-to-right, which is probably due to the writing order in Western languages (Sandnes, 2008) (see Figure 3 LEFT-TO-RIGHT). In context of the QWERTY keyboard divided into three left and three right groups it is also natural to imagine groups scanned from the center and outwards (see Figure 3 FROM-CENTER). For the left groups this means a right-to-left scan direction while a left-to-right scan direction for the right groups.

Figure 1 shows that the frequencies decrease from left-to-right for the ASDFG and NM groups, while it mostly decrease from right-to-left for the YUIOP and HJKL groups. The ordering is less obvious for the QWERT and ZXCVB groups. It is therefore interesting to explore the effect of altering the scanning direction, namely to scan from both ends towards the center where the left side scanned from left-to-right, and the right side scanned from right-to-left (see Figure 3 TO-CENTER).

A further improvement that is explored herein is to use towards-the-center scanning direction described in the previous paragraph and starting with a specific high frequency character (see Figure 3 SPECIFIC-START). For the QWERTY row the left-to-right scans start with the E, meaning that the three most frequent characters are traversed first. By traversing the YUIOP group right-to-left starting with O the characters are traversed in decreasing order of frequency. The ASDFG is scanned by starting with A giving a perfect decrease in frequency. Moreover, a perfect decrease in frequency is achieved by starting the right-to-left scan of the HJKL group with H. The NM group is started with N and the ZXCVB group starts with C.

The final cell scan path explored herein is to traverse the characters within a group according to the decreasing character frequency (see Figure 3 FREQUENCY). The QWERT group is traversed as ETRWQ, YUIOP as OIUYP, ASDFG unchanged, HJKL as HLKJ, MN unchanged and ZXCVB as CBVXZ. This is attempted for the optimal group sequence and the optimized group sequence.

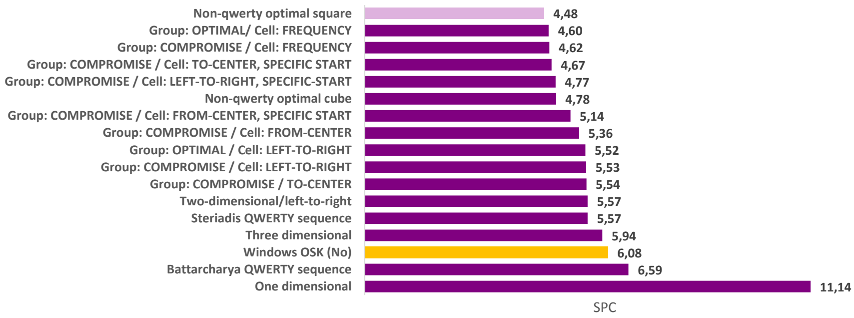

Figure 4. Scan steps per character.

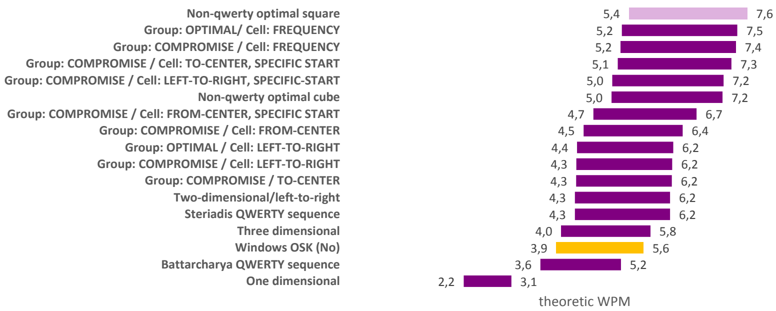

Figure 5. Theoretic words per minute (WPM) with dwell times in the range of 350 to 500 milliseconds.

4. Results

4.1. Scan steps per character

Figure 4 shows the designs discussed visualized in terms of their scan steps per character (SPC) which is defined as the sum of the character frequency and scan length products for each character divided by the number of characters. Clearly, simple one-dimensional scanning requires an average of 11.14 scan steps per character. Bhattacharya's simple three-row sequence is nearly a 50% improvement with 6.59 SPC.

The Microsoft Windows OSK Keyboard with Norwegian configuration gives a SPC of 6.08 (note that only the alphabetic characters are included in the statistic). On one hand, this is a remarkably high number considering it is a three level hierarchal configuration. The reason is that the keyboard also has many non-alphabetic keys in desirable low-distance locations. In comparison, the optimal cube layout yields a SPC of 4.78. Moreover, the Windows OSK is just slightly worse than a three level QWERTY layout with a 2×4 group configuration (5.94 KSP). One explanation for why the three-level configurations do not yield higher SPCs is that the benefit of three-level codes only becomes comparable with larger character sets (such as the Windows OSK). For smaller character sets, the simple two-level configurations are better as demonstrated by the results.

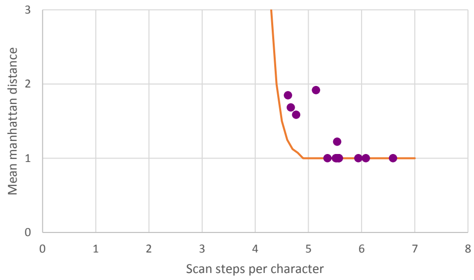

Figure 6. Trade-off between scan steps per character and mean Manhattan distance.

Next, there is a small improvement by optimizing the groups (left-to-right cell scanning) as the STRUCTURED (Two-dimensional/left-to-right) group sequence gives a SPC of 5.57, while swapping the order of the last two groups (COMPROMISE) gives a SPC of 5.53. The OPTIMAL group sequence gives a SPC of 5.52. Since the improvement achieved by first visiting ASDFG instead of YUIOP is so small that it may seem more sensible to follow the more regular order YUIOP → ASDFG.

Next, altering the cell scanning direction has a noticeable effect. An improvement is achieved by scanning FROM-CENTER of 5.36 SPC, which is better than scanning TO-CENTER (5.54 SPC). If scanning FROM-CENTER starting with a high frequency character one achieves a SPC of 5.14. However, an even better result is achieved scanning FROM-LEFT-TO-RIGHT starting with the most frequent character (4.77 SPC), or when scanning TO-CENTER starting with a high frequency character (4.67 SPC). The best results achievable with the QWERTY layout and the OPTIMAL group sequence, with cells in decreasing frequency order gives a SPC of 4.62.

In comparison, the optimal SPC achieved with a frequency optimized square layout is 4.48 SPC (4.45 is possible with a triangle). Thus, the best QWERTY design is only 3.1% less efficient than the frequency optimized layout or 2.6% less efficient than the QWERTY layout with the lowest SPC (OPTIMAL group sequence, and cell sequence).

4.2. Words per minute

Theoretical words per minute were computed (see Figure 5). These estimates and should not be used as a substitute for real observations, as actual words per minute will be lower. The estimates are based on dwell times of 350 and 500 milliseconds, since the dwell time is the dominant bottleneck in scanning-based text entry. The estimates were computed using $WPM = 60 / (5D \times SPC)$, where D is the dwell time and 5 represents the average English word length. Clearly, with the best QWERTY design it is theoretically possible to achieve text entry speeds of up to 7.4 words per minute. This is much better than what can for instance be achieved with the Windows OSK with a maximum theoretical speed of 5.6 words per minute (32.1% performance improvement).

4.3. SPC/MMD Trade-off

The results suggest that the more the keyboard layout or scan path is optimized in terms of scan steps per characters the more difficult they are to use. To explore this further, the mean Manhattan distance was used as a measure of layout unfamiliarity, inspired by (Francis & Johnson, 2011). The mean Manhattan distance (MMD) is defined as the sum of the frequency and Manhattan distance products for all the characters divided by the number of characters. The Manhattan distance between characters *a* and *b* on the layout is simply $d(a, b) = |a_{row} - b_{row}| + |a_{column} - b_{column}|$. It is assumed that the QWERTY layout is a familiar two-dimensional configuration and adjacent cursor visits in this space is given a Manhattan distance of 1. If the cursor jumps, the distance is larger. Note that it is also assumed that starting at the top row or with the left cell gives a Manhattan score of 1. Moreover, for the left set of rows, to start in the right side also gives no penalty as it is assumed that all scans originates from the center of the QWERTY keyboard.

Figure 6 plots the MMD against SPC for the QWERTY designs. An impressionistic boundary line is plotted to emphasize the trade-off. Note that this borderline tends towards 1 with many scan steps as it assumes that these designs use familiar layouts where the Manhattan distance is always 1. Clearly, for QWERTY designs with simple linear scan path motions the MMD is 1, that is, no penalty. However, the MMD becomes larger once the scan path optimizations become more complex. The more jumps in the scan path the larger the MMD penalty. Clearly, the design where each group is traversed in decreasing frequency order gives the shortest SPC of 4.61, but also one of the highest MMD of 1.8. The only design with a lower MMD of 1.9 is the one where the scanning direction is from the center and outwards starting with the most frequent character. The design with scans from left to right starting with the most frequent character gives the best balance of SPC of 4.76 and MMD of 1.58. If one optimize the scan path further it is expected that the MMD penalty will grow with small improvements in SPC.

5. Conclusions

This study has explored the QWERTY layout in terms of scanning keyboards. Several alternative scan paths have been explored using a theoretical methodology. The results shows that a very good trade-off between difficulty of use and performance can be achieved by dividing the characters of the qwerty keyboard into a 2×3 grid with 2 to 5 cells each, which should be traversed top to bottom and left to right for the two first rows and right to left for the last row. Moreover, cells should be traversed outwards going towards the center by starting with the most frequent character in each group. Results shows that when characters are traversed according to their respective frequency within each group the results are comparable to those of frequency-optimized layout. The QWERTY keyboard provides a familiar configuration for many users. The results are obtained using US English letter frequencies. Different languages may give different results; however, the same methodology can be applied. Future work needs to conduct measurements of the performance of actual users to confirm the theoretical predictions discussed herein. Such participants should be recruited among individuals with motor disabilities.

References

Abascal, J., Gardeazabal, L., & Garay, N. (2004). Optimisation of the selection set features for scanning text input. In *International Conference on Computers for Handicapped Persons* (pp. 788-795). Springer Berlin Heidelberg.

Baljko, M., & Tam, A. (2006). Indirect text entry using one or two keys. In *Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility* (pp. 18-25). ACM.

Bhattacharya, S., Samanta, D., & Basu, A. (2008). Performance models for automatic evaluation of virtual scanning keyboards. *IEEE Transactions on neural systems and rehabilitation engineering, 16*(5), 510-519.

Chiapparino, C., Stasolla, F., De Pace, C., & Lancioni, G. E. (2011). A touch pad and a scanning keyboard emulator to facilitate writing by a woman with extensive motor disability. *Life Span and Disability, 14*, 45-54.

Felzer, T., & Rinderknecht, S. (2009). 3dScan: an environment control system supporting persons with severe motor impairments. In *Proceedings of the 11th international ACM SIGACCESS conference on Computers and accessibility* (pp. 213-214). ACM.

Francis, G., & Johnson, E. (2011). Speed–accuracy tradeoffs in specialized keyboards. *International Journal of Human-Computer Studies, 69*(7), 526-538.

Hamidi, F., & Baljko, M. (2012). Reverse-engineering scanning keyboards. In *Proceedings of Computers Helping People with Special Needs* (pp. 315-322), Springer Berlin Heidelberg.

Higger, M., Moghadamfalahi, M., Quivira, F., & Erdogmus, D. (2016). Fast Switch Scanning Keyboards: Minimal Expected Query Decision Trees. *arXiv preprint arXiv*:1606.02552.

Jones, P. E. (1998). Virtual keyboard with scanning and augmented by prediction. In *Proceedings of European Conference series on Disability, Virtual Reality and Associated Technologies* (pp. 45-51).

Koester, H. H., & Levine, S. P. (1994). Learning and performance of able-bodied individuals using scanning systems with and without word prediction. *Assistive Technology, 6*(1), 42-53.

Lesher, G., Moulton, B., & Higginbotham, D. J. (1998). Techniques for augmenting scanning communication. *Augmentative and Alternative Communication, 14*(2), 81-101.

Levine, S., Gauger, J., Bowers, L., & Khan, K. (1986). A comparison of Mouthstick and Morse code text inputs. *Augmentative and Alternative Communication, 2*(2), 51-55.

MacKenzie, I. S., & Zhang, S. X. (1999). The design and evaluation of a high-performance soft keyboard. In *Proceedings of the SIGCHI conference on Human Factors in Computing Systems* (pp. 25-31). ACM.

Mackenzie, I. S., & Felzer, T. (2010). SAK: Scanning ambiguous keyboard for efficient one-key text entry. *ACM Transactions on Computer-Human Interaction (TOCHI), 17*(3), 11.

MacKenzie, I. S. (2012). Modeling text input for single-switch scanning. In *International Conference on Computers helping people with special needs* (pp. 423-430). Springer Berlin Heidelberg.

Miró-Borrás, J., Bernabeu-Soler, P., Llinares, R., & Igual, J. (2009). Ambiguous keyboards and scanning: The relevance of the cell selection phase. In *Proceedings of Human-Computer Interaction–INTERACT 2009*.

Polacek, O., Sporka, A. J., & Slavik, P. (2015). Text input for motor-impaired people. *Universal Access in the Information Society, 16*, 51-72.

Sandnes, F. E., Thorkildssen, H. W., Arvei, A., & Buverad, J. O. (2004). Techniques for fast and easy mobile text-entry with three-keys. In *Proceedings of the 37th Annual Hawaii International Conference on System Sciences*. IEEE.

Sandnes, F. E. (2006). Can spatial mnemonics accelerate the learning of text input chords?. In *Proceedings of the working conference on Advanced visual interfaces* (pp. 245-249). ACM.

Sandnes, F. E., & Huang, Y. P. (2006). Chording with spatial mnemonics: automatic error correction for eyes-free text entry. *Journal of information science and engineering*, *22*(5), 1015-1031.

Sandnes, F. E., & Aubert, A. (2007). Bimanual text entry using game controllers: Relying on users' spatial familiarity with QWERTY. *Interacting with Computers, 19*(2), 140-150.

Sandnes, F. E. (2008). Directional bias in scrolling tasks: A study of users' scrolling behaviour using a mobile text-entry strategy. *Behaviour & Information Technology, 27*(5), 387-393.

Sandnes, F. E. (2015). Reflective text entry: a simple low effort predictive input method based on flexible abbreviations*. Procedia Computer Science, 67*, 105-112.

Sandnes, F. E., & Medola, F. O. (2016). Exploring Russian Tap-Code Text Entry Adaptions for Users with Reduced Target Hitting Accuracy. In Proceedings of the 7th International Conference on Software Development and Technologies for Enhancing Accessibility and Fighting Info-exclusion (pp. 33-38). ACM.

Simpson, R. C., & Koester, H. H. (1999). Adaptive one-switch row-column scanning. *IEEE Transactions on Rehabilitation Engineering, 7*(4), 464-473.

Simpson, R.C., Mankowski, R., & Koester, H.H. (2011). Modeling one-switch row-column scanning with errors and error correction methods. *The Open Rehabilitation Journal, 4***,** 1-12.

Steriadis, C. E., & Constantinou, P. (2003). Designing human-computer interfaces for quadriplegic people. *ACM Transactions on Computer-Human Interaction (TOCHI), 10*(2), 87-118.

Zhang, X. C., Fang, K., & Francis, G. (2013). Optimization of switch keyboards. In *Proceedings of the 15th International ACM SIGACCESS Conference on Computers and Accessibility* (p. 60). ACM.

Zhang, X., Fang, K., & Francis, G. (2016). How to optimize switch virtual keyboards to trade off speed and accuracy. *Cognitive Research: Principles and Implications, 1*(1), 6.