The Visual Computer (2020) 36:1559–1572
https://doi.org/10.1007/s00371-019-01758-8

ORIGINAL ARTICLE

Gestural flick input-based non-touch interface for character input

Md. Abdur Rahim[1] · Jungpil Shin[1] · Md. Rashedul Islam[1]

Published online: 1 October 2019

Abstract
A non-touch character input is a modern system for communication between humans and computers that can help the user to interact with a computer, a machine, or a robot in unavoidable circumstances or industrial life. There have been many studies in the field of touch and non-touch character input systems (i.e., hand gesture languages), such as aerial handwriting, sign languages, and the finger alphabet. However, many previously developed systems require substantial effort in terms of learning and overhead processing for character recognition. To address this issue, this paper proposes a gesture flick input system that offers a quick and easy input method using a hygienic and safe non-touch character input system. In the proposed model, the position and state of the hands (i.e., open or closed) are recognized to enable flick input and to relocate and resize the on-screen virtual keyboard for the user. In addition, this system recognizes hand gestures that perform certain motion functions, such as delete, add a space, insert a new line, and select language, an approach which reduces the need for recognition of a large number of overhead gestures for the characters. To reduce the image-processing overhead and eliminate the surrounding noise and light effects, body index skeleton information from the Kinect sensor is used. The proposed system is evaluated based on the following factors: (a) character selection, recognition and speed of character input (in Japanese hiragana, English, and numerals); and (b) accuracy of gestures for the motion functions. The system is then compared to state-of-the-art algorithms. A questionnaire survey was also conducted to measure the user acceptance and usability of this system. The experimental results show that the average recognition rates for characters and motion functions were 98.61% and 97.5%, respectively, thus demonstrating the superiority of the proposed model compared to the state-of-the-art algorithms.

Keywords Human–computer interaction · Hand gestures · Non-touch input · Gesture recognition · Kinect sensor

1 Introduction

Humans communicate with computers in many ways, and the interface between humans and computers is an important factor in the convenience of this interaction. Most of the research in this field therefore aims to improve the usability of the computer interface and to enhance human–computer interactions [1]. In the last few decades, the keyboard and mouse have played important roles in human–computer interactions. At present, the user is expected to use both touch and non-touch interfaces to interact with the machine, and various different devices are used in daily life, such as computers, tablet PCs, and smartphones. A keyboard is typically used to input characters on computers, and software keyboards to input characters on smartphones or tablet PCs. The user needs to touch the keyboard to input characters, even on smartphones or tablet PCs. There is also a large demand in industry to provide a hygienic and safe environment, and rugged and reliable computing and automation devices may be needed to increase production. These tough environments and tight regulations require special equipment to keep the environment sterile and safe. In order to meet the highest hygiene and safety requirements, to ensure the traceability of all processing steps and to ensure the implementation of existing standards, many industrial applications have used advanced human–machine interface (HMI) technology [2]. There are many industrial computer production companies (e.g., JLT mobile computers, Noax Technology, Teguar Computer, and others) that offer fully sealed, waterproof, industrial touch-screen automation computers for production applications to

✉ Jungpil Shin
jpshin@u-aizu.ac.jp

Md. Abdur Rahim
rahim_bds@yahoo.com

Md. Rashedul Islam
rashed.cse@gmail.com

1 School of Computer Science and Engineering, The University of Aizu, Fukushima 965-8580, Japan

ensure a hygienic environment. However, there are several problems with the touch input methods, for example:

1. Users cannot input characters in circumstances when traditional input devices such as keyboards or mice are not available or applicable, or when they cannot touch any devices.
2. Users need to be in a safe place to input the characters in industrial environments or food factories, and when using touch devices in unhygienic or unsafe environments.
3. When using a screen on a touch device (e.g., in an ATM), attackers can gather users' information and biometrics, which can cause security problems with user interference and in many other areas.

In non-touch technology, the character input system is especially useful for user interfaces that do not require users to type on a keyboard, write on a trackpad/touchscreen, or input text to control the smart system in many applications. For this reason, a new research area has emerged that is related to the development of a non-touch character input system. Table 1 shows the results of some previous research on such systems. However, several problems with these systems have been noted. Schemes that use the finger alphabet [1,3,4], sign language [5–8], or dynamic hand gestures [9] as a character input method demand a relatively large amount of knowledge and practice before they can be used, and therefore have only a limited number of users. Hand gestures were also used as a paint tool box in [10]. The character input system of aerial handwriting [11–15] is an alternative that is not suitable for fast input, because it involves very large computations. In view of this, Shin et al. proposed a character input method based on hand tapping gestures to perform character input using English and Japanese hiragana characters, which can be used to facilitate interactivity between humans and computers [16]. Hand tapping gestures are used in a system for input using aerial virtual keypads, which can be effectively used as an alphabet. However, it is difficult for the user to remember the input characters represented by the hand tapping gestures, and it is also difficult to detect fingertips when the input of a hand tapping gesture overlaps the fingers.

In view of the above, this paper proposes a new character input system based on the gestural flick input method that considers the position, state and motion gestures of hands, and which can be used to facilitate human–computer interaction. In this study, we use the Kinect v2 sensor [17] to detect the color and depth of an image, body skeleton information, and hand motion gestures. Based on recognizing the position and the open or closed state of the hands, we developed a gestural flick input system that can be used to input characters, delete characters, add a space, insert a new line, and select a language. As a result, users can perform non-touch input under conditions that do not require touching any devices.

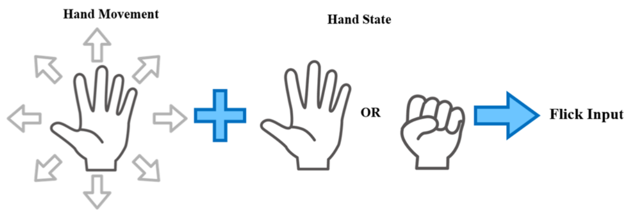


Fig. 1 General flow of the gestural flick input system

The general flow of the gestural flick input system is shown in Fig. 1.

This paper is organized as follows. In Sect. 2, we introduce our proposed model. Section 3 presents experimental results and a questionnaire survey about the usability of this system. In Sect. 4, we discuss our system based on the experimental results, and Sect. 5 concludes this paper.

2 Proposed model

This section describes the overall process of the proposed system, which allows non-touch character input based on captured information about the hand state (open or closed) and gestures via the flick input system. Figure 2 presents the overall procedure for the proposed system. The input information is acquired by the Kinect sensor and processed using our proposed model. We divide our system into two parts: (a) hand state and gesture recognition; and (b) the flick input system. Using the hand state and gesture recognition part, the system detects and recognizes the skeleton, the hand area with fingertips that expresses the state of the hand (open/closed), and various motion functions using hand gestures. The flick input system provides a way to input the character, and allows for modification of the size and the location of the flick keyboard. It allows gestural flick input to be performed without touching any device. Finally, the character entered via this system is considered as output.

2.1 Hand state and gesture recognition

2.1.1 Skeleton recognition

The Kinect sensor provides a method of communication between humans and electronic devices. It can collect image, voice, and depth data and is connected to a PC [18]. It can detect the user at a distance of 0.8–4.0 m and accepts as input the user's skeleton data and images in real time. The skeleton data are divided based on the users body into 25 coordinated positions, which are shown in Fig. 3, and use the head, left shoulder, right shoulder, left hand, and right hand as coordinates (labeled as 0, 3, 9, 6, and 12, respectively, in Fig. 3). Although it is possible to determine the location of

Table 1 Some experimental results of previous research

References	Gesture	Camera	Matching	Character	Recognition rate
[1]	Finger alphabet	Kinect	Template matching	Japanese (79 words)	Involving motion (approx. 76%) Not involving motion (approx. 69%) Sum total (approx. 73%)
[3]	Finger alphabet	Depth and image camera of infrared ToF	Template matching	Japanese (75 words)	Sum total (approx. 85%)
[4]	Finger alphabet	Depth camera	Template matching	Japanese (41 words)	Sum total (approx. 91%)
[5]	Sign language	Leap motion	BLSTM-NN classifiers	2240 sign gestures	Accuracy 63.57%
[6]	Sign language	RGB-D Kinect camera	Closing hand edge	French sign Alphabet	Sum total (approx. 93%)
[7]	Sign language	Magnetism sensor	Hidden Markov Model (HMM)	Sign language word (81 words)	50.27%
[8]	Sign language	Kinect	Hidden Markov model (HMM)	Sign language sentence	Approx. 86%
[9]	Hand gesture	XKin	Hidden Markov Model (HMM)	ASL Alphabet and dynamic gestures	90% (on 24 poses) 70% (on 16 gestures)
[10]	Hand gesture	RGB	Machine learning	6 hand gestures	96%
[11]	Handwriting	Accelerometer with a micro controller	k-nearest neighbor (k-NN)	Japanese (katakana) characters	Average recognition rate 79.6%
[12]	Handwriting	RGB camera	Template matching	Numeral character (10: 0 to 9)	91.9%
[13]	Handwriting	Wearable video camera	DP matching	Alphanumeric character (36 words)	75.5%
[14]	Handwriting	Pen installed on RGB camera	DP matching	Japanese (46 words)	87.2%
[15]	Handwriting	Kinect	Benchmarks	Digit, Alphabet, Symbol	96.5%
[16]	Finger alphabet	Kinect	Template matching	Japanese (hiragana) and English	94.3%

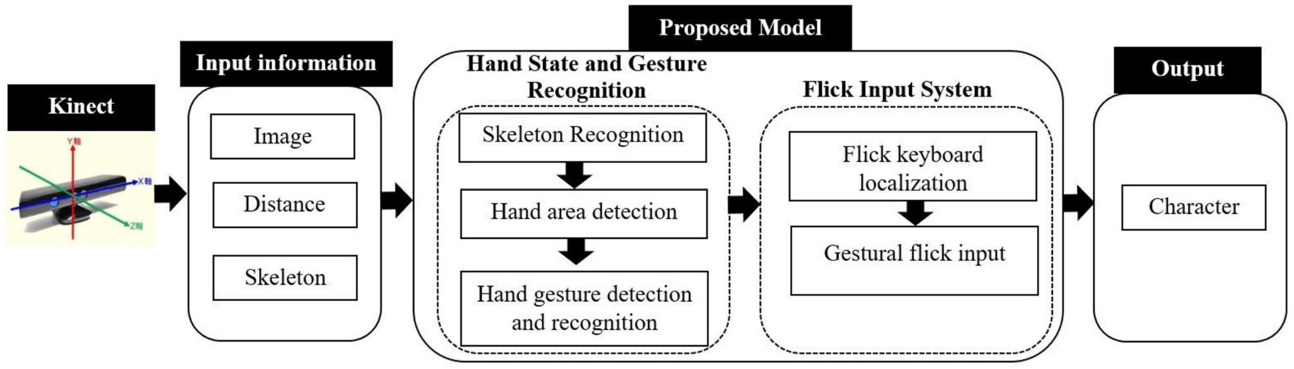


Fig. 2 Overall scheme for the proposed system

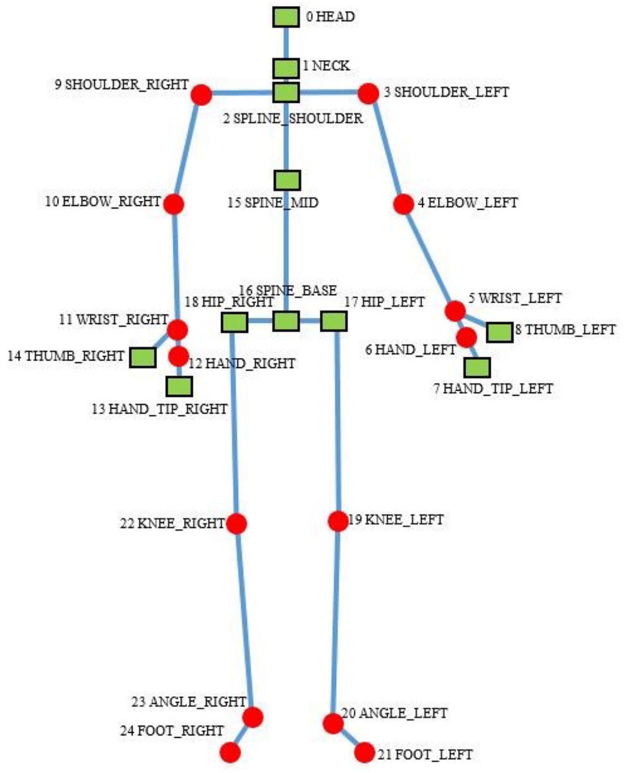


Fig. 3 Recognition of a skeleton by Kinect

both hands using the skeletal information, which is correlated with the head and shoulders of the user, and the position of the palm can therefore be determined, it is difficult to obtain the positions of the stretched fingers from the Kinect data. In the following subsection, we explain the way in which the proposed method effectively obtains the positions of the fingers by identifying the fingertips in user images.

2.1.2 Detection of the hand area

The system selects the hand area of the user image to find the palm position. To identify the actual area of the dominant hand in real-time images, a contour line can be used. The identification of the actual hand area may reduce the detection costs of finding the fingertips. The presence or absence of fingers in the hand area helps the system to determine the state (i.e., open or closed) of the dominant hand. During the detection of hand areas, the system initially creates a square in the XY-plane, which represents the location of the three-dimensional coordinates of the center of the dominant hand. In addition, the system filters out the unwanted area based on the Z-coordinate, where the Z-coordinate value of the other part is more than 10 cm from the center position of the dominant hand. This filtering process decreases the false detection rate for the hand area, which is caused by objects in the background (e.g., the face and chest). We set the length of the square as 36 cm using a trial and error process, since the average size of a Japanese persons hand is less than 20 cm [19]. Following this, the pixels in the square region of the XY-coordinates are extracted using the system, as shown in Fig. 4a. However, since a fingertip is a characteristic aspect of the contour line, we can identify the fingers of the dominant hand using a discriminatory inequality in the change of the curvature of the contour line of the dominant hand [20]. The steps taken to identify the fingertip from the mask image in the dominant hand area in order to recognize the open or closed state of the hand are as follows:

1. Identify the outline of the dominant hand. To do this, the system uses the FindContours function in OpenCV as features of the Suzuki85 algorithm. This function identifies the contour from a binary image [21] called the “mask image,” as shown in Fig. 4b. When the mask image of the dominant hand area has been detected, the outline and fingertip can also be detected. The position of the fingertip is at the extreme point from the hand center among each “long enough” convex part points of the effective hand outline. The distances (max_extrim) between the most extreme point at the top of the middle finger and the palm position are therefore calculated and are taken

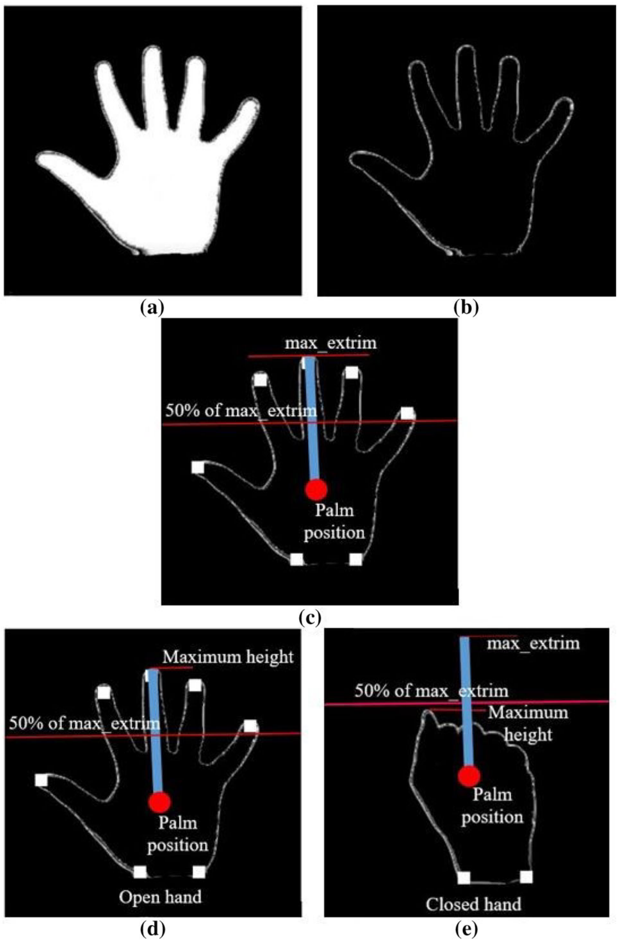


Fig. 4 Process of detection of the hand area and the open/closed hand state

as the maximum finger length of the open hand. To detect whether the hand is open or closed, we extend a straight line to the (50%) point of max_extrim, as depicted in Fig. 4c.

2. To detect whether the hand is open or closed, the distances of all the top extreme points are calculated from the palm position in both states (open and closed).
   (a) If the maximum distance is more than 50% of max_extrim, the system identifies the location of the candidate fingertip and determines that the hand is open. An open hand is shown in Fig. 4d.
   (b) Alternatively, if the maximum distance is less than 50% of max_extrim, the system identifies that the candidate fingertip does not exist and determines that the hand is closed. A closed hand is shown in Fig. 4e.

2.1.3 Detection and recognition of hand gestures

This subsection describes the recognition of different motion functions. In the proposed system, several text editing functions such as "delete," "add a space," "insert a new line," and "select language" are introduced, which are executed using hand gestures. Table 2 summarizes all of the motion functions considered in this paper. After confirming the modification of the size and location of the keyboard on the screen, the user can perform the motion functions. To execute these functions, the distances between the different positions of the skeleton are measured using a Euclidean distance, which is formulated as follows:

Table 2 Motion functions

Motion function	Description of hand gestures
Delete	Move the right hand from right to left
Add a space	Move the left hand from left to right
Insert a new line	Move both hands from top to bottom
Select language	Cross both hands

$$D(P, Q) = \sqrt{(q_x - p_x)^2 + (q_y - p_y)^2} \quad (1)$$

where p_x, p_y and q_x, q_y are the coordinate values of two points $P = (p_x, p_y)$ and $Q = (q_x, q_y)$.

The steps used to recognize the motion function from the mask image are as follows:

1. Calculate the distance of the palm position of the dominant hand from both shoulders. The distances from the right shoulder (shoulder_right) to the palm position and from the left shoulder (shoulder_left) to the palm position are expressed as SRP1 and SLP1, respectively, for the right hand. The distances from the shoulder_right to the palm position and from the shoulder_left to the palm position are expressed as SRP2 and SLP2, respectively, for the left hand. When SRP1 is greater than SLP1, the "delete" function is executed.
2. When SLP2 is greater than SRP2, the "add a space" function is executed.
3. Calculate the distances for the palm positions of both hands, from the shoulder position to the hip position horizontally. When the distance of the palm position (at the same time, and for both hands) is greater than that from the shoulder position to the hip position, the "insert a new line" function is performed.
4. When SRP1 is greater than SLP1 and SLP2 is simultaneously greater than SRP2, the select language function is performed. Users can select any language using this method.

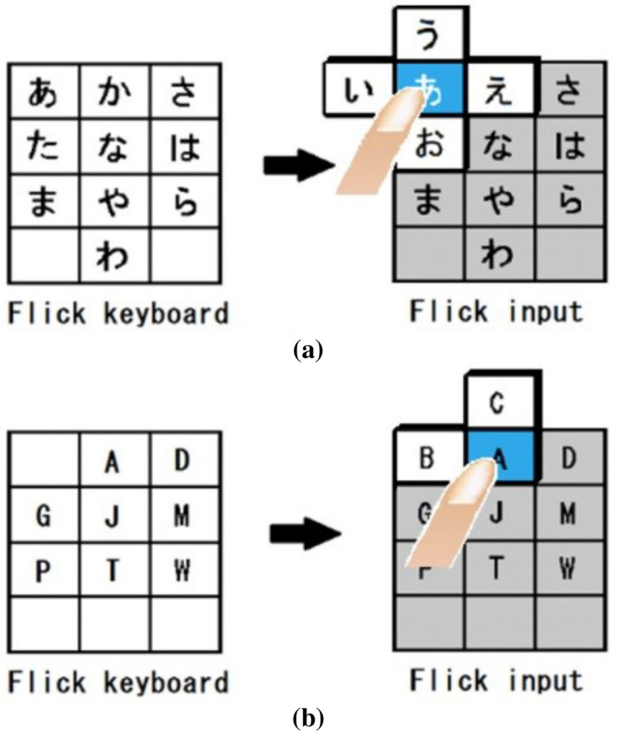


Fig. 5 The flick keyboard used in smartphones: **a** Japanese; and **b** English

2.2 Flick input system

The flick input system is a Japanese input method that is used on smartphones, meaning that many people are familiar with how it works. The system uses a motion gestural flick input as a character input method. These motion gestures are simple enough that a user can remember and perform them. Using the key arrangement of a numeric keypad mounted on the feature phone, the user selects a vowel in the direction of a consonant and flicks at the position of the tap to input characters. This is often used as a method of inputting characters on smartphones, since it allows users to enter characters quickly with simple operations, and we therefore decided to use flick input as the character input method of this system. Figure 5 shows a flick keyboard for inputting Japanese and English characters. In general, the layout used is a 12-key input. Each key, such as 'あ (a)', is associated with a particular sequence of characters, i.e., あ (a), い (i), う (u), え (e), お (o) and a, b, c for a. Instead of pressing a key repeatedly, users can swipe from a particular key direction to enter the desired character. However, this system uses the user's hand motion gesture to select the desired character.

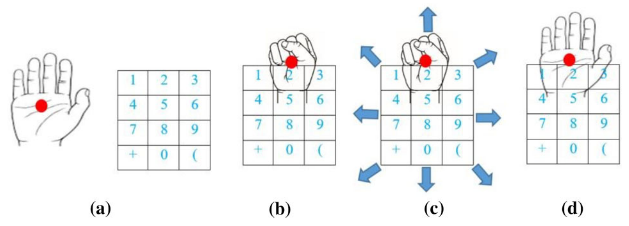


Fig. 6 Keyboard localization: **a** hand detection and keyboard; **b** keyboard selection using a closed hand; **c** changing the location of the keyboard by moving the closed hand; and **d** confirming the location of the keyboard by opening the closed hand

2.2.1 Localization of the flick keyboard

In this proposed system, the location and size of the keyboard can be changed dynamically, and it can input characters based on the position and the state of the hands on the virtual keyboard. Therefore, the character input can easily be performed independently of the standing point. The process of keyboard localization is described below and depicted in Fig. 6.

1. The user places the dominant hand outside of the keyboard and closes the hand to select the keyboard.
2. The user can change the location and size of the keyboard by moving the closed hand. The location of the keyboard can be changed by moving the dominant hand either vertically or horizontally. The upper part of the keyboard moves around following the user's dominant hand, and the size of the keyboard can be changed by moving the dominant hand back and forth. The keyboard becomes larger if the hand is moved forward, and decreases if it is moved backward.
3. The user opens the dominant hand to confirm the modification.

2.2.2 Gestural flick input

We use the traditional flick input layout for the proposed non-touch flick input system. Using the keyboard map shown in Fig. 7, input is performed using the 12 characters in the 4×3 matrix key map in the display screen. In the flick input system, each character is associated with a particular sequence of characters, and five characters can be mapped to simple flick movements. This system supports a keyboard map for three character sets: Japanese hiragana, the English alphabet, and numerals. We also include several special characters and operators in the English alphabet and a numerical key map.

The character sequences in the flick input system follow the order a, i, u, e, and o, as shown in Fig. 7a. For example, as shown in Fig. 7b, the flick input sequences for the Japanese hiragana characters in the first row and second column are か

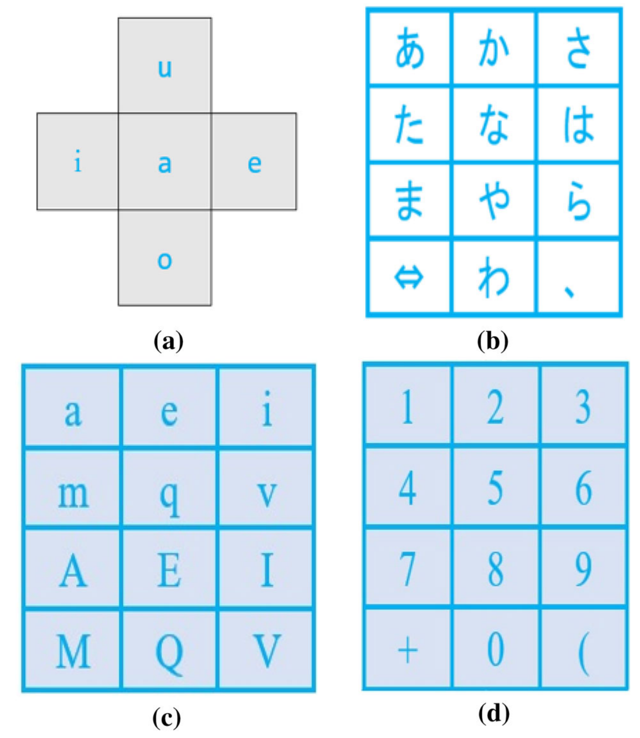


Fig. 7 Gestural flick input keyboard map: **a** flick input options; **b** input screen display for Japanese (hiragana) characters; **c** input screen display for the English alphabet; and **d** input screen display for English numeric characters

(ka), き (ki), く (ku), け (ke), and こ (ko) for か (ka). The flick input layout for all Japanese hiragana characters is shown in Fig. 8a, the English alphabet (both capital and small letters) with some special characters is shown in Fig. 8b, and the flick input sequence of numerals is expressed in Fig. 8c. By viewing the flick input keyboard map in the display screen, the user can input the desired character by carrying out a hand gesture to enter a desired character. The gestural flick input action for the character selection is illustrated in Fig. 9.

The character input system shown in Fig. 10 uses the following procedures:

1. The user holds a hand open over the sequence for the first character that they want to enter as input, and then closes the hand. The flick keyboard is shown in the display monitor.
2. The user selects a suitable character by moving the closed hand on the flick keyboard to the key that they want to enter as input, and then opens the hand.
3. The selected character is input.

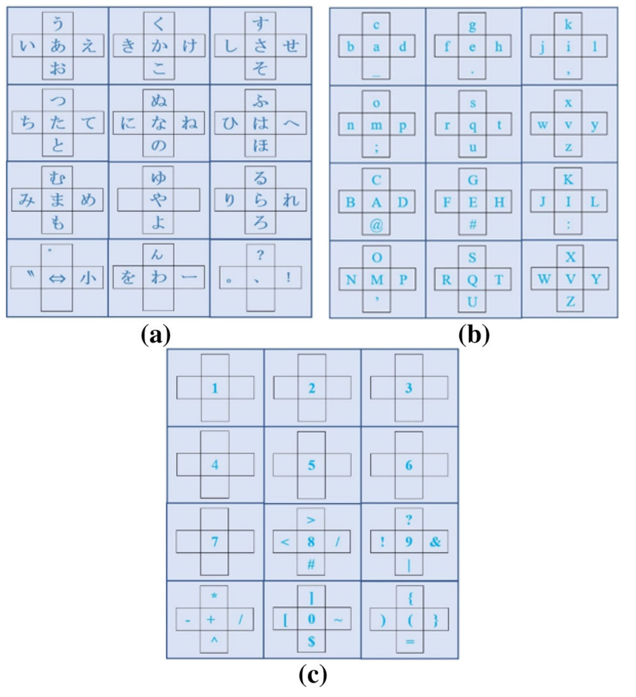


Fig. 8 Character sequences of the flick input map: **a** Japanese hiragana characters; **b** English alphabet with some special characters; and **c** numeric characters

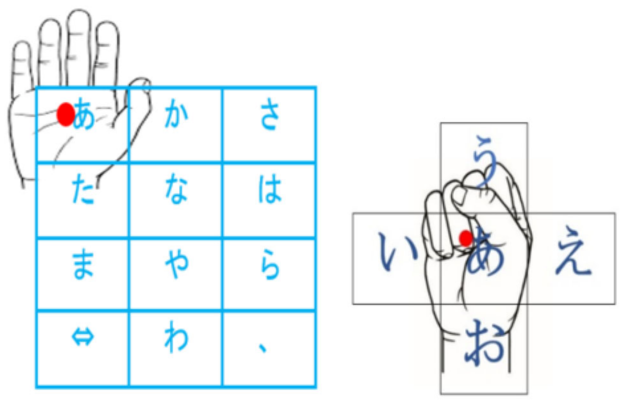


Fig. 9 The flick input action

3 Experiment, results, and evaluation

The experimental setup included a Kinect v2 sensor, a computer (Intel core i5-2400, 3.10 GHz) and a monitor. The system displays a flick keyboard on the monitor, which is impressed on the color image, and skeletal information is acquired using the Kinect v2 sensor. The user can therefore input characters by viewing the display screen, holding a closed hand over the first set of flick input characters on the display screen and moving the closed hand to select the desired character. The selected character is input by opening the hand. The character input screen is shown in Fig. 11. We conducted two experiments in this study, the first of which

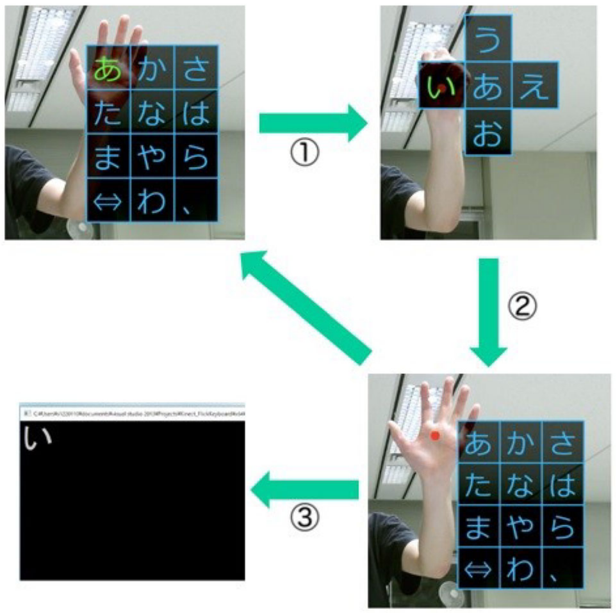


Fig. 10 Process of the gestural flick input system

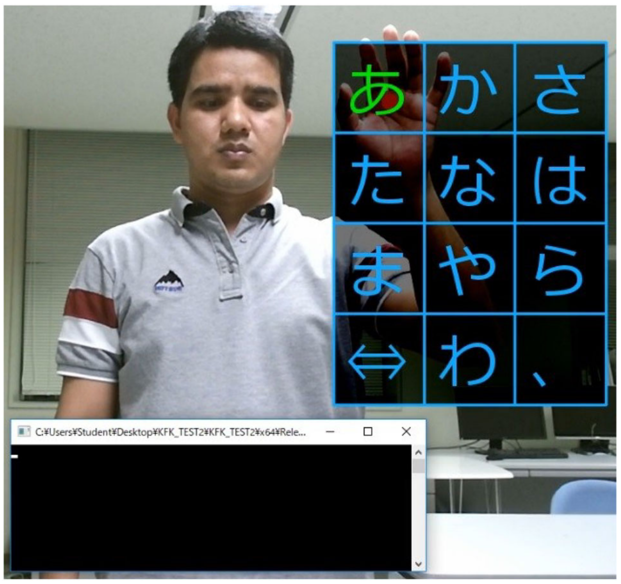


Fig. 11 Example of a non-touch character input display screen

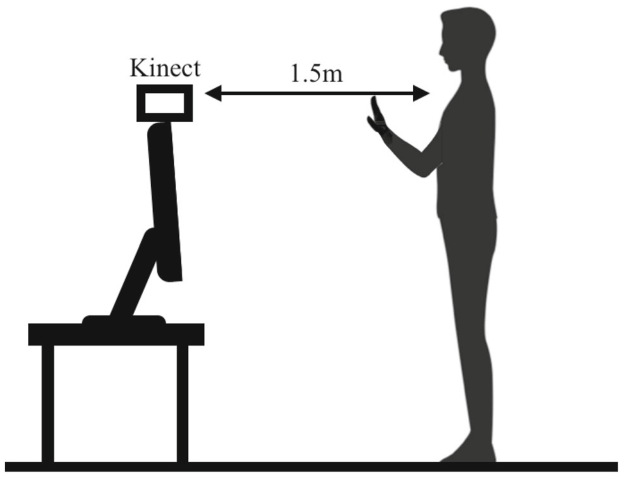


Fig. 12 The position of the Kinect

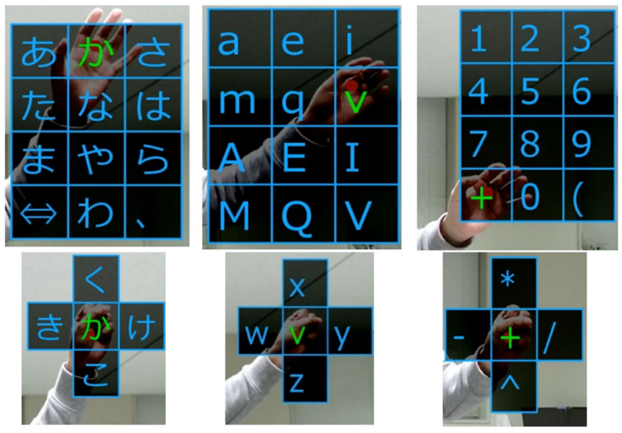


Fig. 13 Flick keyboard display screen in the proposed system

evaluated character selection and recognition and the speed of character input, and the second the recognition rate for the motion function gestures. This experiment was carried out in the experimental setting described in Fig. 12. The user was positioned 1.5 m from the Kinect sensor and entered each character as an input. When these experiments were complete, we conducted a questionnaire survey of 20 random participants on the usability of this system. The results of this survey are shown in Table 10.

In the first experiment, users input the Japanese sentence "あいづわかまつのあしたのてんきははれです," and the English sentence (with numeral characters) was Room no: 301 Pattern Processing Lab. The user performed this experiment five times. Characters were selected by viewing the display screen and entering the desired character as input. The system supports the flick input method on the display screen keyboard. The gestural flick input screen display of our system is shown in Fig. 13. Users can adjust the location and size of the keyboard by moving the dominant hand, to customize it to reflect the standing position, the size and the handedness of the user. The process of changing the location and size of the keyboard is illustrated in Fig. 14. One participant in the experiments with our gestural flick input system is shown in Fig. 15.

During these experiments, we evaluated the proposed system based on (a) the recognition rate and speed of character input; (b) the recognition rate of the motion functions; and (c) the user experience and usability testing.

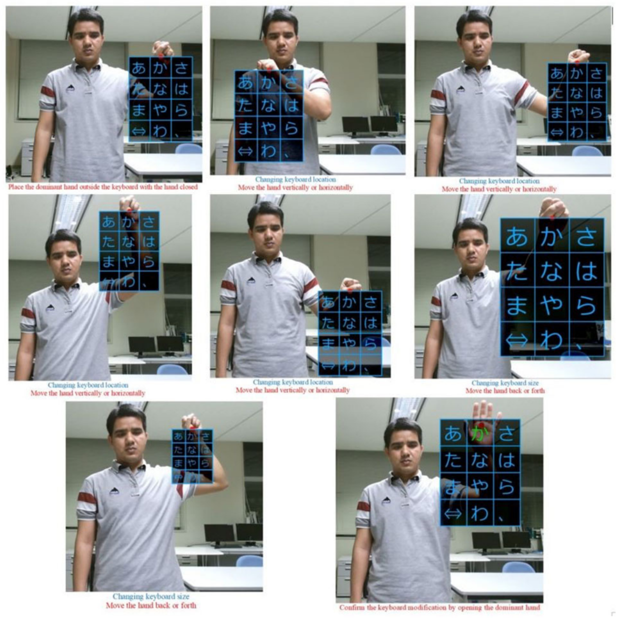
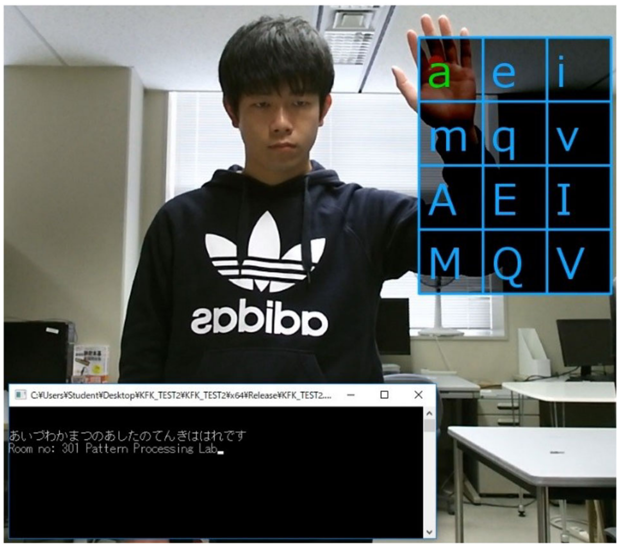

Place the dominant hand outside the keyboard with the hand closed
Changing keyboard location
Move the hand vertically or horizontally
Changing keyboard location
Move the hand vertically or horizontally
Changing keyboard location
Move the hand vertically or horizontally
Changing keyboard location
Move the hand vertically or horizontally
Changing keyboard size
Move the hand back or forth
Changing keyboard size
Move the hand back or forth
Confirm the keyboard modification by opening the dominant hand

Fig. 14 Changing the size and location of the gestural flick input keyboard of the system



```
C:¥Users¥Student¥Desktop¥KFK_TEST2¥KFK_TEST2¥x64¥Release¥KFK_TEST2...

あいづわかまつのあしたのてんきははれです
Room no: 301 Pattern Processing Lab_
```

Fig. 15 Simulation of character input with gestural flick input keyboard

3.1 Recognition rate of characters and speed of character input

Participants entered a total of 54 characters as an input, including Japanese, English, and numeric characters. They were asked to repeat the full character input experiment five times. The average accuracy of character selection and the recognition rate for all users are shown in Table 3. The system recognizes both correct and incorrect selections of user input characters as the user selects the input character using the flick keyboard. Several incorrect characters were selected for input based on the users performance of the hand gestures, and as a result, the system can evaluate the average recognition accuracy by measuring the correct input characters. The error rate of character selection and the speed of character selection over one minute are shown in Table 4. The average speed of character input for all users is shown in Fig. 16. Table 5 shows our experimental results for character-based selection and recognition and a comparison with some previous research results.

Table 3 Experimental results

User	Avg. character selection rate (%)	Avg. character recognition rate (%)
User 1	98.15	100
User 2	100	100
User 3	100	100
User 4	96.3	100
User 5	98.15	100
User 6	98.15	100
User 7	96.3	100
User 8	98.15	100
User 9	96.3	100
User 10	94.44	100
User 11	100	100
User 12	98.15	100
User 13	96.3	100
User 14	96.3	100
User 15	96.3	100
User 16	94.44	100
User 17	94.44	100
User 18	96.3	100
User 19	96.3	100
User 20	100	100
Average	97.22	100
Avg. recognition rate	98.61	

Table 4 Character selection accuracy using proposed system

No. of users	No. of characters	Avg. input speed of whole experiment (in s)	Input speed (characters/min)	Avg. error rate of character selection
20	54	112	29.0	2.78%

3.2 Recognition rate of motion functions

In the second experiment, we performed different functions such as "delete," "add a space," "insert a new line," and "select language (Japanese, English, or numeric) using the hand ges-

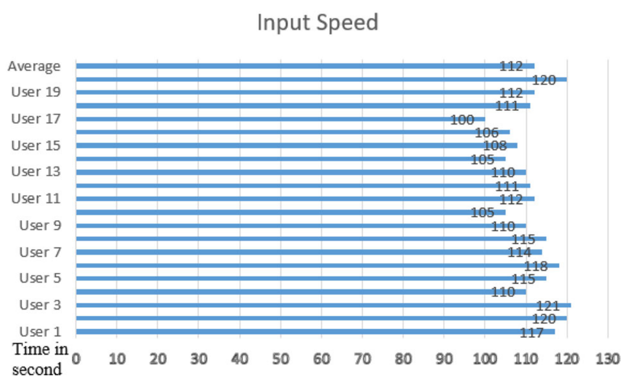


Fig. 16 Acquisition of the average speed of character input

tures shown in Table [2]. The gesture for a deleting character is executed by swiping the right hand from right to left, and the user can add a space character by swiping the left hand from left to right. A new line can be entered by moving both hands from top to bottom, and the language can be selected by crossing both hands. Figure [17] shows the different motion functions using hand gestures.

The users performed each gesture five times. The functions were executed by making hand movements that were recognized by the system. Users completed the “add a space” function without problems, and the recognition rate for this gesture was the highest (100%); however, the system reported several errors in the “insert a new line” function, for which the recognition rate was the lowest (95%). The overall average recognition rate for the motion functions in this system was 97.5%. The average recognition rates for all gestures are shown in Table [6].

3.3 User experience and usability testing

When the two experiments on this system were complete, we conducted a questionnaire survey on the usability of this system, using the System Usability Scale (SUS) [26], a very well-known usability perception system. This method is not generalized for specific applications, and we therefore used this method as a tool to evaluate the ease of use of our system. The questionnaires consisted of the 10 items shown in Table [7], and the evaluation criteria are shown in Table [8]. We evaluated each item based on a calculation of the SUS score, as this has proven to be a valuable evaluation tool that is

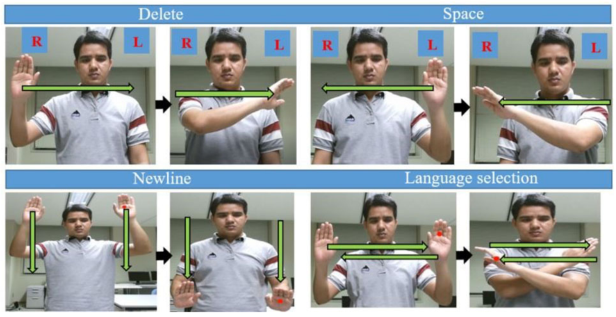


Fig. 17 The different motion functions using hand gestures

Table 6 Recognition rate of motion functions

Motion function	Avg. recognition rate (%)
Delete	97.0
Add a space	100.0
Insert a new line	95.0
Select a language	98.0
Average	97.5

robust and reliable. The scores for each item are in the range one to 10.

To calculate the SUS score, we subtract one from the user response score for every odd-numbered question and then multiply it by 2.5. For every even-numbered question, we subtract the user response score from five and multiply it by 2.5. We then calculate the average score for each question for all users. After summing the scores for all questions, we obtain the final SUS score. The scores for the odd- and even-numbered questions are calculated using Eqs. ([2]) and ([3]), with scores represented by QO and QE, respectively. Therefore, we measure the average score of QO and QE by following Eqs. ([4]) and ([5]). Finally, the SUS score is calculated according to the sum of the average scores for each questionnaire item for all users using Eq. ([6]).

$$QO_{ik} = (QO_iUSER_k - 1) \times 2.5 \tag{2}$$

$$QE_{jk} = (5 - QE_jUSER_k) \times 2.5 \tag{3}$$

$$QOAVG_i = \sum_{k=1}^{N} \frac{QO_{ik}}{N} \tag{4}$$

Table 5 Comparing the accuracy result with other related works

Reference	Functions	Recognition accuracy	Accuracy in our research
[22,23]	Motion gestures	Character- based (98.1%) Character- based (90.16%)	Character-based avg. 98.61% Motion function avg. 97.50%
[24,25]	Fingertip detection/ character input	Character input 90% Character input Avg. 97.12%	Character input 100%

$$QEAVG_j = \sum_{k=1}^{N} \frac{QE_{jk}}{N} \quad (5)$$

$$SUS_{score} = QOAVG_i + QEAVG_j \quad (6)$$

Table 7 Questionnaire items relating to SUS [26]

Q1	I think that I would like to use this system frequently
Q2	I found the system unnecessarily complex
Q3	I thought the system was easy to use
Q4	I think that I would need the support of a technical person to be able to use this system
Q5	I found that the various functions in this system were well integrated
Q6	I thought there was too much inconsistency in this system
Q7	I would imagine that most people would learn to use this system very quickly
Q8	I found the system very cumbersome to use
Q9	I felt very confident using the system
Q10	I need to learn a lot of things before I could get going with this system

Table 8 User responses to questionnaires items relating to SUS

Score	Content
5	Strongly agree
4	Agree
3	Neutral
2	Disagree
1	Strongly disagree

Table 10 Evaluation results (scores for SUS)

Question	Score
Q1	9.00
Q2	7.75
Q3	7.25
Q4	9.38
Q5	9.03
Q6	8.75
Q7	9.75
Q8	8.75
Q9	9.40
Q10	7.00
Total	86.10

where i = 1, 3, 5, 7, 9 and j = 2, 4, 6, 8, 10; N is the total number of users; QO_iUSER_k and QO_jUSER_k are the responses to the odd- and even-numbered questions, respectively. $QOAVG_i$ and $QEAVG_j$ represent the average score of odd- and even-numbered questions. Table 9 presents the example of SUS score calculation process based on the three user responses.

Based on the assessment criteria, we calculated all the evaluation scores shown in Table 10. We also compared the results with related flick input systems in terms of the character input speed and total error rate. A previous study has shown that the character entry speed for beginners using the No-look Flick input scheme [27] was 27.2 character per minute (CPM), and this rose to 35 CPM on smartwatches using BubbleFlick [28]. However, the character recognition error rates of these systems were 4.8% and 7%, respectively. These results indicate that in terms of character input, our system shows high performance, as it has a character error rate of 2.78% and an input speed of 29.0 CPM for Japanese hiragana, English characters, and numbers.

Table 9 Example of calculation process of SUS score

Questionnaire items	User response			Score calculation (using Eqs. 2 and 3)			Average score (using Eqs. 4 and 5)
	User 1	User 2	User 3	User 1	User 2	User 3	
Q1	5	5	5	10	10	10	10
Q2	3	2	1	5	7.5	10	7.5
Q3	3	4	5	5	7.5	10	7.5
Q4	2	1	2	7.5	10	7.5	8.33
Q5	4	5	5	7.5	10	10	9.17
Q6	1	2	1	10	7.5	10	9.17
Q7	5	5	5	10	10	10	10
Q8	3	1	2	5	10	7.5	7.5
Q9	4	5	5	7.5	10	10	9.17
Q10	2	2	1	7.5	7.5	10	8.33
Calculation of final SUS score using Eq. (6)							86.67

Table 11 Mean of SUS scores

Score	Content
92	Best imaginable
85	Excellent
72	Good
52	Ok/Fair
38	Poor
25	Worst imaginable

4 Discussion

Based on the experimental results, we find that the average recognition rate of our system is 98.61%, as shown in Table 3. The recognition rate of this system is higher than in the finger alphabet [1,3,4,16], sign language [5–8], and handwriting [11–15] functions, and the character recognition rate is 100%. However, some errors in character selection occurred in the system. The reason for the erroneous selection of character input may be that the position of the detected hand shifted between the closed and opened states. This causes incorrect key selection when the user selects a key that is near to a boundary. A possible way to solve this problem would be to introduce a gap between the keys. In this way, it would be possible to reduce the rate of incorrect key selection near the boundaries between the keys. The character selection rate could also be further improved. The proposed system achieves an error rate of 2.78% for character selection, as shown in Table 4. The overall average for the accuracy of character recognition was 98.61%, which is a meaningful improvement in character selection and recognition accuracy compared to the other related works presented in Table 5. From Table 4, it can be seen that the average speed of character input was 29.0 character/min; this is better than in [16] due to the use of gesture motions to input characters rather than finger tapping. In [16], the author used finger tapping for a non-touch character input system using an aerial virtual keyboard. However, the user was required to remember the input characters using hand tapping gestures, which made the system difficult. In order to develop a non-touch character input system that is easier and more acceptable to the user, our proposed system uses motion gestures. To produce the desired character, the user performs the character input using a gesture motion on the flick keyboard. The experimental evaluation of user responses of most questionnaire items shows that the system achieves the user satisfaction. However, according to the average score of Q3 and Q10, it may need to improve the user interface, so that the user can use this system in an easy way with minimum prior knowledge.

In Table 6, the average recognition rate of the motion functions is 97.5%. The system recognizes the motion function with the help of the proposed method. Since it has a high recognition rate, we can conclude that users can use motion functions easily.

In Table 10, the average SUS score was 86.10 out of 100, which is much higher than the average SUS score of 68 [29]. Based on the mean of the SUS scores (as shown in Table 11), we can confirm that this system is robust and easy to use. The scores for questionnaire items Q7 and Q9, which relate to the learning of the system, are 9.75 and 9.40 out of 10, respectively, as shown in Table 10. Since these results are very high, we can conclude that users can easily learn how to use this system within a relatively short time.

5 Conclusion

This paper proposes a non-touch character input system based on gestural flick input, using a Kinect sensor. The proposed model recognizes the position, state (open or closed) and movements of a hand entering a character and performing other motion function inputs. A dynamic flick keyboard was developed for Japanese hiragana, English, and numbers, via which the user can perform character input quickly and easily. Hand positions and movements are used to indicate the character of the flick keyboard for input, and the motion function is used for additional tasks in the character input system. Users can also adjust the size and location of the keyboard by hand motions. An experimental setup was established in a laboratory environment with 20 participants, and the results show that the system has a high level of precision, can quickly recognize input characters, and is easy to use. The proposed system is compared with other models based on several evaluation criteria and is shown to outperform alternative state-of-the-art approaches. The proposed system offers a natural and efficient non-touch interface technique that is specially designed to provide reliable performance for the most demanding environments and applications; this can reduce the gap between humans and computers, machines or robots and can resolve the underlying problems with an advanced input system, allowing the user to manage the system in a hygienic and safe way.

Compliance with ethical standards

Conflict of interest All authors declare that they have no conflict of interest.

References

1. Tanaka, S., Takuma, M., Tsukada, Y.: Recognition of finger alphabet remote image sensor (in Japanese). In: Proceedings of the 76th National Convention (IPSJ), vol. 76(2), pp. 2275–2276 (2014)

2. Khan, Z.H., Khalid, A., Iqbal, J.: Towards realizing robotic potential in future intelligent food manufacturing systems. Innov. Food Sci. Emerg. Technol. (2018)
3. Miyake, T., Wakatsuki, D., Naito, I.: A basic study on recognizing fingerspelling with hand movements by the use of depth image. Technol. Tech. Rep. Tsukuba Univ. **20**(1), 7–13 (2012)
4. Takabayashi, D., Ohkawa, Y., Setoyama, K., Tanaka, Y.: Training system for learning finger alphabets with feedback functions. The Institute of Electronics, Information and Communication Engineers, Technical Report of IEICE 112(483), HIP2012-90, pp. 79–84 (2013)
5. Kumar, P., Saini, R., Behera, S.K., Dogra, D.P., Roy. P.P.: Real-time recognition of sign language gestures and air-writing using leap motion. In: IEEE 15th IAPR International Conference on Machine Vision Applications (MVA), pp. 157–160 (2017)
6. Ben Jmaa, A., Mahdi, W., Ben Jemaa, Y., Ben Hamadou, A.: A new approach for hand gestures recognition based on depth map captured by RGB-D camera. Computación y Sistemas **20**(4), 709–721 (2016)
7. Maehatake, M., Nishida, M., Horiuchi, Y., Ichikawa, A.: A study on sign language recognition based on gesture components of position and movement (in Japanese). In: Proceedings of Workshop on Interactive Systems and Software (WISS), Japan, pp. 129–130 (2007)
8. Nishimura, Y., Imamura, D., Horiuchi, Y., Kawamoto, K., Shinozaki, T.: HMM sign language recognition using Kinect and particle filter. The Institute of Electronics, Information, and Communication Engineers, Pattern Recognition and Media Understanding (PRMU), vol. 111(430), pp. 161–166 (2012)
9. Pedersoli, F., Benini, S., Adami, N., Leonardi, R.: XKin: an open source framework for hand pose and gesture recognition using Kinect. Vis. Comput. **30**(10), 1107–1122 (2014)
10. Gajjar, V., Mavani, V., Gurnani, A.: Hand gesture real time paint tool-box: Machine learning approach. In: IEEE International Conference on Power, Control, Signals and Instrumentation Engineering (ICPCSI), pp. 856-860 (2017)
11. Kuramochi, K., Tsukamoto, K., Yanai, H. F.: Accuracy improvement of aerial handwritten katakana character recognition. In: IEEE 56th Annual Conference of the Society of Instrument and Control Engineers of Japan (SICE), pp. 116–119 (2017)
12. Ohkura, M., Manabe, R., Shimada, H., Shimada, Y.: A recognition algorithm of numerals written in the air by a finger-tip. IPSJ J. Inf. Process. (JIP) **52**(2), 910–916 (2011)
13. Sonoda, T., Muraoka, Y.: A letter input system based on handwriting gestures. IEICE, D-II J86-D-II, pp. 1015–1025 (2003)
14. Fujii, Y., Takezawa, M., Sanada, H., Watanabe, K. An aerial handwritten character input system (in Japanese). IPSJ (MBL), Tech. Rep. **50**(6), 1–4 (2009)
15. Kane, L., Khanna, P.: Vision-based mid-air unistroke character input using polar signatures. IEEE Trans. Hum.-Mach. Syst. **47**(6), 1077–1088 (2017)
16. Shin, J., Kim, C.M.: Non-touch character input system based on hand tapping gestures using Kinect sensor. IEEE Access **5**, 10496–10505 (2017)
17. Cai, Z., Han, J., Liu, L., Shao, L.: RGB-D datasets using microsoft Kinect or similar sensors: a survey. Multimed. Tools Appl. **76**(3), 4313–4315 (2016)
18. Zhang, Z.: Microsoft Kinect sensor and its effect. IEEE Multimed. **19**(2), 4–10 (2012)
19. Kawauchi, M.: Dimension Data of the Hands of the Japanese (2012). https://www.dh.aist.go.jp/database/hand/index.html
20. Nakamura, K.: Kinect for Windows SDK Programming C++. Shuwa System, Tokyo (2012). (in Japanese)
21. Suzuki, S., Abe, K.: Topological structural analysis of digital binary image by border following. J. Comput. Vis. Graph. Image Process **30**(1), 32–46 (1985)
22. Chen, M., AlRegib, G., Juang, B.H.: Air-writing recognition: Part I: Modeling and recognition of characters, words, and connecting motions. IEEE Trans. Hum.-Mach. Syst. **46**(3), 403–413 (2016)
23. Chen, M., AlRegib, G., Juang, B.H.: Air-writing recognition: Part II: detection and recognition of writing activity in continuous stream of motion data. IEEE Trans. Hum.-Mach. Syst. **46**(3), 436–444 (2016)
24. Ye, Z., Zhang, X., Jin, L., Feng, Z., Xu, S.: Finger-writing-in-the-air system using Kinect sensor. In: IEEE International Conference on Multimedia and Expo Workshops (ICMEW), pp. 1-4 (2013)
25. Zhang, X., Ye, Z., Jin, L., Feng, Z., Xu, S.: A new writing experience: finger writing in the air using a Kinect sensor. Multimed. IEEE **20**(4), 85–93 (2013)
26. Brooke, J.: SUS: a quick and dirty usability scale. Usability Eval. Ind. **189**, 4–7 (1996)
27. Fukatsu, Y., Shizuki, B., Tanaka, J.: No-look flick: Single-handed and eyes-free Japanese text input system on touch screens of mobile devices. In: Proceedings of the 15th International Conference on Human–Computer Interaction with Mobile Devices and Services. ACM, pp. 161–170 (2013)
28. Tojo, T., Kato, T., Yamamoto, S.: BubbleFlick: Investigating effective interface for Japanese text entry on smartwatches. In: Proceedings of the 20th International Conference on Human–Computer Interaction with Mobile Devices and Services. ACM, p. 44 (2018)
29. Sauro, J., Lewis, J.R.: Quantifying the user experience: Practical statistics for user research. Morgan Kaufmann, (Jul. 2016)

Md Abdur Rahim is currently working toward the Ph.D. degrees in the Pattern Processing Laboratory, Graduate School of Computer Science and Engineering, The University of Aizu, Fukushima, Japan. He is working (now on study leave) as an assistant professor, Department of Computer Science and Engineering, Pabna University of Science and Technology, Bangladesh. He received the Bachelor of Science (Honours) and Master of Science (M.Sc.) degrees in Computer Science and Engineering from University of Rajshahi, Bangladesh in 2008 and 2009. His research interests include human computer interaction, pattern recognition, computer vision and image processing, human recognition and machine intelligence.

Jungpil Shin received a Ph.D. in Computer Science and Communication Engineering from Kyushu University, Japan in 1999. He became an Associate Professor, Senior Associate Professor, and Professor in the School of Computer Science and Engineering, the University of Aizu, Japan, in 1999, 2004, and 2019, respectively. His research interests include pattern recognition, character recognition, image processing, and computer vision. He is currently researching the following advanced fields: penbased interaction system, real-time system, oriental character processing, computer education, human computer interaction, machine intelligence, and neurological disease analysis. He served several conferences as a program chair and program committee member for numerous international conferences and serves as a reviewer for several SCI major journals and an editor of journals. He is a member of IEEE, ACM, IEICE, IPSJ, KISS, and KIPS.

Md. Rashedul Islam received the B.S. degree in computer science and engineering from the University Rajshahi, Rajshahi, Bangladesh, in 2006, the M.S. degree in informatics from the Högskolan i Borås (University of Boras), Boras, Sweden, in 2011, and Ph.D. degree in electrical, electronic, and computer engineering at the University of Ulsan, Ulsan, South Korea, in 2016. He is currently working as a visiting researcher in University of Aizu, Japan also an Associate Professor (on study leave) in the Department of Computer Science and Engineering, University of Asia Pacific (UAP), Dhaka, Bangladesh. His research interests include Signal & Image processing, Machine learning, Feature selection and optimization, Human computer interaction, Data-driven bearing fault diagnosis and prognostics, and parallel algorithm/processing. He has several publications in major journals (SCI, SCIE) and conferences. He is reviewer of few SCIE journals. He has contributed to several conferences as organizing secretary, member of program committee, session chair and a reviewer.