Hindawi
Advances in Human-Computer Interaction
Volume 2021, Article ID 6657155, 14 pages
https://doi.org/10.1155/2021/6657155

Research Article

Usability Evaluation of Optimized Single-Pointer Arabic Keyboards Using Eye Tracking

A. Benabid Najjar,[1] **A. Al-Wabil**,[2] **M. Hosny**,[3] **W. Alrashed**,[1] **and A. Alrubaian**[3]

[1]*Software Engineering Department, King Saud University, Riyadh, Saudi Arabia*
[2]*Software Engineering Department, Alfaisal University, Riyadh, Saudi Arabia*
[3]*Computer Science Department, King Saud University, Riyadh, Saudi Arabia*

Correspondence should be addressed to A. Benabid Najjar; abbenabid@ksu.edu.sa

Received 10 November 2020; Revised 3 February 2021; Accepted 10 February 2021; Published 20 March 2021

Academic Editor: Antonio Piccinno

This paper presents the design and usability evaluation of an Arabic keyboard for applications that predominantly use single-pointer input device. Such applications are particularly used in mobile devices like Portable Data Assistants (PDAs) and smartphones. They are also valuable in gaze-controlled interfaces that constitute a growing mode of communication and that particularly empower people with mobility impairments. A special focus is given to the optimization of the key arrangement based on the movement time and character transition frequencies. An optimization model as well as a Simulated Annealing algorithm are presented. Then, the performance of the optimized layout is assessed showing that it outperforms the commonly used Arabic keyboard in terms of the estimated typing speed. However, the main limitation that the new layout might face is that a new arrangement of keys may not be adopted by users, even if the currently used layouts are not optimum. Therefore, a usability evaluation of the optimized layouts was conducted using eye-tracking and task-based testing involving the end-users and considering both objective and subjective measures of usability. Implications for the design are also discussed.

1. Introduction

Nowadays, with virtual keyboards, a simple program can easily switch from the traditional keyboard layout into a new one and back again. This enables the user to easily test and adopt new layouts on the same keyboard. Virtual keyboards are widely used particularly in handheld mobile devices such as smartphones and PDAs. In these devices, usually only one pointer (finger or stylus) is used for data input. These one-pointer-based text entry keyboards are increasingly used in writing long messages, emails, and even text documents.

Furthermore, for people with severe disabilities, typing a text with traditional keyboards is often a difficult task or in some cases could even be impossible. Using alternate text input has been made possible by recent developments of gaze-controlled applications. These applications involve an eye-tracking device that detects eye movements as well as an on-screen keyboard. The selection is computed within a predefined dwell time that corresponds to a prolonged gaze fixation. Majaranta [1] provides an extensive review of the original research studies conducted in the area of gaze-based text entry. In 2014, Microsoft patented eye-tracking keyboard software that could be used on a number of devices. Then, 3 years later, Windows 10 included built-in eye-tracking support and an eye-control module that can be used with a compatible eye tracker to operate a virtual mouse, keyboard, and text-to-speech experience using only eyes movement [2].

Designing a keyboard where only one pointer is used for text input, either for severely disabled people using gaze-controlled typing or people using handheld mobile devices, is a prospering research area. Several research studies [3–6] were conducted to enhance the interface design of virtual keyboards in order to improve the comfort and typing speed of the end-users. In fact, even a small modification in the key arrangement may lead to a significant enhancement in

typing performance since the improvement is accumulated in a repetitive task like text typing.

However, the research on the optimization of the keyboard layout considering non-Latin languages is relatively scarce. Arabic is one of the most populous languages in the world and is widely taught in schools and universities and used in workplaces and media. It is written with the Arabic alphabet, which has a particular script and is written from right to left. However, few research works [7, 8] considered the problem of optimizing the Arabic keyboard for one-pointer use that is adapted for mobile devices or even needed by people with physical disabilities.

On the other hand, despite the considerable effort conducted to enhance the performance of typing through optimizing the keys distribution, the optimized layouts are not widely adopted which might be due not only to the reluctance of people to change but also to the lack of empirical research and user research to support the adoption of optimized layouts. In fact, the optimized layouts reported in the literature were tested only from a mathematical point of view, and, to the best of our knowledge, there is no up-to-date study about the end-users' feedback and satisfaction. Therefore, evaluating the usability is of great importance to ensure the effectiveness and user satisfaction of the interface design. It reflects the significant effect of users' human behavior on their experiences with new applications, through measuring task performance indicators such as the task execution time, the number of errors, and the users satisfaction measurement while using an application.

This paper tackles the problem of designing and evaluating the usability of an optimized Arabic keyboard for applications that predominantly use single-pointer input device (i.e., finger, stylus, and eye). A special focus was given to investigating the impact of the minimization of the estimated movement time on the actual typing speed with real users. The main contributions can be summarized as follows:

(i) The optimization of keys arrangement according to the transition frequencies of characters in the Arabic language is examined.

(ii) A two-phase Simulated Annealing algorithm is proposed and proven to generate optimized layouts that outperform the commonly used Arabic keyboards in terms of estimated movement time.

(iii) An eye-tracking virtual keyboard embedding different layouts is implemented; Tobii X120 eye tracker is used along with different interfaces implementing the commonly used as well as the newly optimized Arabic keyboards for gaze typing.

(iv) A usability evaluation is conducted to examine the usability of different layouts in terms of effectiveness, efficiency, and user satisfaction. Performance and observational studies are discussed, and future research directions are presented.

This paper is organized as follows: Next, Section 2 summarizes the related work with special focus on one-pointer optimized keyboards. Then, Section 3 presents the optimization model and the Simulated Annealing algorithm to generate the optimized layouts. Finally, Section 4 presents the usability evaluation framework and discusses the corresponding results.

2. Related Work

Eggers et al. [3] and Anon [9] showed that an effective interface design of keyboards is critical for the end-users performance and can reduce health pains including eye fatigue, stress, and strain. It even may affect user productivity and health due to the repetitiveness in text entry related work. The standard layout (QWERTY) is the most widely used keyboard layout. However, it was initially introduced to reduce the number of key clashes in the mechanical type basket and is not optimized for efficiency. Then, many text input keyboards were developed in the last years. For example, an optimized keyboard layout, known as the Dvorak keyboard [10], was designed based on the frequencies of the different letters and was demonstrated to be more efficient than the traditional one. Wagner et al. [4] and Eggers et al. [3] proposed a keyboard design to improve typing performance as well as ergonomic criteria including the load on fingers, number of key hits, hand alternation, avoidance of key hits by the same finger, avoiding large steps, and a key hit direction from little finger to thumb.

Although these keyboard layouts outperform the QWERTY layout, this last one remains the most widely used due to the reluctance of many people to change. However, with virtual keyboards, a simple program can easily switch from the traditional layout to a new one and back again. This enables the user to easily test and adopt new layouts on the same keyboard. Furthermore, a dynamic keyboard, Optimus Maximus keyboard [11], was first introduced in 2007 presenting each key as a stand-alone display that can dynamically change and shows the function currently assigned to it. This customizable layout allows users to easily change from English to other languages particularly those with a different alphabet like Greek, Arabic, or Chinese. This idea has been recently renewed by Apple that has patented a new keyboard featuring tiny screens on each key [12]. This new keyboard would open doors for easily integrating and adopting newly designed and optimized keyboard layouts.

2.1. Gaze-Based Text Entry. Polacek et al. [14] provided a review, based on 150 publications, discussing current state-of-the-art about accessible text entry for motor-impaired people. It discussed the common techniques of text entry including selection of keys, approaches to character layouts, use of language models, and interaction modalities. Particularly, text entry by eye gaze is used by people with severe motion impairment. An eye-tracking device follows the user's eye movements, and a computer program analyzes the gaze behavior. To type by gaze, the user typically points at the characters on an on-screen keyboard by looking at them and selects them by means of dwell time. For severely disabled people, dwell time is often the best and the only means of selection. Dwell means a prolonged gaze fixation: the user needs to fixate on the key for longer than a

predefined threshold time (typically, 500–1000 ms) in order for the key to be selected. Sarcar et al. [14] proposed a gaze-based text entry system, EyeK, that aimed to reduce the dwell time and to mitigate visual search time. The experiments showed that the proposed interface achieved higher text entry rate over the existing interfaces. Recently, Sandnes et al. [15] proposed to reduce scanning keyboard input errors by introducing longer dwell time for the first element in scan sequences. They explored several designs and evaluated their effect on overall text entry performance. While traditional gaze-based virtual keyboards were assessed for a single language (usually English), Cecotti et al. [16] proposed a multiscript gaze-based assistive virtual keyboard, where it is possible to change the layout of the graphical user interface in relation to the script so that the keyboard can be accessed by people who communicate with Latin, Bangla, and/or Devanagari scripts.

As for Arabic language, iWriter was introduced by Al-Wabil et al. [17] as an Arabic software keyboard that was designed for use by people with physical disabilities using gaze-controlled text typing. However, eye typing using this keyboard can be very slow since it depends on the dwell time threshold that sets a limit on the maximum typing speed. In practice, typing speed is even much more reduced since people need time for cognitive processing, to think what to type next, to search for the next key on the keyboard, etc. Actually, the layout used in the iWriter system was not optimized for typing text with only one pointer. The hypothesis, in our study, states that an optimized layout in which the letters are organized according to the transition frequency of characters would minimize the distance between the most frequent pairs of characters, reduce the eye movements distance traveled for a given text, as well as the time needed to find the next key, and then increase the overall typing speed.

2.2. Single-Finger Text Entry. For single-finger or stylus-based text entry, several different keyboards were proposed in the literature: Hooke, Metropolis, Lewis, OPTI, and FITALY [18] keyboards attempted to minimize distances of the commonly associated pairs of characters in the English language. Then, Li et al. [19] proposed to model the problem using an integer programming (IP) model considering the character transition frequency of words in the English language and different shapes of keyboard. The results showed that the proposed layouts minimized the finger movement and outperformed the existing English keyboards. They tackled this problem using two models: (1) an IP model, where the transition is fixed between any two keys, and (2) a two-stage heuristic, where the first stage uses a swap neighborhood move to obtain a new solution while the second stage applies Simulated Annealing to enhance the solution.

Yin and Su [20] proposed to tackle this problem using Particle Swarm Optimization (PSO) and considering multiple objectives and motor-impaired users. Experimental results considering different shapes and different layouts showed that the Cyber Swarm keyboard outperformed several benchmark keyboards. Moreover, the proposed algorithm performed better than other competing algorithms including random, heuristic, and standard PSO. A usability evaluation involving six participants, traditional QWERTY users, showed that the Cyber Swarm keyboard can be quickly and effectively learned and that the typing rate almost doubled after some practice.

Murali and Panicker [21] proposed a Genetic Algorithm (GA) to solve this problem and used the TOPSIS (Technique for Order of Preference by Similarity to Ideal Solution) to rate the best layouts obtained by the GA. Three attributes were considered to assess the best generated keyboards, namely, the flow distance, the average words per minute, and the learning percentage criteria obtained.

Recently, Pradeepmon et al. [6] tackled the same problem and proposed a layout for the single-finger keyboard that was optimized using a variant of Genetic Algorithm, namely, the Estimation of Distribution Algorithm. The suggested layout was found to be efficient in terms of rapid typing compared with some of the existing keyboard layouts.

All the previous studies focused on optimizing the keyboard for English language. However, Wolosik and Tabedzki [5] proposed to solve this problem for Polish language and estimated that the proposed arrangement would shorten the time to input sample texts by about 30%.

For multiple languages, Dell'Amico et al. [22] considered the problem of minimizing the average time needed to write a text for different languages: English, French, Italian, and Spanish. They considered a generalization of the problem by considering more locations in the keyboards than the symbols to be assigned. Then, the layout was not predefined. They solved this problem using different metaheuristics including Local Search, Simulated Annealing, Tabu Search, Variable Neighborhood Search, and Fast ANT (FANT).

Recently, Herthel and Subramanian [23] also tackled the problem of designing optimized single-finger keyboard layouts on smartphones. They extended the line of research introduced by [22] in modeling the problem and in considering benchmark instances for English, French, Italian, and Spanish, to which they added benchmark instances for Portuguese and Bilingual variants (combining English with other languages). To solve this problem, they proposed a Local Search-based metaheuristic which is composed of three neighborhood structures. The results of their experiments were particularly competitive in terms of both solution quality and CPU time and showed the potential practical benefits of adopting optimized single-finger layouts, when compared to well-known layouts, namely, QWERTY and AZERTY.

2.3. Optimized Layouts for Arabic. For the Arabic keyboard, the commonly used layout is derived from the Arabic typewriter's layout and is not optimized for performance. Malas et al. [24] and Khorshid et al. [25] proposed to optimize the design of the Arabic keyboard layout for typing speed and ergonomic criteria. Then, an Ergonomic Arabic Keyboard, proposed by Osman [26], was patented in 2012. It

was designed to reduce strain and avoid long-term musculoskeletal injuries to the forearm, wrist, and hand. The design was based on the actual frequency of use of Arabic characters. All these research studies proposed to consider the key distances and frequencies of Arabic characters while trying to distribute the typing effort among the ten fingers taking into consideration the finger used and hand alternation.

For Arabic single-pointer keyboards, the currently available systems for mobile devices still use the Arabic typewriter's layout. For example, iWriter system [17] embeds an Arabic virtual keyboard using gaze-controlled text typing, but the system's developers highlighted the importance of designing an optimized Arabic keyboard layout adapted for such applications in order to improve as much as possible the text entry performance since (1) eye typing can be very slow due to the dwell time threshold that sets a limit on the maximum typing speed, and (2) it can also lead to eye fatigue and pain when involved in long time computer related work due to the repetitiveness of eye and head movements.

The first attempt was in a previous work, conducted by Benabid Najjar [7], where a Simulated Annealing algorithm was developed to optimize the Arabic keyboard design for single-pointer applications. The design aimed to maximize the typing speed by minimizing the distance between the most frequent pairs of Arabic letters. Then, Alswaidan et al. [8] tackled the problem of optimizing the single-finger Arabic keyboard using a Genetic Algorithm based on three measures in the objective function: the distance between pairs of letters, a weight for each row in the keyboard, and the hit direction of the finger. The performance of the proposed layouts was assessed by virtually estimating the speed of typing, using the distance between letters in a given text. The results showed a noticeable improvement in terms of the measured typing speed of the optimized layouts over multifinger and conventional single-finger keyboards.

2.4. Summary and Discussion. Table 1 summarizes some of the existing studies on the optimization of the single-pointer keyboard. As discussed above, it can be noticed that the majority of the studies focused on optimizing the keyboard for the English language [6, 19–22]. For the Arabic language, the majority of the research was dedicated to multifinger keyboard optimization [25–27].

Moreover, it can be noticed from Table 1 that the main objective, for the single-pointer keyboard optimization, was to maximize the typing speed through minimizing the movement time, based on Fitts' law [27], between the keys that are assigned to the frequently associated pairs of letters. Therefore, almost all the studies have modeled this problem in terms of the famous Quadratic Assignment Problem (QAP).

However, it is worth mentioning that [22, 23] considered a generalization of the QAP, namely, SK-QAP, in which the shape of the layout was not predefined.

Different algorithms were proposed to solve this problem. The main used algorithms are based on Local Search, Simulated Annealing, and evolutionary algorithms such as Genetic Algorithm. It should be noted that, in the literature [28], Local Search approaches were found to be more effective for QAP compared to the search methods based on solution construction such as PSO. However, [29, 30] investigated the applicability of PSO on the QAP, and the results were promising. Particularly, [20] proposed a PSO-based algorithm for multiobjective optimization of the single-pointer keyboard, which opens the door for further exploration of this research direction.

Finally, an important observation that can be easily derived from Table 1 is the lack of user research studies to assess the usability of the optimized layouts. While this is an important step toward the validation of the optimized keyboards, it is clear that the research in this direction is still scarce. Therefore, this paper focuses on investigating users' behavior, their performance while using the newly generated layouts, and their acceptability of the change.

3. Optimization Model

This section presents the optimization model proposed in this study as well as the Simulated Annealing algorithm that was used to generate the optimized layout that minimizes the movement time based on the character transition frequencies.

3.1. Character Transition Frequency. In order to enhance the typing speed, the idea is to minimize the distances between keys assigned to the frequently associated pairs of characters in the Arabic language, by positioning the characters, with relatively high transition frequency, close to each other, in order to avoid large steps between them and thus minimize the overall movement time for a given text. The transition frequencies between the different pairs of characters, as computed in [8] by parsing Arabic Wikipedia articles from different fields, are considered. These articles cover subjects across almost all disciplines, and thus, the vocabulary is not reduced to a specific domain. It can be easily noticed that the pair "Alef" and "Lam" has the highest frequency as expected since this particular pair is the definition article in the Arabic language that is equivalent to the article "The" in the English language.

3.2. Movement Time. Typing speed, low error rate, and comfort in use are different criteria that may reflect the performance of a keyboard's layout. A special focus is given to the typing speed that can be derived from the movement time. In fact, the movement time is a measurable characteristic, and thus can be computed to evaluate the layouts of the keyboard. The movement time from one key to another may be estimated proportionally to the Euclidean distance between these two keys. It is usually measured using Fitts' law, introduced in [27], that states that the time and the difficulty required to type consecutively two symbols i and j can be estimated as follows:

Table 1: Summary of some of the existing studies on the optimization of single-pointer keyboard.

Ref.	Year	Language	Optimization		Usability evaluation		
			Algo.	Objective(s)	Participants	Task(s)	Criteria
[19]	2006	English	SA	Movement time		None	
[20]	2011	English	CSA, PSO	Key accessibility, posture comfort, keystrokes, word clashes	6 participants, familiar with QWERTY	Typing an article (<500 words) 5 times	Participants feedback
[21]	2016	English	GA + TOPSIS	Flow distance, learning percentage, typing speed		None	
[6]	2018	English	2 EDAs: UMDA, PBILA.	Movement time	2 participants: one familiar with QWERTY, one 10-year-old child	Typing a small poem	Typing time
[8]	2014	Arabic	GA, SA	Distance between letters, hit direction, row weights		None	
[5]	2016	Polish	GA, SA	Distance between letters, hit direction, row weights		None	
[22]	2009	English, French, Italian, Spanish	LS, SA, TS, VNS, FANT	Movement time		None	
[23]	2020	English, French, Italian, Spanish, Portuguese, Bilingual	ILS	Movement time		None	
This paper	2021	Arabic	SA	Movement time and hit direction	12 participants, 18–33 years old, familiar with Arabic keyboard	Typing 4 sentences using 3 different layouts	Eye-tracking metrics, effectiveness, efficiency, and satisfaction

SA: Simulated Annealing; GA: Genetic Algorithm; CSA: Cyber Swarm Algorithm; PSO: Particle Swarm Optimization; TOPSIS: Technique for Order of Preference by Similarity to Ideal Solution; EDA: Estimation of Distribution Algorithm; UMDA: Univariate Marginal Distribution Algorithm; PBILA: Population-Based Incremental Learning Algorithm; LS: Local Search; TS: Tabu Search; VNS: Variable Neighborhood Search; FANT: Fast ANT; ILS: Iterated Local Search.

$$T_{ij} = \alpha + \beta \cdot \log_2\left(\frac{d_{ij}}{W_j} + 1\right), \quad (1)$$

where the parameters α and β are constants, d_{ij} is the distance between the two keys assigned to i and j, and W_j is the width of the target key assigned to j. The constant α may be omitted since its contribution to the overall typing time is independent of the arrangement of the keys. The constant value β has been experimentally determined by [31] to be equal to 1/4.9 bits/sec in the special case of using a stylus for data entry. Moreover, the keys are assumed to have equal widths, which imply that W_j can be considered equal to 1, and then the movement time depends only on the distance between pairs of keys.

3.3. Keyboard Shapes. In FITALY and OPTI keyboards, which were designed for single-finger pointing, the characters were arranged in a 5×6 matrix. This configuration seems to be more adapted for one-pointer use since it is closer to a square and thus reduces the maximum movement distance. However, it is not the most suitable layout to be displayed on a computer's screen or on a mobile device in the landscape mode.

The common keyboard layout is the rectangular one with three main rows. Keyboards such as QWERTY and Dvorak consider a 10×3 matrix to handle the 26 characters plus few symbols. The Arabic keyboards consider a 12×3 matrix to represent the 34 characters. However, the Arabic layout used in mobile devices consider an 11×3 (Android devices) or 10×3 (iPhone devices) layouts including the most frequently used characters as well as a Delete key. Accordingly, in order to design a virtual keyboard that would be properly displayed on a screen and in order to be able to compare the optimized layout to the existing layouts, a rectangular shape is considered in this study. Finally, the adopted shape is an 11×3 matrix including the 32 most frequent Arabic letters.

3.4. Problem Formulation. For several years, the problem of keyboard design has been addressed by researchers in the ergonomics domain before being formulated as a combinatorial optimization problem. The keyboard's layout optimization problem aims to find an optimal arrangement of a given number of symbols and thus can be modeled in terms of the famous Quadratic Assignment Problem (QAP) as defined by [19]. The QAP involves assigning n facilities to n locations. Two measures are used in the objective function:

the first measure is the flow c_{ij} between the facilities i and j. The second measure is the distance d_{kl} between location k and location l. When a facility i is assigned to location $p(i)$ and facility j is assigned to location $p(j)$, the cost of this assignment is calculated as $c_{ij} \times d_{p(i)p(j)}$. Then, the total cost of assignment of all facilities can be defined as

$$\text{cost} = \min \sum_{i=1}^{n} \sum_{j=1}^{n} c_{ij} \times d_{p(i)p(j)}. \quad (2)$$

The flows simply represent the characters' transition frequencies in Arabic language. The distance matrix represents the movement times needed to type consecutive characters. The objective function reflects the performance of a given keyboard arrangement and thus would be used to evaluate the different keyboard layouts.

Alswaidan et al. [18] proposed to consider the frequency of pairs of letters as distance between the letters in the Arabic layout. Moreover, the hit direction was considered by adding a penalty when the pointer moves from left to right in order to maintain the direction of Arabic writing from right to left. Furthermore, the most frequent letters were placed in the middle row, to be more convenient for the user, while the less frequent ones were placed in the upper or bottom rows. Therefore, the overall score (objective) of the solution was defined by

$$F = \min a \sum_{i=1}^{n} \sum_{j=1}^{n} D_{ij} + b \sum_{i=1}^{n} \sum_{j=1}^{n} S_{ij} + c \sum_{i=1}^{n} R_i, \quad (3)$$

where a, b, and c are weights that determine the importance of each term; D_{ij} is the distance between a pair of letters; S_{ij} indicates the hit direction; and R_i corresponds to the row i in the layout.

In this paper, the proposed design aims to maximize the typing speed by minimizing the distance between the most frequent pairs of Arabic letters. The transition frequency is used for this purpose, and the movement time is computed using Fitts' law. Moreover, in a similar way to [8], a penalty is added to reflect the hit direction so that the distance between the pairs is considered lower when the pointer moves from right to left than when it moves from left to right. Therefore, the objective function can be defined as follows:

$$S = \min \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} \times T_{ij} = \frac{1}{4.9} \min \sum_{i=1}^{n} \sum_{j=1}^{n} f_{ij} \times \log_2\left(\frac{d_{ij}}{W_j} + 1\right), \quad (4)$$

where f_{ij} is the frequency between a pair of letters and d_{ij} is the distance between them including the hit direction penalty.

3.5. Optimization Algorithm. The QAP is known to be an NP-hard problem as shown in [32]. Consequently, there is no known algorithm that solves this problem in polynomial time, and the exact approaches are limited to small instances of the problem. Therefore, heuristic and metaheuristic approaches are usually used for the QAP, rather than exact solution methods. The goal of such approaches is to find a relatively good solution in a reasonable amount of time, rather than the optimal one. A survey of several metaheuristic approaches for solving QAP can be found in [33].

To tackle this problem, Alswaidan et al. [8] proposed a Genetic Algorithm (GA) approach for optimizing a single-finger Arabic keyboard layout, whereas this paper extends the research line introduced in [7] and presents an algorithm based on Simulated Annealing (SA). Simulated Annealing (SA) is originally inspired by a process used in metallurgy that alternates cycles of heating and slow cooling (annealing), which tend to minimize the energy of the physical system. It is used in optimization to help a Local Search procedure to escape from getting stuck in local minimum. To simulate the evolution of a physical system to its thermodynamic equilibrium at a given temperature T, Metropolis algorithm [34] can be used, starting from a given configuration (in our case, an initial random layout), and then a Local Search explores the neighboring solutions.

SA allows accepting solutions with greater objective functions than the current solution according to a probability function and thus enables exploring different regions in the solution space. Figure 1 represents the proposed algorithm which includes two main stages: The first stage aims to compute the initial solution. It starts by a greedy search trying to find a better solution by randomly exchanging the position of any two characters until an improved layout cannot be found in 10 consecutive attempts.

Then, the second stage corresponds to the SA search using a static cooling scheme starting from an initial high temperature T_0, so that many solutions with higher objective functions are accepted. Then, the current temperature T is reduced at every $L_{\max}$ iteration by means of a linear reduction factor r. After many experiments, the initial temperature T_0 was set to 10^7, the maximum number of iterations at the same temperature $L_{\max}$ set to 15, and the maximum number of reheating steps $M_{\max}$ set to 10, with a factor $r \in \{0.95, 0.98\}$ at each step. The values of T_0 and r are similar to the parameters used in [20]. However, $L_{\max}$ and $M_{\max}$ are slightly higher in order to give the algorithm the chance to further explore the solutions space. In fact, the value of $M_{\max}$ can be reduced as discussed in the following section, where it was noticed that the algorithm started converging since the 6th iteration.

3.6. Experimental Results. Figure 2 shows the convergence curve of the Simulated Annealing search for $M_{\max} = 10$ and $L_{\max} = 15$. The y-axis represents the objective function, given by (4), of the best solution recorded during the reheating process.

During the first reheating step, no improvement was noticed since neighboring solutions with very bad objective functions are accepted. Afterward, the solution starts to converge very fast, and then it gets slower until reaching the final solution after 8 reheating steps. The SA process reaches a constant state within few minutes, and the optimized arrangement computed by SA search is about 20% better than the initial one given by the greedy algorithm. After extensive experiments, the 50 best layouts with the lowest

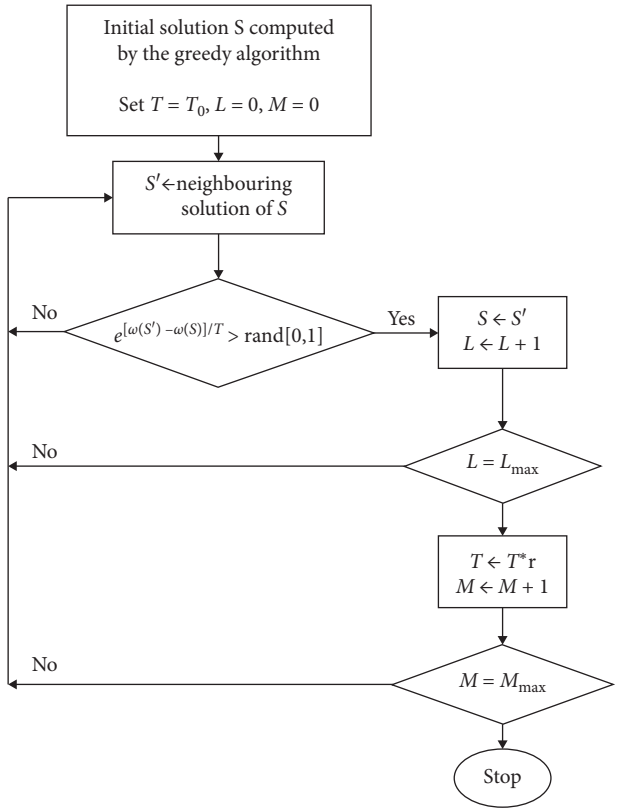


Figure 1: Simulated Annealing algorithm flow chart.

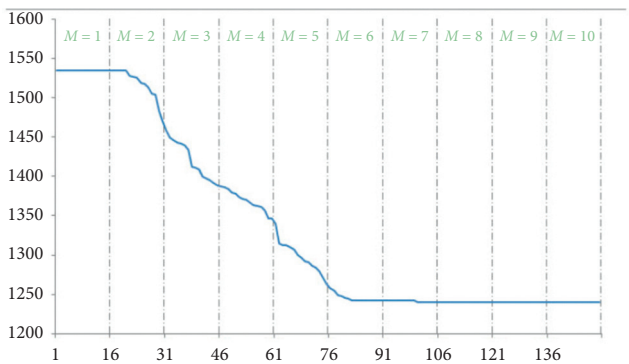


Figure 2: Convergence of the Simulated Annealing search for $M_{max} = 10$ and $L_{max} = 15$.

objective functions were recorded. It was noticed that the pattern highlighted in grey in Figure 3 appeared several times among the best layouts.

It can be noticed that the keys "Alef" and "Lam" (highlighted in red) are adjacent to each other and generally in the middle row close to the center of the keyboard. This pair of letters is equivalent to the article "The" in the English language, and thus, even in the standard keyboards, they are usually placed in the center of the layout. Similarly, it was also noticed that the pair "Ya" and "Ta" (highlighted in green) are adjacent and toward the center, almost in all computed layouts. In fact, while the first pair is frequently used in the beginning of the words in Arabic language, the latest pair marks the end of many words.

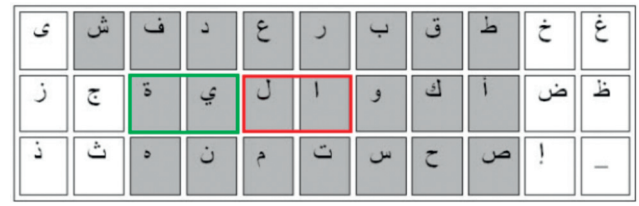


Figure 3: The optimized arrangement for the Arabic keyboard using Simulated Annealing algorithm.

Compared to the Genetic Algorithm and using the same objective function proposed in [35], the best result produced by the SA was 190.3687, which is very close to 190.3563, the best result produced by the GA. Moreover, the average of the solutions produced by the SA was slightly better than the average produced by the GA experiments. Furthermore, the average processing time of the SA was 10 times shorter than that of the GA. However, it should be noted that the processing time is not critical in this type of optimization, where the optimization is done only once and then the result is adopted for long-term use. The processing time might be significant in case of dynamic layout generation in adaptive user interfaces or in dynamic keyboards, where the keys arrangement, layout, and elements change according to the needs of the user or context.

3.7. Validation. With more than 11 thousands articles from different disciplines extracted from Wikipedia to compute the frequencies, this source can be considered reliable and not biased. However, in order to validate the optimized layouts, different sources were considered and 50 articles were randomly chosen from two Arabic newspapers covering 5 different disciplines (10 articles in each). Aljazeera [36] is a widely known broadcaster and is the first source of Arabic news and current affairs in the Arab world. Asharq AlAwsat [37] is the leading Arabic international daily newspaper, printed simultaneously on four continents in 14 cities. The distance and thus the movement time needed to type a given text were computed. The movement time is computed based on Fitts' law [27] which is a successful and well-studied model for the prediction of human movement time required to move to a target area. Using the parameters presented in Section 3.2, the time required to type consecutively two different symbols i and j can be estimated by $T_{ij} = (1/4.9) \cdot \log_2(d_{ij}/(W_j + 1))$ for $i \neq j$, and $T_{ij} = 0.127$ s if $i = j$. This metric was used to evaluate and compare the optimized keyboard arrangements to the commonly used Arabic keyboards used in the commercial mobile devices. First, comparing the virtual keyboards of the iPhone and the Samsung Galaxy systems, it can be noticed that they are very similar, especially the 1st and 2nd rows where the keys are mainly arranged based on the similarities of letters' transcription. Then, compared to the optimized layouts, it turned out that the optimized layout is 20% better than the currently available systems for mobile devices.

Figure 4 shows the estimated time, in seconds, required to type the given texts by the two layouts. It turned out that, for all the test cases, the estimated movement time using the optimized

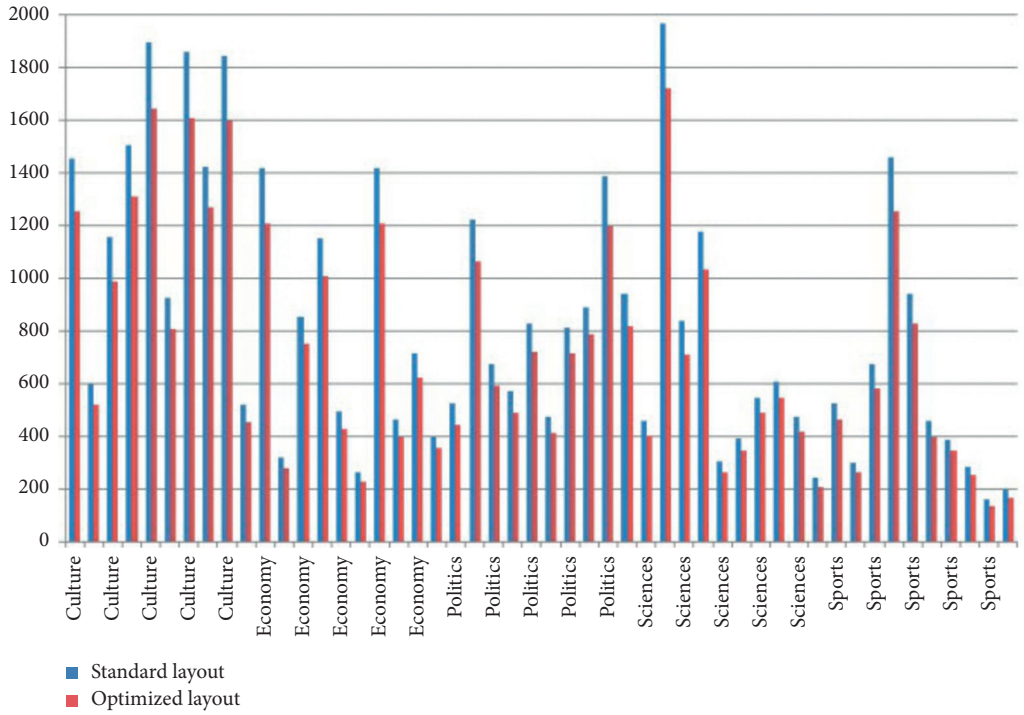


Figure 4: The estimated time (in seconds) to type the text for each test case.

layout is less than the currently used layout. The performance stability of the optimized keyboard can be noticed regardless of the different domains. Moreover, the enhancement is more important when the typed text is longer, as shown in Figure 5, where the x-axis represents the lengths of the articles (expressed in kilobytes (kB)), while the y-axis represents the difference between the estimated typing times of the commonly used layout and the optimized one (in seconds).

These results confirm that the arrangement of the keys may lead to a significant enhancement in typing performance since the improvement is accumulated in a repetitive task like text typing.

However, the main limitation that the proposed new layouts may face is that, even if the currently used keyboards are not optimum, a new arrangement of keys, even a better one, may not be adopted by users. Therefore, the theoretically optimized new layouts require an assessment involving the end-users and comparing them from a practical point of view.

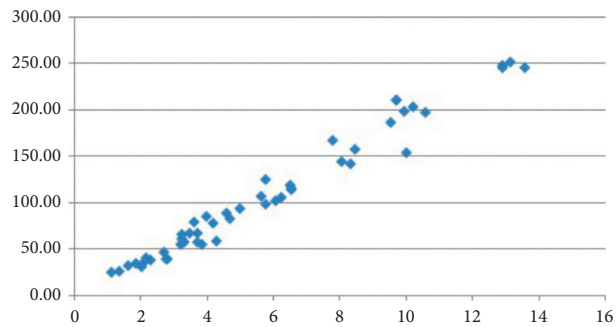


Figure 5: The estimated improvement (in seconds) depending on the articles' lengths (in kilobytes).

4. Usability Evaluation

The ISO 9241-11:2018 standard covers a wealth of information on every aspect of usability including hardware, software, and usability processes. It identifies efficiency, effectiveness, and satisfaction as major attributes of usability that is defined as "the extent to which a system, product or service can be used by specified users to achieve specified goals with effectiveness, efficiency and satisfaction in a specified context of use":

(i) Effectiveness measures the ability of specified users to completely and accurately achieve specified goals in specific environments;

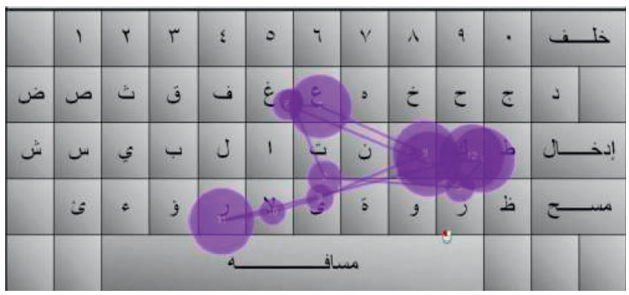
(ii) Efficiency measures the quantity of resources spent in comparison with the effectiveness of goals achieved;

(iii) Satisfaction measures how acceptable and comfortable the system is for its specific users.

In this paper, a task-based usability evaluation with actual users was conducted to assess the usability of the 3 layouts with respect to the above-mentioned criteria.

4.1. iWriter Software. The iWriter [17] is an Augmentative and Alternative Communication (AAC) system that is operating entirely by eye gaze and that was initially developed to provide alternative forms of communication for people with severe motor disabilities. It includes an embedded Arabic virtual keyboard. Some screenshots of the interface of the system are displayed in Figure 6.

(a)



(b)

Figure 7: Screenshots of the iWriter software [17].

A pilot experiment was conducted in [17] to assess the accuracy of eye fixation and to evaluate the key size. Six participants were asked to type four sentences with different variations in letters using the usually used virtual Arabic keyboard and Tobii X120 device to track user's eye gaze. Each participant was given a different combination of the four sentences in random order to control learning effects. The number of drifts that occurred while the participant fixated a static location on a screen were measured using Tobii studio by studying the gaze plots. This experiment showed a good accuracy of the typing process in general and that the size of the key used in the experiment (1.5 cm) supported a systematic visual pattern of selection and is thus appropriate as it can be perceived easily by the user.

Hence, in this study, the iWriter system has been extended to automatically generate a gaze-controlled interface for any given layout. In fact, the output of the optimization algorithm is given as a text file showing the arrangement of the letters through the main rows of the keyboard. This file is then injected into the iWriter system that automatically generates an interface for the optimized layout.

4.2. Evaluation Sessions. The usability evaluation was conducted as a task-based testing with two facilitators and involved volunteer participants who were asked to enter a text via the iWriter system using different layouts. The time and effort, needed by each user to enter the text via different layouts, were measured.

4.2.1. Participants. A total of 12 healthy participants were enrolled in this study. The age of the participants varied between 18 and 33 years old, with the majority (40%) aged between 18 and 21 years. All of the participants indicated that they were frequently using the Arabic keyboard, and 10 out of the 12 participants indicated that they were using it on a daily basis. It should be noted that Arabic is the native language for the context of the study or the location in which the study is conducted.

4.2.2. Procedure. The study was conducted in a time frame of two weeks. Each participant took part in 3 sessions to evaluate the 3 layouts with an interval ranging from 24 to 48 hours between each 2 consecutive sessions. Four short sentences, consisting of 10 words (60–65 characters) each, were extracted from local newspapers covering different areas including economy, science, politics, and sports. Each participant was asked to type all four sentences in every session in a random order. The experiment flow of each session is depicted in Figure 7.

All participants tested the commonly used layout, depicted in Figure 8(a), in their first session. After that, they were split into two equal groups. The first group tested the GA optimized layout shown in Figure 8(c), and then, the SA optimized layout is shown in Figure 8(b). The second group experienced the optimized layouts in the opposite order.

During each session, all four sentences were read clearly to the participants who were asked to type them using the keyboard displayed on the screen. A one-minute to two-minute rest separated every two sentences to avoid fatigue or stress on the participants' eyes. To rule out the impact of the learning factor, sentences were typed in a random order in each session. At the end of each session, they were requested to provide their overall experience about the keyboard that they have used. Finally, a System Usability Scale (SUS)

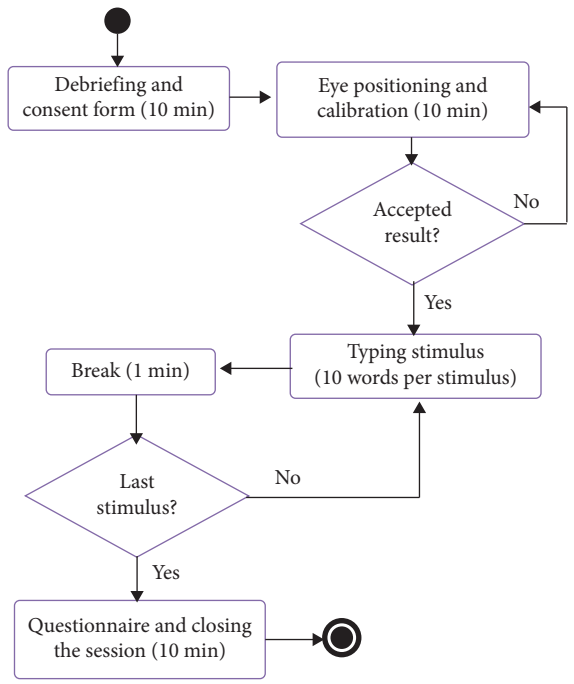


Figure 7: Procedure followed in each evaluation session.

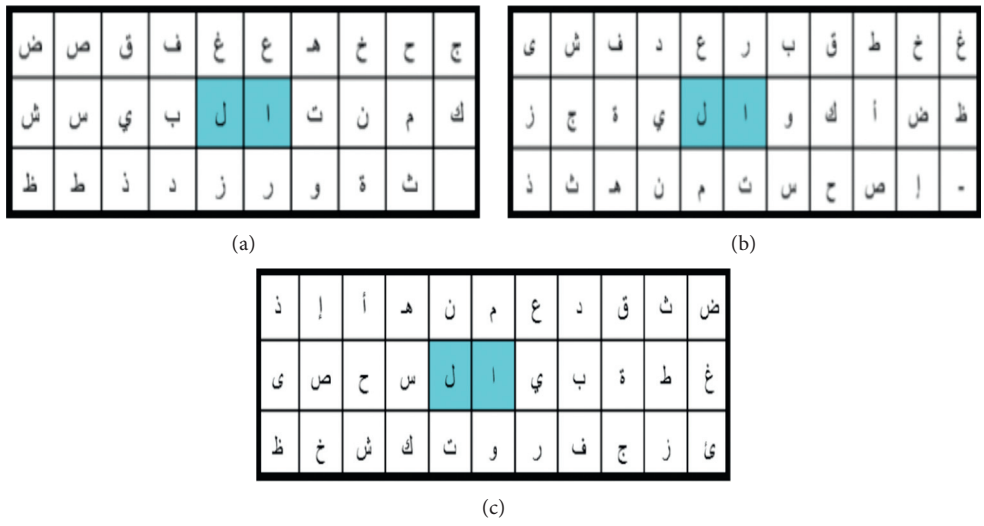


Figure 8: Keyboard layouts used in the experiments: (a) commonly used layout; (b) optimized layout by the SA; (c) optimized layout by the GA in [8].

survey was given to them to gather their impression about the keyboard that was used.

4.3. Results. Each usability criterion was evaluated through both direct and indirect assessment. The indirect assessment was conducted via a questionnaire through directly asking the participants at the end of each session. On the other hand, the direct assessment was conducted through an empirical usability evaluation method following a systematic way to collect data regarding the user's experience with the system being evaluated. The effectiveness was evaluated using the total distance parsed by the eye to type a given text as well as the number of errors in typing. The efficiency has been evaluated based on the actual typing time spent by the participants. The System Usability Scale (SUS) was used to evaluate the user's satisfaction including the likeability and learnability.

It should be noticed that the order of the stimuli was randomized to control learning effects in the experiments. However, no significant correlations were observed between the evaluation metrics and the order of the stimuli or the stimuli themselves. Moreover, no significant differences in the results related to demographic characteristics of the

participants were observed. Therefore, the results are aggregated among all the participants for all the presented stimuli.

4.3.1. Efficiency. The efficiency was evaluated using the text entry rate (typing speed) which corresponds to the time needed to type a given text. Table 2 shows the text entry rates for the three tested layouts. It depicts the average of the typing speed for each stimulus entered using the different layouts. It turned out that the commonly used layout outperforms the optimized ones in terms of typing speed but without significant differences.

It should be noticed that the typing speed was in average around 2 to 3 minutes per stimulus (3 to 5 words per minute (WPM)), except in few cases. For example, in the first session testing the common layout, Participant 5 was moving her head and lost eye contact with the eye tracker several times, so she spent the highest time to type the text (7 min and 37 sec) and made the highest number of errors in typing (14 errors in stimulus #1). Another case that should be noticed was faced by Participant 11 using the GA layout. In several trials, she mistakenly typed a wrong character while trying to select the space bar and thus needed more than 7 minutes to type the text. The results of the indirect assessment of the efficiency are depicted in Figure 9 confirming that typing on the common keyboard was relatively faster as perceived by the participants. This performance is actually due to the familiarity of the participants with the common keyboard. Further testing should be conducted to assess the performance after some time of using the optimized layouts.

Table 2: Text entry rates (min : sec).

	Common	GA	SA
Min	0 : 58	1 : 32	1 : 33
Avg	2 : 24	3 : 12	2 : 54
Max	7 : 37	7 : 57	8 : 16
Std. Dev.	0.05	0.06	0.05

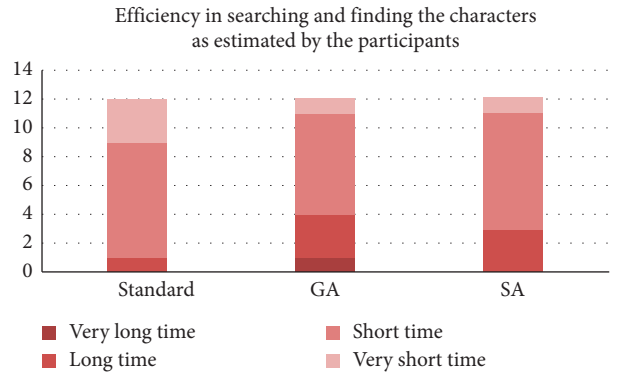


Figure 9: The efficiency as evaluated by the participants.

Table 3: Total parsed distances.

	Common	GA	SA
Min	390 mm	393 mm	380 mm
Avg	511 mm	549 mm	486 mm
Max	705 mm	971 mm	695 mm
Std. Dev.	63.08	122.89	62.41

4.3.2. Effectiveness. The effort needed to type a text was estimated by computing the total distance traveled by the eye to do this task. This distance is equal to the sum of distances between successive keys typing (between successive eyes' fixations) using the coordinates of the different eyes' fixations. Thus, the effectiveness has been evaluated using the total distance parsed by the eye to type a given text as well as the number of errors in typing.

A. *Total parsed distance.* The total parsed distances among all the participants for all the presented stimuli are depicted in Table 3 for each keyboard's layout. It shows the average of the total parsed distances among all the participants for each stimulus entered using different layouts. It can be noticed that the optimized layout generated by the SA algorithm particularly enhanced the distance parsed by the eye to enter a given text. Knowing that the stimulus consisted of a short sentence (10 words composed of 60 to 65 characters), the improvement in the performance can be more noticeable for longer texts as shown in Section 3.7.

On the other hand, the results of the optimized layout generated by the GA are surprisingly the worst. This might be due to the distribution of the characters with respect to the row weights that gives priority to the middle row rather than the keys that are around the center of the keyboard. Moreover, it was noticed that the participants struggled to find the desired character using this layout and made some errors while typing, which increases the total of the parsed distance.

The results of the indirect assessment of the effectiveness as evaluated by the participants are depicted in Figure 10 showing that the typing was relatively challenging on the optimized keyboards, particularly on the layout generated by the GA.

B. *Error rates.* The error rates among all the participants for all the presented stimuli are depicted in Table 4 for each keyboard's layout.

It can be noticed that the highest number of errors was recorded for the common keyboard. This is also confirmed by the results of the indirect assessment as perceived by the participants and shown in Figure 11.

This result might be due to the fact that the new layouts slowed down the typing speed and thus the users tended to make fewer errors while typing. Moreover, this result might be also related to the order of the sessions, since the first session was the most challenging. In fact, eight participants made most of the errors while typing in the first session, since they were not familiar with using the eye tracker. Hence, the number of errors that they made was not related to stimuli, which were randomized, but to the use of this new technology in typing. On the other hand, three participants made most of the errors in the second session, which was the first time that they were exposed to an optimized keyboard.

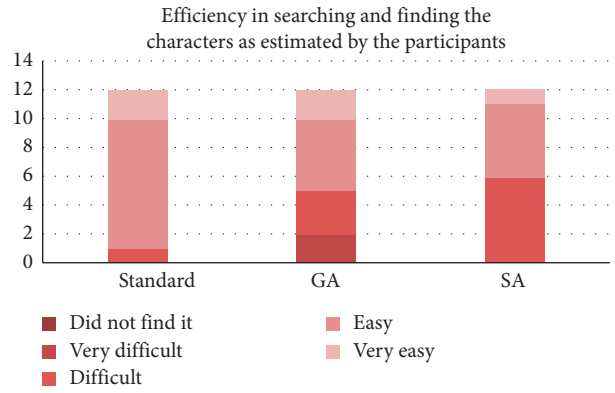


Figure 10: The effectiveness as evaluated by the participants.

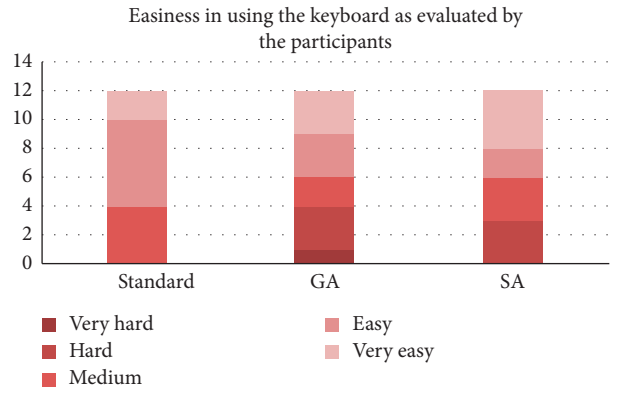


Figure 12: The usability as evaluated by the participants.

Table 4: Error rates.

	Common	GA	SA
Min	0 errors/stimulus	0 errors/stimulus	0 errors/stimulus
Avg	3 errors/stimulus	2 errors/stimulus	2 errors/stimulus
Max	14 errors/stimulus	8 errors/stimulus	5 errors/stimulus
Std. Dev.	2.83	1.81	1.67

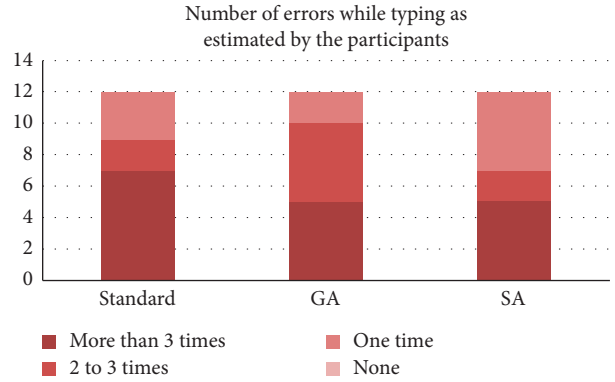


Figure 11: The error rates as evaluated by the participants.

Accordingly, the unfamiliarity with the typing technique or with the keyboard's layout has a particular impact on the error rate. This is confirmed by the total number of errors recorded for all participants which was 269 errors in the first session, 227 in the second session, and 159 in the third session. This improvement indicates that there is a gradual learning curve that can be particularly enhanced by practice.

4.3.3. Satisfaction. Finally, the participants were asked to give an overall evaluation of the three tested layouts, and the results confirm the order of preferences as expected from the above analysis, where the common layout comes first followed by the SA layout followed by the GA layout, as shown in Figure 12. However, most of the participants showed interest in the new layouts: 8 out of 12 would like to use the GA layout, and 11 out of 12 are particularly interested in the SA layout.

5. Conclusion

This paper presented an optimization model of keys' arrangement in the Arabic keyboard for applications that predominantly use just a single pointer. By calculating the total parsed distance while typing a given text, the results show that the optimized layout outperforms the commonly used one in terms of estimated typing speed. However, the usability evaluation showed that the familiarity of the participants with the commonly used keyboard has an impact on the typing speed but without significant differences. Moreover, it was noticed that there is a gradual learning curve that leads to promising results for the optimized layouts that can be particularly enhanced by practice.

Future work aims to test the optimized keyboards for users' convenience in order to assess the effect of including the hit direction and rows weights on the typing speed and comfort. The optimized layouts can be adopted in applications that rely on typing using one pointer and can be tested for use by people with special needs. In addition, the proposed usability evaluation framework can be used to evaluate optimized keyboards for other languages. Furthermore, this research work aimed to optimize the layout for virtual keyboards; it would be interesting to investigate the applicability of this study for the dynamic keyboard that has been recently patented by Apple [12].

Moreover, other research directions can be further explored such as modeling the problem using SK-QAP, a generalization of QAP. Moreover, it was noticed in the literature review that the Particle Swarm Optimization (PSO) has received little attention to solve the keyboard optimization problem. Recent research showed some efforts toward discretizing PSO and its variants in order to make them suitable for QAP. Hence, it would be promising to investigate their performance in solving the keyboard optimization problem compared to other competing algorithms.

Notations

d_{ij}: Distance between the two keys assigned to i and j
f_{ij}: Frequency between a pair of letters
T_{ij}: Movement time from one key to another
W_j: Width of the target key assigned to j
D_{ij}: Distance between a pair of letters
S_{ij}: Hit direction
R_i: Row i in the layout
T_0: Initial temperature in Simulated Annealing
T: Current temperature
$L_{\max}$: Maximum number of iterations at the same temperature
r: Linear reduction factor of the temperature
$M_{\max}$: Maximum number of reheating steps
a, b, c, α, and β: Constants.

Data Availability

The usability evaluation data used to support the findings of this study are included within the article.

Conflicts of Interest

The authors declare that there are no conflicts of interest regarding the publication of this paper.

Acknowledgments

The authors would like to sincerely thank the participants in the usability evaluation for their valuable time and feedback. This research project was supported by a grant from the Research Center of the Female Scientific and Medical Colleges, Deanship of Scientific Research, King Saud University.

References

[1] P. Majaranta, "Text entry by eye gaze," vol. 11, University of Tampere, Tampere, Finland, 2009, Ph.D. thesis.
[2] "Eye control for Windows 10," https://www.microsoft.com/en-us/garage/wall-of-fame/eye-control-windows-10/.
[3] J. Eggers, D. Feillet, S. Kehl, M. Oliver Wagner, and B. Yannou, "Optimization of the keyboard arrangement problem using an ant colony algorithm," *European Journal of Operational Research*, vol. 148, no. 3, pp. 672–686, 2003.
[4] M. O. Wagner, B. Yannou, S. Kehl, D. Feillet, and J. Eggers, "Ergonomic modelling and optimization of the keyboard arrangement with an ant colony algorithm," *Journal of Engineering Design*, vol. 14, no. 2, pp. 187–208, 2003.
[5] M. Wolosik and M. Tabedzki, "Screen keyboard arrangement optimization for polish language," *Advances in Computer Science Research*, vol. 13, pp. 75–93, 2016.
[6] T. G. Pradeepmon, V. V. Panicker, and R. Sridharan, "Hybrid estimation of distribution algorithms for solving a keyboard layout problem," *Journal of Industrial and Production Engineering*, vol. 35, no. 6, pp. 352–367, 2018.
[7] A. Benabid Najjar, "Toward an optimized Arabic keyboard design for single-pointer applications," in *Proceedings of the 15th Annual Conference Companion on Genetic and Evolutionary Computation (GECCO '13 Companion)*, C. Blum, Ed., ACM, New York, NY, USA, pp. 1717-1718, July 2013.
[8] N. Alswaidan, M. I. Hosny, and A. Benabid Najjar, "A genetic algorithm approach for optimizing a single-finger Arabic keyboard layout," *Intelligent Systems in Science and Information*, Springer, Berlin, Germany, 2015.
[9] Anon, "Keyboard design is effective in decreasing user pain," *IIE Solutions*, vol. 31, no. 7, p. 14, 1999.
[10] A. Dvorak, N. Merrick, W. Dealey, and G. Ford, *Typewriting Behavior*, American Book Company, Woodstock, GA, USA, 1936.
[11] *Optimus Maximus Keyboard*, Art. Lebedev Studio, Moscow, Russia, 2020, http://www.artlebedev.com/optimus/.
[12] R. Wilson James, "Electronic devices having keys with coherent fiber bundles," vol. 570, Apple Inc., Cupertino, CA, USA, 2020, United States Patent 10.877.
[13] O. Polacek, A. J. Sporka, and P. Slavik, *Text Input for Motor-Impaired People, Universal Access in the Information Society*, Springer, Berlin, Germany, 2007.
[14] S. Sarcar, P. Panwar, and T. Chakraborty, "EyeK: an efficient dwell-free eye gaze-based text entry system," in *Proceedings of the 11th Asia Pacific Conference on Computer Human Interaction (APCHI'13)*, pp. 215–220, Rotorua, New Zealand, September 2013.
[15] F. E. Sandnes, E. Eika, and F. O. Medola, "Reducing scanning keyboard input errors with extended start dwell-time, Advances in Usability, User Experience and Assistive Technology (AHFE 2018)," *Advances in Intelligent Systems and Computing*, Vol. 794, Springer, Berlin, Germany, 2019.
[16] H. Cecotti, Y. Meena, B. Bhushan, A. Dutta, and G. Prasad, "A multiscript gaze-based assistive virtual keyboard," in *Proceedings of the 2019 41st Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, IEEE, Berlin, Germany, July 2019.
[17] A. Al-Wabil, A. Al-Issa, I. Hazzaa, M. Al-Humaimeedi, L. Al-Tamimi, and B. Al-Kadhi, "Optimizing gaze typing for people with severe motor disabilities: the iWriter Arabic interface," in *Proceedings of the 14th International ACM SIGACCESS Conference on Computers and Accessibility (ASSETS'12)*, pp. 261-262, ACM, New York, NY, USA, October 2012.
[18] J. Ichbiah, "Method for designing an ergonomic one-finger keyboard and apparatus therefor," US Patent 5487616, 1996.
[19] Y. Li, L. Chen, and R. S. Goonetilleke, "A heuristic-based approach to optimize keyboard design for single-finger keying applications," *International Journal of Industrial Ergonomics*, vol. 36, no. 8, pp. 695–704, 2006.
[20] P.-Y. Yin and E.-P. Su, "Cyber Swarm optimization for general keyboard arrangement problem," *International Journal of Industrial Ergonomics*, vol. 41, no. 1, pp. 43–52, 2011.
[21] G. Murali and V. Panicker, "Optimization of a single finger keyboard layout using genetic algorithm and TOPSIS," *International Journal of Scientific and Engineering Research*, vol. 7, no. 2, pp. 102–105, 2016.
[22] M. Dell' Amico, J. C. D'iaz, M. Iori, and R. Montanari, "The single finger keyboard layout problem," *Computers and Operations Research*, vol. 36, pp. 3002–3012, 2009.
[23] A. B. Herthel and A. Subramanian, "Optimizing single-finger keyboard layouts on smartphones," *Computers and Operations Research*, vol. 120, 2020.
[24] T. Malas, S. Taifour, and G. Abandah, "Toward optimal Arabic keyboard layout using genetic algorithm," in *Proceedings of*

the 9th Int'l Middle Eastern Multiconference on Simulation and Modeling (MESM 2008), Amman, Jordan, August 2008.
[25] E. Khorshid, A. Alfadli, and M. Majeed, "A new optimal Arabic keyboard layout using genetic algorithm," *International Journal of Design Engineering*, vol. 3, pp. 25–40, 2010.
[26] M. Z. Osman, "Ergonomic Arabic keyboard," US Patent 667414S1.
[27] P. M. Fitts, "The information capacity of the human motor system in controlling the amplitude of movement," *Journal of Experimental Psychology*, vol. 47, no. 6, pp. 381–391, 1954.
[28] E. M. Loiola, N. M. M. d. Abreu, P. O. B. Netto, P. Hahn, and T. Querido, "A survey for the quadratic assignment problem," *European Journal of Operational Research*, vol. 176, no. 2, pp. 657–690, 2007.
[29] T. Gong and A. Tuson, "Particle Swarm optimization for quadratic assignment problems–A forma analysis approach," *International Journal of Computational Intelligence Research*, vol. 2, 2007.
[30] F. Hafiz and A. Abdennour, "Particle Swarm algorithm variants for the quadratic assignment problems-a probabilistic learning approach," *Expert Systems with Applications*, vol. 44, pp. 413–431, 2016.
[31] I. S. MacKenzie and R. W. Soukoreff, "Text entry for mobile computing: models and methods, theory and practice," *Human-Computer Interaction*, vol. 17, no. 2, pp. 147–198, 2002.
[32] R. Burkard, M. Dell'Amico, and S. Martello, *Assignment Problems*, SIAM, Philadelphia, PA, USA, 2008.
[33] Z. Drezner, P. M. Hahn, and É. D. Taillard, "Recent advances for the quadratic assignment problem with special emphasis on instances that are difficult for meta-heuristic methods," *Annals of Operations Research*, vol. 139, no. 1, pp. 65–94, 2005.
[34] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller, "Equation of state calculations by fast computing machines," *The Journal of Chemical Physics*, vol. 21, no. 6, p. 1087, 1953.
[35] M. I. Hosny, N. Alswaidan, and A. Benabid Najjar, "An optimized single-finger arabic keyboard layout," in *Proceedings of the 2014 Science and Information Conference*, pp. 321–328, August 2014, London, UK.
[36] Aljazeera News: http://www.aljazeera.net.
[37] Asharq Alawsat Newspaper: http://www.asharq-e.com/.