A New Virtual Keyboard with Finger Gesture Recognition for AR/VR Devices

Tae-Ho Lee and Hyuk-Jae Lee(✉)

Department of Electrical Engineering, Inter-university Semiconductor Research Center, Seoul National University, Seoul 151-742, Korea
{taehov,hyuk_jae_lee}@capp.snu.ac.kr

Abstract. This paper proposes a system that types the virtual keyboard by recognizing hand gestures in a single camera based environment. A virtual keyboard is designed in a multi-tab method of 3 × 4 arrays that are widely used in a mobile environment. In order to make it easy to identify the type-in action and consequently, to increase the type-in speed, this paper proposes a new definition of type-in action as the contact between the thumb and the index finger. To further reduce the input time, the keyboard layout is optimized by efficiently arranging alphabet keys according to the frequency of character appearance. Experimental results show that the proposed type-in action is effective to give commands with a virtual keyboard. Furthermore, the proposed keyboard layout achieves the speed-up by an average of 46.16% compared to the most conventional 'ABC' keyboard.

Keywords: Hand gesture · Virtual keyboard · Convex-hull
Keyboard layout optimization · Hand recognition

1 Introduction

Recently, augmented reality/virtual reality (AR/VR) has attracted great attention since major companies introduced commercial devices that enable AR/VR experience [1]. Smart glasses for AR applications have been developed for a number of companies. Head-mounted and see-thru displays have also been introduced and used by emerging applications. Hand gesture recognition is widely used as an interface to give commands to AR/VR devices [2, 3]. By each hand gesture representing a predefined command, it is effective to input a fixed number of commands [4–8]. For a large number of commands, an AR/VR device should support a virtual keyboard to input characters, like a computer or smartphone keyboard. To this end, a virtual keyboard interface using hand gesture recognition has been investigated extensively and developed successfully with various types of sensors. However, the virtual keyboard with a low-cost camera has not been implemented successfully even though its need is growing [9].

The research on hand gesture recognition for a virtual keyboard interface has actively been under way [10, 11]. Markussen et al. propose a virtual keyboard interface by using the hand gesture of catching the target key in the air while watching the virtual keyboard projected on the wall [12]. A high-resolution display is used in this study and a user

M. Kurosu (Ed.): HCI 2018, LNCS 10903, pp. 56–67, 2018.
https://doi.org/10.1007/978-3-319-91250-9_5

wears gloves to keep track of the position of the hand. Togootogtokh et al. propose fingertip movement recognition for character input with a virtual keyboard by using the geometric features of a hand [13]. This approach is advantageous over the previous one because it does not require additional equipment other than a low-cost camera. However, it is difficult to identify the exact timing of the type-in action by hand gesture. This difficulty makes it not easy to complete the type-in action, and consequently, decreases the type-in speed, which is the main disadvantage of this approach.

For keyboards for PCs and smartphones, a number of research activities have been made to increase the type-in speed by re-arranging the keyboard layout [14–16]. Bi et al. attempt to increase the character typing speed by modifying QWERTY keyboard and optimizing the keyboard layout in terms of clarity, input speed, and QWERTY-friendliness [17]. Arnott et al. propose 'Frequency keyboard' to evenly arrange characters so that the frequency of tapping each key is as close as possible [18]. Yin et al. propose 'Cyber Swarm' keyboard by using a general mathematical model that integrates various keyboard layouts. In addition, it accommodates ergonomic criteria and ambiguity effects at the same time [19].

Even with the optimized keyboard layout, the type-in speed by hand gesture recognition still remains too slow for a practical use in an AR/VR application because of the fundamental difficulty in the detection of the exact timing of type-in action. In order to overcome this difficulty and increase the type-in speed, this study proposes a new hand gesture recognition that is easy to identify the type-in action. To this end, the type-in gesture is defined as the contact between the thumb and the index finger. The index finger points to the input character and remains still the same position while the contact of the thumb to the index finger makes the type-in action. This gesture is easy to be recognized by an image sensor. Furthermore, a user can sense the touch of the two fingers by himself/herself so that he/she can start the next gesture as soon as the contact is made. This self-recognition of the type-in action can increase the type-in speed like a PC or smartphone keyboard for which a user can sense the type-in action from the touch feeling from fingers. The keyboard layout for the proposed type-in action is optimized for further speed-up of character inputs. Simulation results show that the proposed virtual keyboard achieves the speed-up by an average of 46.16% compared to the most conventional 'ABC' keyboard.

The rest of this paper is organized as follows. Section 2 introduces related work and Sect. 3 describes the proposed keyboard input method based on hand gesture recognition. Section 4 proposes an optimization of the keyboard layout to increase the type-in speed. Section 5 presents experimental results and Sect. 6 draws the conclusion of this paper.

2 Related Work

This section presents the previous work that gives a mathematical model to estimate the time to type in characters for a given keyboard layout. This model is used to optimize the layout for the minimization of type-in time.

This section introduces Fitt's Law to estimate the type-in time with a mathematical model [20]. Let MT denote the type-in time that includes the time to move from the

previous key to the current key and also the time to click the current key. Then, Fitt's law formulates *MT* as follows:

$$MT = a + b \cdot \log_2(\frac{D}{W} + 1) \tag{1}$$

where D is the distance between the keys, and W is the width of the key. Coefficients a and b are the parameters to be determined by the characteristics of the keyboard. An example to explain D and W in (1) is shown in Fig. 1 where character 'M' is about to type in and character 'T' is the previous character. For the sake of simplicity, only nine keys in the center of the computer keyboard are presented. The movement distance D represents the distance between the center positions of 'T' and 'M' whereas W represents the width of each key.

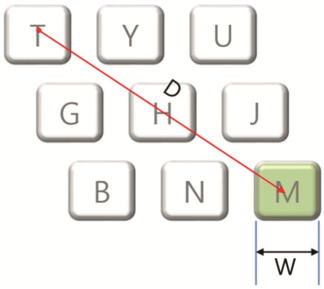


Fig. 1. Example keyboard type-in from character 'T' to 'M'

A keyboard layout has an impact on the typing speed. If back-to-back input characters are placed close to each other, the input time is fast [19, 21, 22]. If K characters are typed in, then the optimal layout can be obtained by minimizing the following formulation:

$$T_{tot} = \sum_{k=1}^{K} (MT(k)) \tag{2}$$

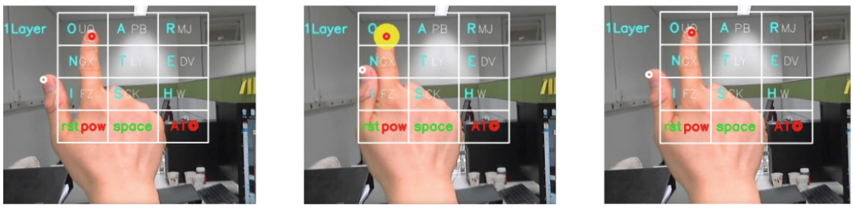
where $MT(k)$ represents the input time of the kth character. In (2), T_{tot} represents the time to enter entire K characters.

3 Proposed Algorithm

3.1 Definition of Type-In Gesture

This section proposes a new definition of the hand gesture that is easy to represent the type-in action. The definition is explained with Fig. 2 which shows an example of typing in character 'O'. In the first step, the index finger is placed on 'O' key of the virtual keyboard (Fig. 2(a)). The thumb is separated from the index finger at this initial step as shown in Fig. 2(a). The corresponding character is typed in when the thumb touches the

index finger (Fig. 2(b)). After completion of the type-in action, the thumb is separated again from the index finger (Fig. 2(c)). In this way, the index finger moves to the next key to repeat the type-in gesture for the next character. The proposed type-in gesture is simple and clear even with a slight movement. Therefore, it is possible to input quickly and to allow a high recognition rate.

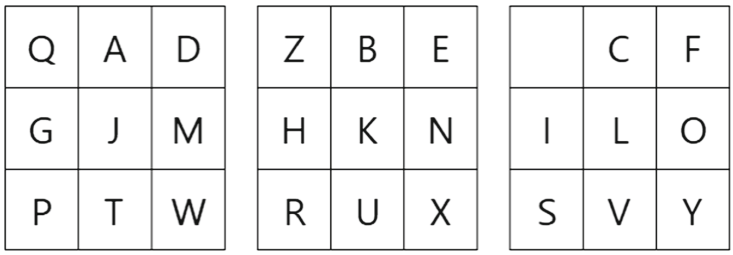


(a) (b) (c)

Fig. 2. Definition of type-in gesture (a) initial step, (b) type-in action, (c) finial step

3.2 Multi-layer Keyboard Design

The proposed type-in gesture allows only a single character pointed by the index finger to be type-in per each gesture. This is different from a PC keyboard for which multiple fingers are used together to increase the type-in speed. On the contrary, the proposed type-in method is somewhat similar to a smartphone keyboard for which one or two fingers are used for type-in. Therefore, the proposed type-in gesture is appropriate to type in a small keyboard of about 3×3 arrays like a smartphone keyboard. In this case, a keyboard consists of only 9 characters. Therefore, it is necessary to have 3 sets of keyboards to represent 26 characters in alphabet. This is also similar to a smartphone keyboard. Figure 3 shows an example layout of 26 alphabet using three 3×3 arrays. 9 characters are assigned to layer-1 and layer-2, respectively, and 8 characters to layer-3. With the three layers, it is necessary to define an additional gesture for the selection of one layer among the three.



(a) (b) (c)

Fig. 3. 3×3 keyboard arrays (a) 1-layer characters (b) 2-layer characters (c) 3-layer characters

A straightforward way to select a layer is similar to that used in smart-phones. There exists a default character array which is selected if the type-in gesture is performed. If

the second array is to be selected, the contact of the thumb and index finger is performed twice. If the third array is to be selected, the contacts are made three times. In the example shown in Fig. 3, character 'A' is the second key is in the first array. For typing 'A' in, the finger index points to this character with a single contact of the thumb and index finger. For character 'B', the index finger points to this character and then the thumb contacts to the index finger twice. If the same character needs to be type-in consecutively, two type-in contacts between the thumb and index finger may be recognized as 'B' in the second array. To avoid this confusion, the type-in method used by a smartphone is adopted. In this case, the type-in action is delayed by a sufficient amount of time to distinguish between back-to-back type-ins of 'A' or a single type-in of 'B'.

4 Virtual Keyboard Layout Optimization

As discussed in the previous section, the virtual keyboard with hand gesture is best used with a small size such as a smartphone keyboard. Therefore, this section discusses the optimal layout of a small keyboard aiming to increase the input speed. Similar to a smartphone keyboard, alphabet characters are placed in the upper 3×3 keys of the keyboard. The 26 alphabet characters are arranged as 9, 9, and 8 in 1st layer, 2nd layer, and 3rd layer, respectively. The arrangement shown in Fig. 3 is an example. For a systematic optimization, formulation of the objective functions is necessary, which is discussed in Subsect. 4.1. Given the objective function, a heuristic to select the optimal keyboard layout is discussed in Subsect. 4.2

4.1 Formulation of the Objective Function

The objective function given in (2) based on Fitt's law is formulated for a keyboard with a single layer, and therefore, it is not suitable for that with multiple layers. A new objective function is defined as follows for a multi-layer keyboard:

$$T_{tot} = \sum_{k=1}^{l} (MT(k) + S(k)) \tag{3}$$

where additional term *S(k)* represents the time to select the layer among multiple layers. This layer selection time *S(k)* depends on how the keyboard layout is selected. For the selection method discussed in the previous section, *S(k)* is expressed as follows:

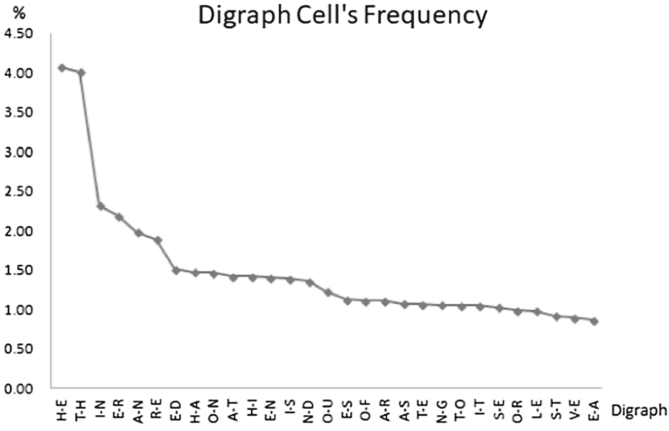
$$S(k) = C(k) \cdot n(k) + W(k) \tag{4}$$

where *C(k)* is the time for the type-in gesture and *n(k)* is the number of type-in actions for layout selection. The last term, *W(k)*, is the waiting time when the same key is to be typed-in repeatedly. For *n(k)*, its value is 1 for the characters in the first layer, 2 and 3 for the second and third layers, respectively. For *W(k)*, the waiting operation is similar to that for smartphones so that 1.8 s is chosen from the waiting time of smartphones.

4.2 A Heuristic for Keyboard Optimization

The minimization of (4) requires to solve a combinatorial optimization problem, which takes too much time to obtain the optimal solution. Therefore, this subsection proposes a greedy heuristic that is derived from the statistics about the type-in frequency of alphabet. To obtain the statistics, experiments are conducted to count how many times each character appear in various books. The books used for these experiments include 'Life English 500' that comprises the most commonly used English sentences by Americans. Additional books used for experiments are 'Alice's Adventures in Wonderland', '20,000 Leagues Under the Sea', 'Gulliver's Travels', and 'Les Miserables'. All these five books include 806,195 characters.

(a) frequency of appearance of each character



(b) frequency of appearance of consecutive characters

Fig. 4. Histogram of character appearance in five books

Figure 4(a) shows the number of appearances of 26 characters in the five books. Among 26 alphabet characters, 'E' appears most frequently by 13.12% whereas 'Z' appears least frequently by 0.08%. For other characters, their percentages of appearances are also given in the graph. Figure 4(b) shows the statistics of the two keys typed-in consecutively. The leftmost graph denoted by 'H-E' represents the number of appearances that character 'E' follows immediately after 'H'. This appearance takes place most frequently by 4.07%. The next case denoted by 'T-H' represents the character 'T' followed by 'H.' This statistics is also important because the type-in time in (1) depends on the distance between consecutive keys.

The value of $S(k)$ in (4) is small if $n(k) = 1,\ldots, k$ belongs to the first keyboard layer. Therefore, the arrangement of frequent characters in the first keyboard layer is effective in reducing T_{tot} in (4). The most frequent nine characters are selected for the first keyboard layer, and the second frequent nine characters are selected for the second layer. The remaining eight characters are placed in the third layer. From the observation shown in Fig. 4(a), characters 'E T A O H I N S R' are chosen for layer-1 and 'D L U C M W F G P' are selected for layer-2. The remaining eight characters, 'Y B V K J X Q Z' are chosen for layer-3. After selecting the keyboard layers, the next step is to find the optimal layout in each array. For the first and second layers, 9! layouts are possible. Therefore, T_{tot} in (3) is calculated for all 9! possible layouts and then the optimal layout is selected. Although the derivation of (3) for 9! times requires a large amount of computation, it is affordable complexity because it needs to be computed just once to form the layout. The derivation of the third layout requires 8! derivations of (3) because eight characters are arranged in this layout. For the derivation of (1), (3) and (4), the parameters used in these equations are determined with experiments. The time of the type-in operation, $C(k)$, is measured as 0.488 s. The parameters in (1) are experimentally obtained as $a = 488$ ms, $b = 105$ ms, and $W = 5.25$. Parameter D depends on the two characters typed consecutively. For example, D is 10.50 for the distance between 'N' and 'H'. As mentioned in Sect. 3.1, $W(k)$ is chosen as 1.8 s. The complexity of the above derivation requires 766,080 (i.e. 9!(layer-1) + 9!(layer-2) + 8!(layer-3)) computations of (3). On the other hand, 26! computations of (3) are necessary if the optimization is performed for all three layers together. Even though the computation is performed just once, 26! is too large to derive in a reasonable amount of time. On the other hand, 9! + 9! + 8! is much less complex which is affordable. As a result of the optimization with 766,080 comparisons, three layers are derived as shown in Fig. 3.

5 Experimental Results

5.1 Implementation of Click Gesture Recognition

To recognize the click gesture, the hand area is first segmented from the image, and the finger positions are extracted. The movement is recognized by tracking position changes. Various studies have been proposed for hand movement or finger recognition. Figure 5 shows the finger recognition algorithm used in this paper.

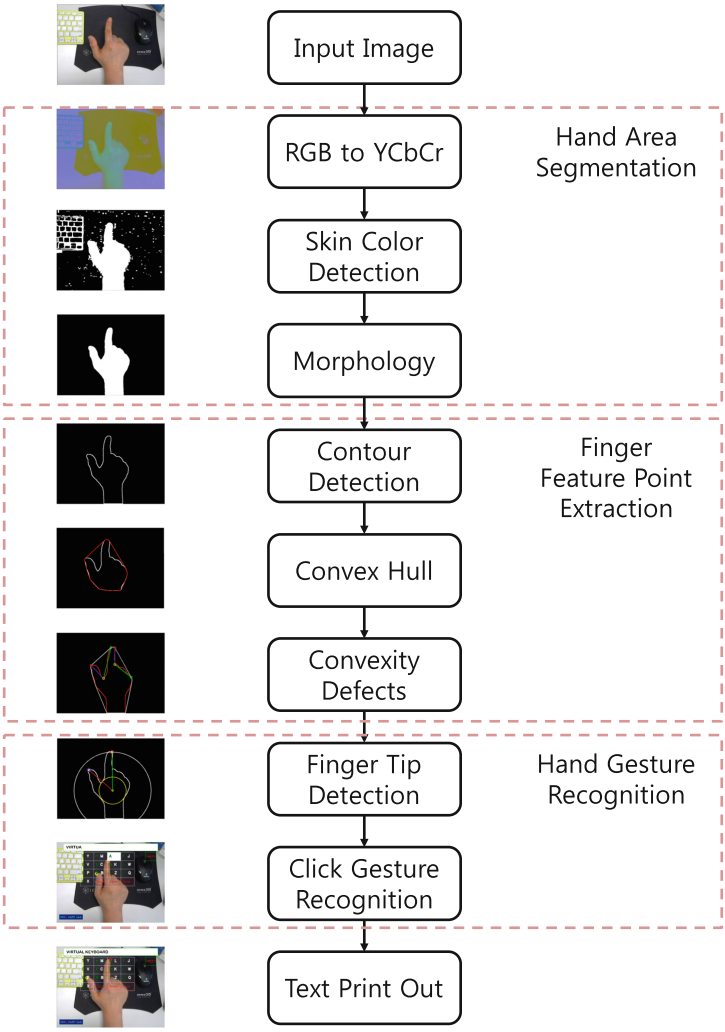


Fig. 5. The flowchart of the click gesture recognition used in this paper (Color figure online)

The RGB image obtained from the camera is highly influenced by even tiny change of illumination, and therefore, it is converted to YCbCr image that is less influenced by illumination [23]. To create a binary image of the hand area, the skin color is separated from the background using Cb and Cr values. The hand area is separated through morphological operations of erosion and dilation in order to remove noise from the binary image. The hand and finger information is searched with the convex hull and convexity defects for the separated hand area, and the positions of the thumb and index finger are tracked. When the thumb touches the body of the index finger, it is recognized as a click gesture, and the position of the index finger is generated as the final output. Further details of this algorithm are given in [24].

5.2 Experimental Evaluation of the Proposed Virtual Keyboard System

In this experiment, the size of each key is chosen as 5.25 (cm) and the values of a and b in the moving time (MT) of (1) are chosen as 488 ms and 105 ms, respectively, that are obtained from experiment.

The first simulation is conducted to show the effectiveness of the keyboard layout presented in Sect. 4 and its results are given in Table 1. The character input time is compared by simulation in the manner described in Sect. 3. The proposed keyboard is compared with conventional mobile keyboards that are 'ABC', 'TOC', 'Swarm', 'Levine', 'frequency' and 'QKLP' keyboards. The second to the last column denoted by 'Proposed' shows the input time using the multi-layer keyboard suggested in Sect. 3. The proposed multi-layer keyboard achieves the speed-up by an average of 46.16% compared to 'ABC'-type keyboard. Even for the comparison with the fastest keyboard among existing keyboards, the proposed keyboard achieves the speed-up of 3.49% on average.

Table 1. Simulation comparison of type-in speed among various keyboards

Sample data		Reference layout						Proposed layout
Text	Number of words	ABC	TOC	Swarm	Levine	Frequency	QLKP	Multi-layer
Life English	1,577	8,096	7,504	7,461	5,964	5,955	5,855	5,695
Alice	28,378	160,863	151,023	146,936	117,758	118,451	116,148	112,022
20,000 leagues	101,657	639,534	592,632	574,458	459,837	457,597	447,143	429,769
Gulliver	106,015	652,641	600,276	586,528	473,522	467,386	457,792	443,067
LesMiserables	568,568	3,568,950	3,315,930	3,193,890	2,565,990	2,554,590	2,481,980	2,395,970

The next experiment is conducted to evaluate the effectiveness of the overall keyboard system that includes the type-in action proposed in Sect. 3 and the layout optimization in Sect. 4. To this end, type-in speeds are evaluated with large keyboard sizes, 4×4 and 4×5. The layouts of these keyboards are optimized as explained in Sect. 4.2. For experiment, the following three texts are used to measure the input time. These are 'Hello world', 'Seoul national university' and 'A new keyboard typing system with hand gesture recognition', 'Old friends and old wine are best', 'Learn from yesterday, live for today, hope for tomorrow', 'You can make more friends with your ears than your mouth', 'Books are ships which pass through the vast seas of time'. Each text is attempted by 7 participants and the average input time is presented in Table 2. The error rate represents the percentage of the cases when wrong texts are typed. CPM represents the average number of characters that are typed in a minute. For comparison, QWERTY keyboard, 3×3 ABC-type keyboard are also evaluated and compared in Table 2. The experimental results show that the proposed keyboard with 3×3 size is the fastest among all keyboards. The next fast keyboard is the proposed with 4×5 size. For comparison of error occurrence, QWERTY keyboard and the proposed 3×3 keyboards result in relative small error rates. For the other three keyboards, error rates are relatively large. The experiment results for CPM measurement show that the

proposed 3 × 3 keyboard achieves the type-in speed of faster than 40 characters per minute which is fast enough for a practical use as a virtual keyboard.

Table 2. Experimental comparison of type-in efficiency among keyboard types

Experiment	Data	QWERTY	ABC	Proposed (3 × 3)	Proposed (4 × 4)	Proposed (4 × 5)
Hello	Time	17.21	19.37	14.43	16.28	15.78
	Error	0.75	1.05	0.63	0.91	0.89
	CPM	34.86	30.98	41.58	36.86	38.02
Seoul	Time	38.64	43.53	31.67	36.81	32.98
	Error	1.25	1.66	1.27	1.85	1.56
	CPM	35.71	31.70	43.57	37.49	41.84
Hand gesture recognition	Time	91.25	99.78	81.23	90.21	84.58
	Error	4.05	4.25	4.03	4.52	4.36
	CPM	32.88	30.07	36.93	33.26	35.47
Old friends	Time	44.06	46.23	37.93	39.59	38.97
	Error	2.37	2.59	2.31	2.58	2.42
	CPM	36.77	35.04	42.71	40.92	41.57
Learn from yesterday	Time	77.52	81.91	70.16	75.23	73.9
	Error	4.18	4.52	4.12	4.41	4.35
	CPM	34.83	32.96	38.48	35.89	36.54
Make more friends	Time	86.4	85.77	79.53	82.29	81.12
	Error	5.24	5.72	5.19	5.63	5.47
	CPM	38.89	39.17	42.25	40.83	41.42
Books are ships	Time	86.89	86.09	81.52	83.13	82.67
	Error	5.17	5.62	5.17	5.69	5.41
	CPM	38.67	39.03	41.22	40.42	40.64

6 Conclusion

This paper proposes a new definition of a type-in action that is the contact of the thumb to the index finger while the character to be selected is pointed by the index finger. This gesture is easy to be recognized by an image sensor. Furthermore, the proposed definition increases the type-in speed thanks to the self-recognition of the type-in action. Simulation results show that the proposed virtual keyboard achieves the speed-up by an average of 46.16% compared to the most conventional 'ABC' keyboard. Experimental results also show that the proposed keyboard achieves the type-in speed of faster than 40 characters per minute which is fast enough for a practical use as a virtual keyboard.

Acknowledgments. This work was supported by the World Class 300 Project (R&D) (S2482780, Development of Core Technologies for 3D Sensing Camera Module) of the SMBA(Korea) and by "The Project of Industrial Technology Innovation" through the Ministry of Trade, Industry and Energy(MOTIE) (10082585,2017).

References

1. Kato, H., Billinghurst, M., Poupyrev, I., Imamoto, K., Tachibana, K.: Virtual object manipulation on a table-top AR environment. In: IEEE and ACM International Symposium on Augmented Reality, pp. 111–119, October 2000
2. Lee, T., Hollerer, T.: Handy AR: Markerless inspection of augmented reality objects using fingertip tracking. In: 11th IEEE International Symposium on Wearable Computers, pp. 83–90, October 2007
3. LaViola, J.J.: A survey of hand posture and gesture recognition techniques and technology. Technical report CS-99-11. Brown University, Providence (1999)
4. Argyros, A.A., Lourakis, M.I.A.: Vision-based interpretation of hand gestures for remote control of a computer mouse. In: Huang, T.S., Sebe, N., Lew, M.S., Pavlović, V., Kölsch, M., Galata, A., Kisačanin, B. (eds.) ECCV 2006. LNCS, vol. 3979, pp. 40–51. Springer, Heidelberg (2006). https://doi.org/10.1007/11754336_5
5. Liu, N., Lovell, B.C.: Hand gesture extraction by active shape models. In: Proceedings of the Digital Imaging Computing: Techniques and Applications, pp. 1–6, December 2005
6. Lockton, R., Fitzgibbon, A.W.: Real-time gesture recognition using deterministic boosting. In: Proceedings of British Machine Vision Conference, vol. 2, pp. 817–826, September 2002
7. Lee, J.W., Lee, K.S., Lee, H.J.: 3D-convolutional neural network for efficient hand gesture recognition. In: International Conference on Electronics, Information, and Communication, January 2017
8. Lee, H.S., Lee, D.H., Kim, J.S., Lee, H.J.: Fast hand gesture recognition with CNN and feature matching. In: 30th Workshop on Image Processing and Image Understanding, February 2018
9. Kolsch, M., Turk, M.: Keyboards without keyboards: a survey of virtual keyboards. Technical report 2002-21. University of California, Santa Barbara, July 2002
10. Sarkar, A.R., Sanyal, G., Majumder, S.: Hand gesture recognition systems: a survey. Int. J. Comput. Appl. **71**(15), 0975–8887 (2013)
11. Qin, S., Zhu, X., Yang, Y.: Real-time hand gesture recognition from depth images using convex shape decomposition method. J. Sig. Process. **74**(1), 47–58 (2014)
12. Markussen, A., Jakobsen, M.R., Hornbæk, K.: Vulture: a mid-air word-gesture keyboard. In: Proceedings of CHI, pp. 1073–1082. ACM (2014)
13. Bergounioux, M.: Applications. Introduction au traitement mathématique des Images - méthodes déterministes. MA, vol. 76, pp. 157–176. Springer, Heidelberg (2015). https://doi.org/10.1007/978-3-662-46539-4_7
14. MacKenzie, I.S., Zhang, S.X.: 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, May 1999
15. Levine, S.H., Trepagnier, G.C., Getschow, C.O., Minneman, S.L.: Multi-character key text entry using computer disambiguation. In: Proceedings of the Tenth Annual Conference on Rehabilitation Engineering, pp. 177–179, June 1987
16. Foulds, R.A., Soede, M., Van Balkom, H.: Statistical disambiguation of multi-character keys applied to reduce motor requirements for augmentative and alternative communication. Altern. Augment. Commun. **3**, 192–195 (1987)
17. Bi, X., Zhai, S.: IJQwerty: what difference does one key change make? Gesture typing keyboard optimization bounded by one key position change from Qwerty. In: Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, p. 49-8, May 2016
18. Arnott, J.L., Javed, M.Y.: Probabilistic character disambiguation for reduced keyboards using small text samples. Augment. Altern. Commun. **8**(3), 215–223 (1992)

19. Yin, P.Y., Su, E.P.: Cyber swarm optimization for general keyboard arrangement problem. Int. J. Ind. Ergonomics **41**, 43–52 (2011)
20. Fitts, P.M.: The information capacity of the human motor system in controlling the amplitude of movement. J. Exp. Psychol. **47**, 381–391 (1954)
21. Smith, B.A., Bi, X., Zhai, S.: Optimizing touchscreen keyboards for gesture typing. In: Proceedings of the CHI 2015 Conference on Human Factors in Computer Systems, pp. 18–23, April 2015
22. Zhai, S., Hunter, M., Smith, B.A.: The metropolis keyboard: an exploration of quantitative techniques for virtual keyboard design. In: Proceedings of the UIST 2000 Symposium on User Interface Software and Technology, pp. 119–128, November 2000
23. Kovac, J., Peer, P., Solina, F.: Human skin color clustering for face detection. In: International Conference on Computer as a Tool, EUROCON 2003, pp. 144–148, September 2003
24. Lee, T.H.: A new keyboard typing system with hand gesture recognition. Doctoral dissertation, Seoul National University, February 2018