Hindawi
Wireless Communications and Mobile Computing
Volume 2022, Article ID 6879206, 11 pages
https://doi.org/10.1155/2022/6879206

Research Article

Motion Gesture Delimiters for Smartwatch Interaction

Yiming Zhao,[1] **Yanchao Zhao**,[1] **Huawei Tu**,[2] **Qihan Huang**,[1] **Wenlai Zhao**,[1] **and Wenhao Jiang**[1]

[1]*College of Computer Science and Technology, Nanjing University of Aeronautics and Astronautics, Nanjing, 211100 Jiangsu, China*
[2]*Department of Computer Science and Information Technology, La Trobe University, Melbourne, Victoria 3086, Australia*

Correspondence should be addressed to Huawei Tu; h.tu@latrobe.edu.au

Received 25 January 2022; Revised 15 April 2022; Accepted 22 June 2022; Published 12 July 2022

Academic Editor: Laura Arjona

Smartwatches are increasingly popular in our daily lives. Motion gestures are a common way of interacting with smartwatches, e.g., users can make a movement in the air with their arm wearing the watch to trigger a specific command of the smartwatch. Motion gesture interaction can compensate for the small screen size of the smartwatch to some extent and enrich smartwatch-based interactions. An important aspect of motion gesture interaction lies in how to determine the start and end of a motion gesture. This paper is aimed at selecting gestures as suitable delimiters for motion gesture interaction with the smartwatch. We designed six gestures ("shaking wrist left and right," "shaking wrist up and down," "holding fist and opening," "turning wrist clockwise," "turning wrist anticlockwise," and "shaking wrist up") and conducted two experiments to compare the performance of these six gestures. Firstly, we used dynamic time warping (DTW) and feature extraction with KNN (K-nearest neighbors) to recognize these six gestures. The average recognition rate of the latter algorithm for the six gestures was higher than that of the former. And with the latter algorithm, the recognition rate for the first three of the six gestures was greater than 98%. According to experiment one, gesture 1 (shaking wrist left and right), gesture 2 (shaking wrist up and down), and gesture 3 (holding fist and opening) were selected as the candidate delimiters. In addition, we conducted a questionnaire data analysis and obtained the same conclusion. Then, we conducted the second experiment to investigate the performance of these three candidate gestures in daily scenes to obtain their misoperation rates. The misoperation rates of two candidate gestures ("shaking wrist left and right" and "shaking wrist up and down") were approximately 0, which were significantly lower than that of the third candidate gesture. Based on the above experimental results, gestures "shaking wrist left and right" and "shaking wrist up and down" are suitable as motion gesture delimiters for smartwatch interaction.

1. Introduction

Smartwatches have become a popular device in people's daily life [1]. People can use smartwatches in many day-to-day activities such as checking emails and sending and receiving messages [2]. Besides, smartwatches are also convenient for health management, e.g., sleep and heart rate monitoring [3, 4].

The questions of how to improve smartwatch interaction has attracted much attention in the HCI field. Currently, most popular commercial smartwatches such as Apple Watch still rely on touch interaction, physical buttons, and voice input [5]. These interaction methods are limited by screen size and environments, restricting the application of smartwatches to a wider extent. Therefore, smartwatches need new interaction methods to improve usability [6].

Motion gestures have potential advantages for smartwatch interaction [7]. For example, a user can draw a circle in the air with the wrist wearing a smartwatch to trigger a specific command of the smartwatch. Compared with interaction methods such as touchscreens, motion gesture interaction is less likely to be limited by the size of the screen. However, motion gesture interaction needs to address two main challenges. The first one is how to effectively obtain motion gesture data. Popular motion gesture recognition systems rely on cameras to capture gesture images or sensors such as gyroscopes and accelerometers to collect user action data [8–11]. Since smartwatches are mainly worn on the wrist and move along with the wrist, we can use in-built sensors of smartwatches to collect gesturing data. Compared to gesture images captured using a camera, sensor data requires fewer computational resources to collect and can be used to identify gesture delimiters more effectively. The second one is how to determine the start and end of an intended gesture [12]. In the process of motion gesture interaction, the smartwatch needs to continuously record movement data, both nonuser-intended (e.g., the wrist keeps swinging while walking) and user-intended (performing defined gestures). Therefore, we need to specify the start and end of the intended gesture. There are two common ways. First, the user clicks the button to determine the start and end of a motion gesture [13], which usually requires the nonwatch-wearing hand to perform the click. This could interrupt the interaction flow. Second, the user performs a defined gesture as a delimiter. The defined delimiter is used to distinguish the gestures that the user intends to input from unintended ones. The delimiter should be significantly different from the common actions and other gestures to avoid false recognition and should be simple enough to perform. We use delimiters to determine the start and end of a gesture, which allows for a more natural way of user interaction and requires no additional hardware than using buttons.

This study is aimed at selecting suitable motion gestures as delimiters for smartwatches to improve smartwatch interaction in low power consumption and natural way. We first selected six candidate gestures: "shaking wrist left and right," "shaking wrist up and down," "holding fist and opening," "turning wrist clockwise," "turning wrist anticlockwise," and "shaking wrist up" (Figure 1). Then, we conducted two experiments to evaluate the performance of these gestures as motion gesture delimiters. Considering the relatively low computing power of the watch and the requirement for fast and stable delimiter recognition, we used DTW (dynamic time warping) [14] and feature extraction with KNN (K-nearest neighbors) [15] to perform gesture recognition based on the data collected by the inbuilt gyroscopes and accelerometers of smartwatches. Results showed that "shaking wrist left and right," "shaking wrist up and down," and "holding fist and opening" achieved significantly higher recognition rates than "turning wrist clockwise," "turning wrist anticlockwise," and "shaking wrist up." In addition, we conducted a usability evaluation to support this conclusion. Hence, we further evaluated the performance of the former three gestures in daily scenes in terms of misoperation rates. "Shaking wrist left and right" and "shaking wrist up and down" had a misoperation rate of approximately 0, which could be primarily considered as the delimiters for smartwatch interaction.

2. Related Work

2.1. Motion Gesture Data Collection and Gesture Recognition. Motion gesture data collection for wearable devices usually relies on sensors. For example, EMG sensors [16, 17] or pressure sensors [18] can be used to collect data generated by hand movements. However, current smartwatches do not have such sensors. Instead, it is common to use inbuilt sensors such as

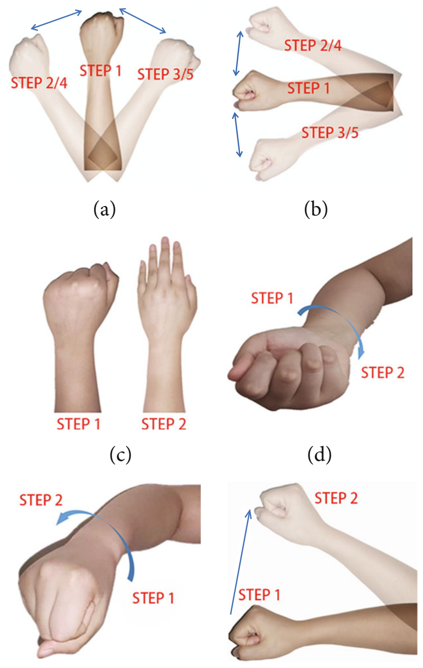


Figure 1: Six delimiter candidates. Gesture (a): "shaking wrist left and right"; gesture (b): "shaking wrist up and down"; gesture (c): "holding fist and opening"; gesture (d): "turning wrist clockwise"; gesture (e): "turning wrist anticlockwise"; and gesture (f): "shaking wrist up.".

accelerometers and gyroscopes to sense wrist movement and collect gesture data [19–21]. Our study also used these sensors for gesture data collection.

There are many methods for motion gesture recognition. Usually, feature extraction is carried out for gesture data, and neural network algorithms such as CNN, RNN, FNN, and HMM are trained for gesture recognition [22–26]. In our study, DTW (dynamic time warping) [14] and feature extraction + KNN(K-nearest neighbors) [15] are used for gesture recognition, respectively, as they need low sensor requirements and low computational requirements.

DTW can match two sequences of different lengths so that the minimum distance between the two sequences can be calculated. Then, the matching result can be compared based on this distance. The DTW algorithm has the advantages of short computation time. Sensors on smartwatches commonly collect gesture data at a fixed time interval. Two gesture data to be compared may have different lengths and cannot be matched directly. The DTW algorithm can be used to match data sequences of different lengths. And the distance and similarity between the two sequences of sample and template are calculated.

The feature extraction + KNN algorithm extracts some features of the whole set of gesture data and classifies the gesture data according to these features for gesture recognition. KNN calculates the distance between the sample X and the template samples and takes the top K template samples closest to it. If the top K samples have the most samples belonging to category R, then sample X also belongs to category R. K is usually an odd number not greater than 20.

Different K values generally lead to different classification results. Therefore, an optimal K value should be selected according to the results. The classification of template samples needs to be accurate as possible to ensure the correct classification of test samples.

2.2. Motion Gesture Interaction and Gesture Delimiters. Motion gestures are a promising way to interact with wearable devices [19, 27, 28]. Gesture interaction is especially suitable for mobile devices, such as changing the screen display direction by tilting the phone [29] and moving the cursor with gestures [30]. In addition, gesture interaction can also achieve more complex operations, such as text input with gestures on smartwatches [31], identity authentication by recognizing user gestures [32], and access data on virtual bookshelves around users [33].

An important step of using motion gestures is to separate normal smartwatch motion from a user's intended input. A common way to achieve it is to press button [13], but such a way requires both hands for interaction, which may not be always feasible. [34] collected IMU data to recognize three distinct phases of gesture entry: the start, middle, and end of a gesture motion for mobile devices. [35] used a dedicated delimiter sensor to detect the start and end of a gesture, which requires additional device support. [36] proposed a method for evaluating smartwatch delimiters using DTW, but using only accelerometers as data. Previous research has proposed to use double flip (i.e., rotating a smartphone along its long side to flip it twice) [12]. However, such a delimiter may not be applicable to wrist-worn devices, as it is in constant motion and therefore more error-prone. To find proper gesture delimiters for smartwatch interaction, our study considered six gesture candidates and examined their performance with two experiments.

3. Candidate Gesture Delimiters

Gesture delimiters applicable to smartwatch interaction should satisfy the following requirements:

(i) Easy to recognize: the smartwatch system needs to recognize gesture delimiters with high accuracy

(ii) Easy to learn: the user can learn gesture delimiters easily and recall them without much effort

(iii) Easy to perform: gesture delimiters would be performed frequently; so, they should have simple and should not lead to high hand and arm fatigue

To satisfy these requirements, six gestures were selected as candidate delimiters in our study. A pilot experiment with 6 right-handed participants was conducted to measure the average time of performing these gestures.

(i) Gesture 1: shaking wrist left and right. As shown in Figure 1(a), the user shakes the wrist twice with a small movement from side to side, and that the mean time of performing this gesture was 0.76 s

(ii) Gesture 2: shaking wrist up and down. As shown in Figure 1(b), the user slightly shakes his wrist twice from top to bottom. The average execution time of this gesture was 0.72 s

(iii) Gesture 3: holding fist and opening. As shown in Figure 1(c), the user clenches all fingers together and then opens them. The average execution time of this gesture was 0.55 s

(iv) Gesture 4: turning wrist clockwise. As shown in Figure 1(d), the user makes a fist and rotates the fist 90 degree clockwise. The average execution time was 1.13 s

(v) Gesture 5: turning wrist anticlockwise. As shown in Figure 1(e), the user makes a fist and rotates the fist 90 degree counterclockwise. The average time taken for gesture 5 was 1.10 s

(vi) Gesture 6: shaking wrist up. As shown in Figure 1(f), the user shakes the wrist upward significantly. Compared with the up-and-down shake of gesture 2, the movement range of gesture 6 is larger. The average time for performing gesture 6 was 0.53 s

Gestures 4 and 5 were designed based on the double flip gesture [12], which has been verified as a usable delimiter for mobile phone interaction. We would like to exam if gestures 4 and 5 would be feasible as a delimiter for smartwatch interaction.

4. Experiment One: Delimiter Recognition with DTW and Feature Extraction + KNN

We conducted an experiment to evaluate the effectiveness of the six delimiter candidates. The experiment consisted of two parts. First, we examined recognition rates of the six delimiters with DTW and feature extraction + KNN algorithms. Then, we evaluated the usability of the six delimiters according to subjective questionnaires.

4.1. Experimental Apparatus and Participants. The experiment was conducted with a Huawei Watch 2 smartwatch, which had a 1.2-inch round AMOLED display with a resolution of 390 × 390 pixels, a speed sensor, and a gyroscope sensor. The program was written in Python. In the experiment, sensor data of the smartwatch were recorded as gesture data, including the three axes of the acceleration and gyroscope sensors. Samples were recorded at 20 ms intervals.

There were a total of 10 participants (8 males) with an average age of 22.6 years. Each participant was asked to perform each gesture 30 times while wearing a smartwatch. Six of the students participated in the pilot study of gesture selection. Others did not have experience using smartwatches. All participants were right-handed and wore the smartwatch on their right hand. To obtain better results, we let the experimenter wear the watch with their dominant hand. Since the two hands are symmetrical, it should be reasonable to generalize the results from the right hand to the left hand.

In total, we collected 1800 gesture samples from 10 participants. We selected one sample per person per gesture for training, which means 60 samples for training and the remaining 1740 gestures for testing.

4.2. Data Preprocessing

4.2.1. Smoothing. Due to hand jitter and false operation of the user, the collected sensor data had a lot of noise. Figure 2(a) shows the raw data of gesture 1 collected from the x-axis of the accelerometer, which have many burrs and spikes. The burrs and spikes would reduce the accuracy and increase the difficulty of gesture segmentation and feature extraction. Smoothing filtering algorithms can reduce the noise.

There are many algorithms for smoothing filtering, e.g., moving average filter, median filter, and Gaussian filter. The algorithm of moving average filter was chosen in this study because it is relatively simple but effective. The moving average filter calculates the average value within a window and collects new data for each movement. The window slides forward, and the average value is calculated as the valid data. Figure 2(b) shows the data from Figure 2(a) after smoothing and filtering. It can be seen that after the moving average filter, the burrs and spikes in the data are effectively reduced.

4.2.2. Gesture Segmentation. In this experiment, we collected continuous three-axis acceleration sensors and three-axis gyroscope sensor data. The data collected by the sensors include unintended-gesture data and intended-gesture data. Gesture segmentation needs to extract valid gesture data from these data. Figure 3 shows part of the collected data of one trial of performing gesture 1. The data from 0 to 3, 4 to 6, and 7.5 to 8.5 s are not related to the delimiter, and the rest data is valid delimiter data. It can be seen that when the participant is performing the delimiter, the collected data is fluctuating significantly; when the participant is not performing the gesture, the collected data is fluctuating gently. So, we can rely on this feature to segment gestures.

We used differential methods to implement gesture segmentation. Differential methods can effectively show the volatility of data and have the advantages of easy implementation and running in real-time. A differential method is performed to obtain the total change data of two sensors, and then the start and end of the gesture are calculated by comparing the total change of two sensors with the preset threshold value. The main steps of gesture segmentation using the differential method are as follows.

Calculate the data variation: the formula for calculating the variation is as follows:

$$\Delta \mathbf{a}_k = |x_k - x_{k-1}| + |y_k - y_{k-1}| + |z_k - z_{k-1}|, \tag{1}$$

where x_k, y_k, and z_k represent the values of the sensor in the x, y, and z axis at the $k-$th data point, respectively. Since the data collected in this experiment come from two different sensors, it is necessary to calculate the variation of two sensors and add the two variations to obtain the total variation ΔA_k. To make the system more robust, we use the simple moving average algorithm (SMA) with a window size of 3 ($W = 3$) to smooth the gesture data. The mean of the k-th and subsequent $W - 1$ data points is denoted as SMA_k, which is the k-th data point after smoothing. SMA_k is calculated as Eq. (2).

$$\mathrm{SMA}_k = \frac{1}{W} \sum_{k}^{k+W} \Delta A_k. \tag{2}$$

Calculate the threshold: Figure 4 shows the data after differential processing in a trial. The results show the fluctuation of the data. The more intense the fluctuation, the more likely the data from valid gestures. For data lasting more than eighty seconds, the differential value fluctuations appear twenty times, and each fluctuation corresponds to the data variation of the gesture's six-axis data in Figure 3. To effectively identify the valid gesture interval, two thresholds need to be set: "Start" represents the start threshold, and "End" represents the end threshold. And to filter out the noise and the integrity of the gesture data, "Start" should be greater than "End."

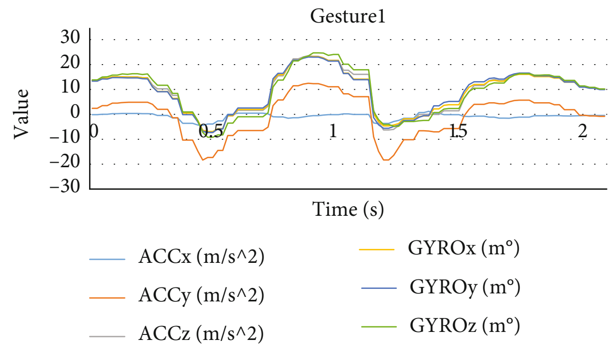
Since the fluctuation range of gestures is different, the thresholds of gestures are also different. The threshold values selected for gestures are calculated based on our experimental data, as shown in Table 1. For example, when we segment the differential data of gesture 1 in Figure 4, we first detect the differential value 0.9 as the beginning of the gesture, and then we mark the end of the gesture when the differential value drops to 0.6. Generally, the greater the fluctuation range of a gesture, the greater the data variation, the larger difference between the "Start" and "End" thresholds, and vice versa.

Result of segmentation is as follows: Figure 5(a) shows the fluctuation graph of a valid gesture after the segmentation of Figure 3. Figure 5(b) shows the fluctuation graph of a single valid gesture data for gesture 2. It can be seen that the trends of the sensor data for gesture 1 and 2 are different due to the different trajectories of gesture movement. Hence, we can perform gesture recognition based on the data characteristics.

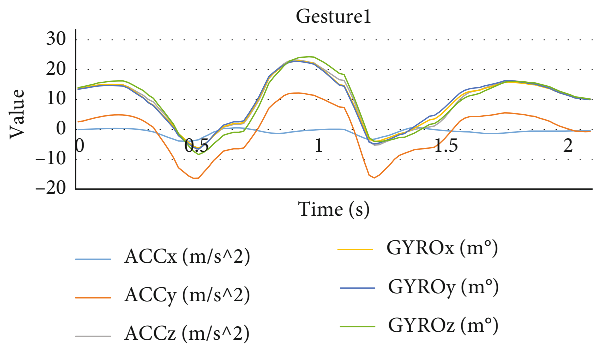
In addition to the data trend, features such as mean, variance, and peak-to-peak values can also reflect the differences in this data. Figure 6 shows the mean and variance between gestures 1 and 2 on different axes. The data for gestures 1 and 2 are very different, except for the average values over the acceleration z-axis. Hence, we can recognize gestures based on such feature differences.

4.3. Classification Methods. This study uses both DTW and feature extraction with KNN methods for gesture recognition. Although these are traditional methods, they are easy to implement and suitable for fast recognition of delimiters on smartwatches with low computational power. Future work will consider other algorithms to cater for other requirements of gesture interaction (e.g., higher recognition accuracy).

4.3.1. DTW. Dynamic time warping (DTW) is a simple recognition algorithm based on the idea of dynamic



(a) Before smoothing



(b) After smoothing

FIGURE 2: Gestures before and after smoothing.

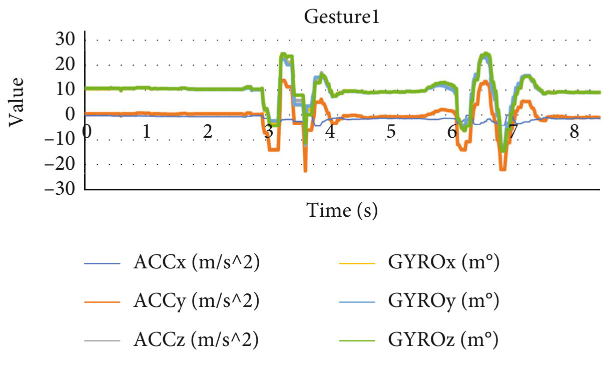


FIGURE 3: Part of the data collected in an experiment.

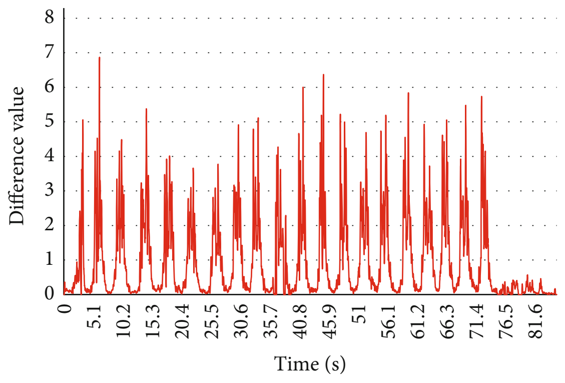


FIGURE 4: Differential processing result.

TABLE 1: Gesture threshold.

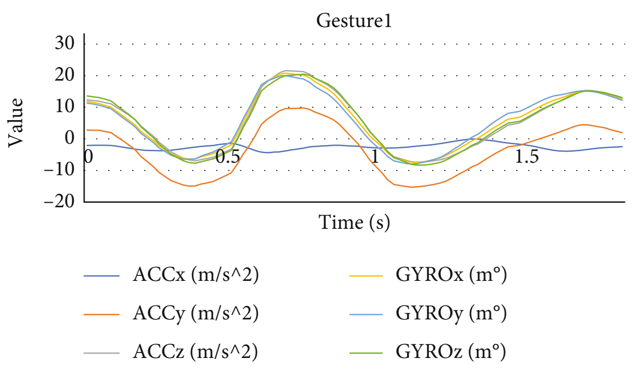
Gesture	Start	End
Gesture 1	0.9	0.6
Gesture 2	1.1	0.4
Gesture 3	0.56	0.1
Gesture 4	0.27	0.18
Gesture 5	0.2	0.1
Gesture 6	1	0.2

programming to solve the matching problem of two different length sequences. It calculates the minimum distance between two samples to match and recognize. DTW is widely used in speech recognition, which has the advantages of low computing time cost and few templates compared with other recognition algorithms. As with speech, the data generated by each person performing the gesture is different. Different performing time leads to different length data and not-strict linear correspondence. Direct matching of templates and samples of different lengths is not available; so, we use dynamic warping for the sequence to solve this problem.

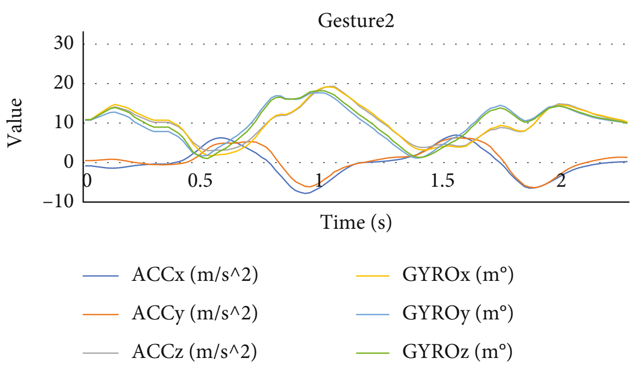
First, the system creates an $N \times M$ matrix D, where the number of rows N represents the number of frames of the sample sequence to be recognized, and the number of columns M represents the number of frames of the template sequence, i.e., the sample sequence to be identified is T $=[T_1, T_2 .... T_{n-1}, T_n]$, and the template sequence is R=$[R_1, R_2 ... R_{m-1}, R_m]$. T_n is the feature of the n-th frame with the frame length f. Similarly, R_m represents the feature of the m-th frame of the template with the frame length f. Since this experiment collects six-axis data from two sensors, the length of each frame is 6, i.e., $f = 6$, which represents the acceleration three-axis coordinate and gyroscope three-axis coordinate. D_{ij} represents the shortest distance between node i of T and node j of R. The D_{NM} is the shortest distance between two sequences.

Then, we calculate D_{ij}. Since the two sample sequences are not equal in length, we use the nonlinear matching method in DTW. As shown in Figure 7, we take T as the horizontal axis and R as the vertical axis and draw a grid diagram in the coordinate system. The intersection points in the grid represent the distance between the template at frame m and the sample to be identified at frame n. We find the shortest path from $(0, 0)$ to (N, M) based on the thought of dynamic programming. The point (i_n, j_m) is the optimal point decided by (i_{n-1}, j_m), (i_n, j_{m-1}), (i_{n-1}, j_{m-1}), and the best choice based on the previous section of the path. Thus, we have $D(i, j)$ as follows:

$$D(i_n, j_m) = \min \{D(i_{n-1}, j_m), D(i_n, j_{m-1}), D(i_{n-1}, j_{m-1})\} + d(i_n, j_m), \quad (3)$$



(a) Valid data for gesture 1



(b) Valid data for gesture 2

Figure 5: Valid data for gestures 1 and 2.

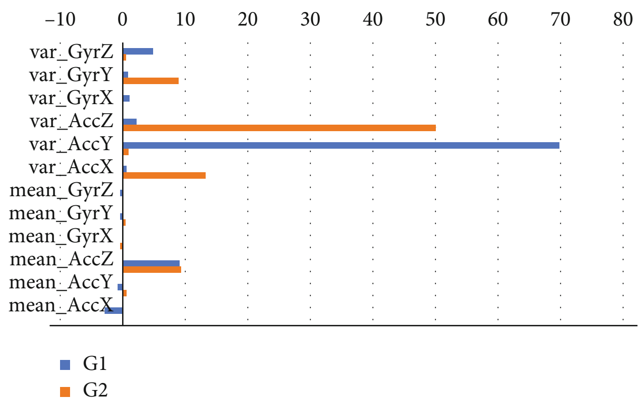


Figure 6: Variance and mean of gesture 1 and gesture 2.

where $d(i_n, j_m)$ is the distance between two sequences at (m, n). This experiment uses the 2-norm algorithm, the Euclidean distance, to calculate the distance between two vectors.

The short distance D_{NM} between the template R and the sample T indicates that the template R has a high similarity with T, which means they may come from the same gesture set. By calculating the shortest distance between the sample to be detected and multiple templates, we can get the best matching gesture based on the maximum similarity, i.e., the shortest matching distance. The algorithmic process of gesture recognition using DTW is shown in Figure 8.

4.3.2. K-Nearest Neighbor. The K-nearest neighbor algorithm (KNN) is a simple method in data mining, and its key idea is that if a sample has K nearest samples, most of which belong to class R, then the sample also belongs to class R. The selection of K has a significant impact on the overall classification result; so, an optimal K value should be selected based on comparative experiments, and K is usually an odd number no greater than 20.

In general, the label of the template data is known, but the label of the test sample is unknown. The system calculates distance (similarity) between the test samples and the templates by Euclidean distance and selects K nearest samples. Based on the class of K nearest samples, the system finds the most occurring class R, which is the label of the test sample. The algorithm described as follows:

(1) Calculating the distance between the test sample and the template data

(2) Sorting in ascending order by distance

(3) Selecting the K nearest samples

(4) Calculating the occurrence frequency of the class of K nearest samples

(5) According to the class and occurrence frequency of K nearest samples, the class R with the highest occurrence frequency is selected, which is the class of test sample

4.4. Results

4.4.1. Result Analysis of DTW. After the preprocessing step in Section 5.2, we used the DTW algorithm to perform gesture recognition. The DTW algorithm depends on the results of matching with templates. To eliminate recognition errors caused by inaccurate templates, we used 10 templates per gesture and took the average value. That is, there were 10 templates R for each gesture, and we need to calculate the distance of the test gesture T from these templates R and take the mean values of them as the final distance between the test gesture T and the training sample. The DTW algorithm is shown in Figure 7.

The DTW algorithm can easily recognize gestures, but it has a high computational cost. Although the time for DTW to recognize a single gesture data is short, it took significantly longer times when the amount of gesture data increases. In this experiment, 100 samples were collected for each gesture, and it took about 3 minutes to calculate the recognition result for each gesture data set. Although the approximate FastDTW algorithm can be considered, it reduces the running time at the expense of lowering recognition rate. In order to achieve a high recognition rate and short recognition time for a single gesture, the classic DTW algorithm was used in this study.

Figure 9 shows the recognition rate of each gesture. Since the motion range of gesture 3 is relatively slight, it is easily

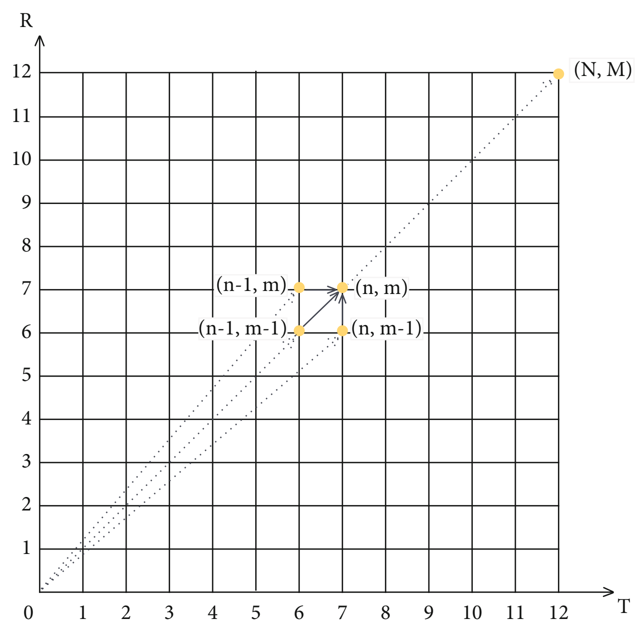


Figure 7: Nonlinear matching method in DTW.

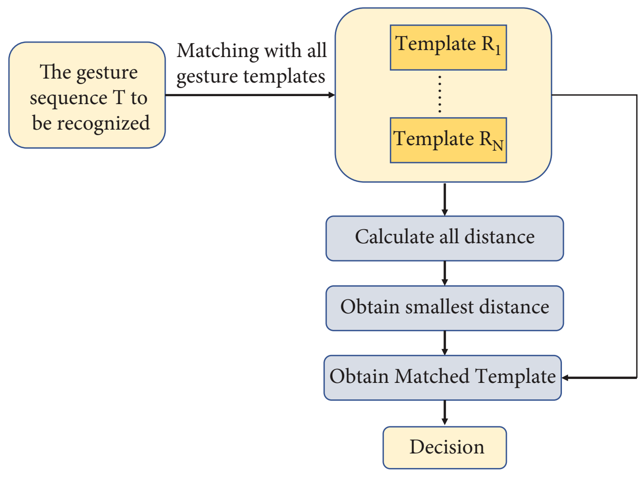


Figure 8: The process of DTW to recognize gestures.

confused with other gestures. After excluding gesture 3, Figure 10 shows the recognition rates of other gestures. It can be found that gesture 6 (shaking wrist up) and gesture 2 (shaking wrist up and down) are easily misidentified due to their high similarity in movement.

4.4.2. Result Analysis of Feature Extraction with KNN. After the preprocessing step in Section 5.2, we used the feature extraction + KNN algorithm for delimiter recognition. We needed to extract features from the data and then performed delimiter recognition based on the KNN algorithm.

We considered 30 features for our purpose. For x, y, and z axis of the accelerometer and gyroscope, we used the features of average value, variance, peak-to-peak, and interquartile. Besides, for the accelerometer and gyroscope, we used the correlation coefficient between x-axis and y-axis, between x-axis and z-axis, and between y-axis and z-axis. We then examined if using a subset of the 30 features could achieve similar recognition rates as using the 30 features. We employed the ExhaustiveSearch method provided on WEKA (Waikato Environment for Knowledge Analysis). This method finds a result with the highest recognition rate in the full set and all subsets. By combining it with evaluation strategies (CfsSubsetEval), we found that using all 30 features could obtain the highest recognition rate. Therefore, all the 30 features were adopted for further analysis with the KNN algorithm.

The K value of KNN has a significant impact on the experimental results, the estimation, and approximation errors. The K value is usually an small odd number to balance the estimation and approximation errors. The K value was set as 1, 3, 5, and 7 in this study. Table 2 shows the result of recognition rates for the six gestures with the four K values. After a comprehensive comparison, the gesture recognition rate with $K = 1$ should have the highest recognition accuracy. As shown in Figure 11, the recognition results of feature extraction with KNN are much better than the traditional DTW algorithm, e.g., the recognition rates of gestures 1, 2, and 3 reached 0.99. Overall, the recognition rates of gestures 4, 5, and 6 are lower than gestures 1, 2, and 3. We hence further look at the false recognition results of the three gestures.

Figure 12 shows that the false recognition results of gestures 4 and 5 are very similar. 86% of the false recognition results of gesture 4 were recognized as gesture 5, and 98% of the false recognition results of gesture 5 were recognized as gesture 4. By analyzing the motion trend of gestures 4 and 5, we found that the difference between gestures 4 and 5 only lies in the direction of rotation, which is weakly reflected in the data of x- and z-axes of gyroscope. The range of motion and the amount of change of different axes of the sensor are not greatly affected by the direction. Therefore, the recognition algorithm of feature extraction with KNN cannot distinguish well the differences between gestures 4 and 5. 84% of the false recognition results for gesture 6 were recognized as gesture 2. This is largely due to similar ranges of motion, i.e., gesture 2 was performed with a small jitter, and gesture 6 with too little range performed by the user was recognized as gesture 2. If the range of the upward fling of gesture 6 is defined with a threshold, then the fatigue of performing gesture 6 would increase and could not be suitable for some people.

Moreover, we deployed the algorithm on Huawei Watch 2 with Snapdragon Wear 2100 processor and tested the algorithm execution time. We used the time module in Java to check the time consumption of feature extraction and the KNN for the test set and then got 12.1 ms and 3.5 ms for every sample. The algorithm's low computing power requirements further contribute to the deployment and research of smartwatch delimiter research on mobile devices.

4.4.3. Impact of Gestures with Different Execution Times. We further discuss the influence of gestures with different execution times on recognition accuracy. Our system can reliably achieve a high recognition rate for gestures with different

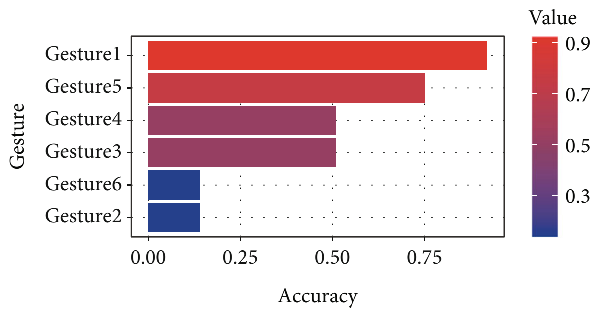


Figure 9: The accuracy of DTW.

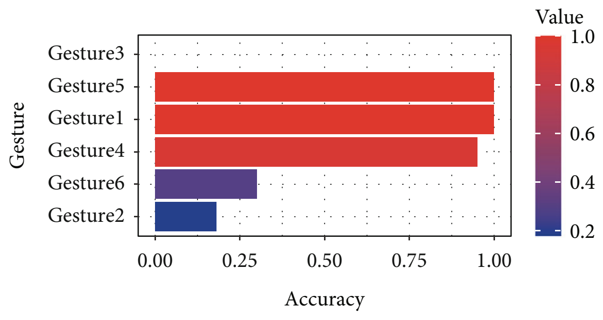


Figure 10: The accuracy of DTW after exclusion of gesture 3.

Table 2: The accuracy of several K.

	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6
$K = 1$	1	0.98	1	0.64	0.47	0.88
$K = 3$	1	0.99	0.99	0.64	0.46	0.79
$K = 5$	0.99	0.99	0.99	0.57	0.41	0.76
$K = 7$	0.99	0.97	0.99	0.42	0.56	0.73

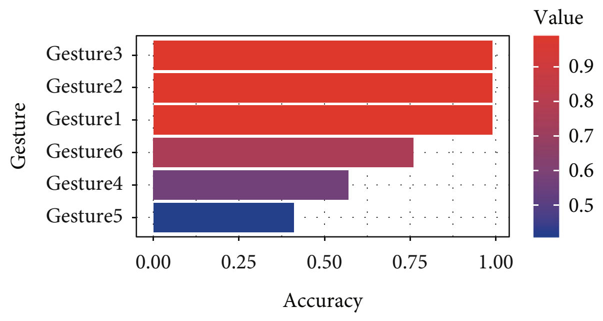


Figure 11: The accuracy of KNN when $K = 5$.

execution times from 0.4 s to 2 s. In the experiment, every participant performed six gestures with as fast as possible, standard, and as slow as possible speed. To test the accuracy of the KNN algorithm for gestures with different execution times, we collected data from another five participants (three males and two females) with the same setting of experiment one. Table 3 shows the accuracy for gestures with different execution times. We observe that the accuracy is similar for gestures with fast and standard execution times, while

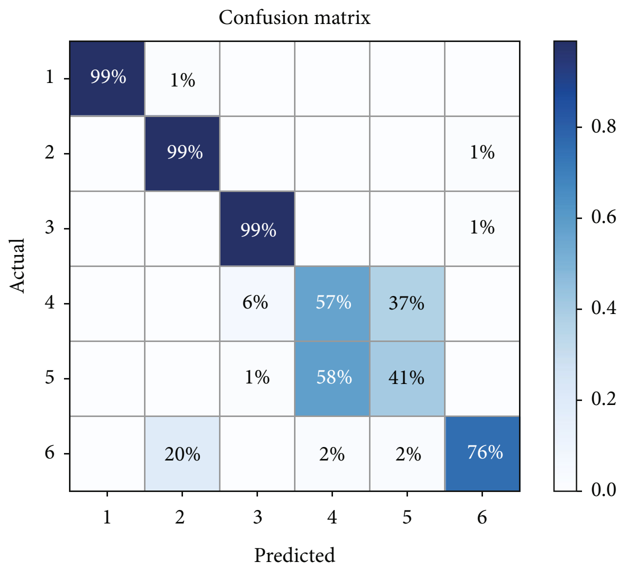


Figure 12: Confusion matrix graph generated using KNN when $K = 5$.

there is a slight decrease in accuracy at shorter execution times. The speed limit of performing the gestures may lead to this difference. The time difference between gestures with fast and standard execution speed is slight (from 0.1 to 0.3 s), leading to similar accuracy. However, the time difference between gestures with slow and standard execution speed is more significant (from 0.4 to 1 s), resulting in slightly lower accuracy. Generally, our system achieves a high recognition rate for gestures with different execution times, benefiting from the five features together.

4.5. Questionnaire Data Analysis

4.5.1. Questionnaire Settings. After completing the experiment task, each participant was asked to fill in a questionnaire to rate the six delimiters on 5-point Likert scales regarding "easy to learn," "easy to perform," "accurate to recognize," "avoid misoperation," and "suitable as the delimiter" (5 for the highest preference and 1 for the lowest preference). We created the questionnaire through an online website and then sent it to each participant through communication applications. Participants fill out the questionnaire via their smartphones or personal computers.

The questionnaire uses a 5-point Likert scale. The 5-point Likert scale has five options with five different scores according to the user's level of agreement, often with scores of 5, 4, 3, 2, and 1. The user chooses suitable options according to their degree of conformity to the declarative statements, and we calculate the total score according to the score assigned to each option of the scale for subsequent analysis. The declarative statements comprise unfavorable and favorable statements. In this experiment, the options indicate five different levels of strongly agree to disagree, and the scores are 5, 4, 3, 2, and 1 if it is a favorable statement and 1, 2, 3, 4, and 5 if it is an unfavorable statement. The gesture with the highest score represents the most

Table 3: Accuracy of feature extraction with KNN for gestures with different execution times.

Execution time	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6
Short	0.95	0.98	1	0.7	0.56	0.77
Standard	1	0.98	1	0.64	0.46	0.79
Long	1	0.88	1	0.6	0.45	0.75

suitable defining gesture from the user's subjective feeling perspective. In designing the questionnaire, it needs to set declarative statements in terms of fatigue, speed of performing gestures, guessability, gesture recognition accuracy, and user's subjective perception, as shown in Table 4.

4.5.2. Experimental Results. We used the chi-square statistics method to calculate the differences of delimiters in the measures. Regarding "easy to learn" and "easy to perform," participants generally regarded that the six delimiters were quite similar. These gestures were simple in form; so, participants thought they were all easy to learn and perform. However, in terms of "accurate to recognize," "avoid misoperation," and "suitable as the delimiter," the scores of gestures 1, 2, and 3 were significantly higher than other gestures (all $p < 0.05$), and there was no significant difference between the three gestures (all $p > 0.05$). In addition, in the questionnaire, participants who regarded gestures 1, 2, and 3 suitable for defining gestures account for 30%, 40%, and 30%, respectively. By combining the above results, the three gestures were selected as the suitable delimiters for further consideration.

Gesture 3 has the highest score, 40% of people thought it was easy to learn, and 60% of people thought it was suitable as a defined gesture. And the recognition rate of gesture 3 was 0.99 by the feature extraction with the KNN recognition algorithm; so, the gesture was considered as the best defined gesture.

The scores of gestures 1 and 2 are high, and the recognition rates of two gestures by the feature extraction with the KNN recognition algorithm are both 0.99. In the questionnaire survey, 20% of people thought that gesture 1 was suitable as the defining gesture, and 60% thought that gesture 2 was suitable as the defining gesture. In a comprehensive view, gesture 1 (shaking wrist right and left) and gesture 2 (shaking wrist up and down) can also be selected as the best defining gestures.

5. Experiment Two: Misoperation Rate of Delimiters

In this experiment, we aimed to investigate misoperation rate of delimiters in representative daily activities. According to experiment one, gesture 1 (shaking wrist left and right), gesture 2 (shaking wrist up and down), and gesture 3 (holding fist and opening) were selected as the candidate delimiters to be tested in this experiment.

5.1. Experimental Settings. The experimental equipment and participants were the same with experiment one.

5.2. Experimental Tasks. We evaluated misoperation rates of the three delimiters in three common scenes in our lives: walking, running, and standing up and sitting down. As a controlled study, we could not cover all daily activities. Instead, we selected three representative activities, that is, walking, running, and standing up and sitting down, to test the misoperation rate of three delimiters. The three scenarios can cover basic day-to-day activities. The data were collected from the participants, who wore the smartwatch to perform ten steps of walking, ten steps of running, and ten times of standing up and sitting down.

As experiment one, we processed the sensor data by the data preprocessing step and then recognized the gesture data to see whether participants accidentally performed gesture 1, 2, and 3 in the three scenes, so as to obtain the misoperation rate of gestures.

5.3. Experimental Results. According to experiment one, the feature extraction with KNN was faster and had better recognition performance than the DTW algorithm. Thus, the feature extraction with KNN was selected as the gesture recognition algorithm in this experiment. The misoperation rates are shown in Table 5.

The misoperation rate of gestures 1 and 2 is 0 in all scenarios. Gesture 3 had 0 in the running and walking scenarios, but 21% in the standing up and sitting down scenario. Therefore, in most scenarios, gestures 1 and 2 are more suitable as the delimiter than gesture 3.

6. Discussion

This study examined six gestures as delimiters for motion gesture interaction with smartwatches. We evaluated the performance of the six delimiters to select the proper ones. First, we used DTW and feature extraction with KNN to obtain gesture recognition accuracy for the delimiters. It is concluded that the feature extraction with KNN has a higher recognition rate for the gesture data, and its recognition rate for gestures 1, 2, and 3 exceeds 0.98. In addition, we checked he misoperation rate of gestures 1, 2, and 3 in three daily scenes. The misoperation rate of the three gestures in the scenarios of walking and running is 0. For standing up and sitting down, the misoperation rate of gestures 1 and 2 is 0, but the misoperation rate of gesture 3 is 21%. Therefore, gesture 1 (shaking wrist left and right) and gesture 2 (shaking wrist up and down) should be more suitable as delimiters for motion gesture interaction on smartwatches. Despite excluding gesture 3 as a delimiter, its excellent recognition accuracy and outstanding performance in the questionnaire still prove its importance as a motion gesture.

Table 4: Average rating of each gesture for each measure.

Measure	Gesture 1	Gesture 2	Gesture 3	Gesture 4	Gesture 5	Gesture 6
Easy to learn	4.3	4.2	4.4	4.2	4.3	4.2
Easy to perform	4.5	4.4	4.4	4.1	4.1	4.4
Accurate to recognize	4.5	4.5	4.1	3.1	3.1	3.3
Avoid misoperation	4.2	4.2	4.1	2.9	3.0	3.3
Suitable as delimiter	4.5	4.4	4.3	3.3	3.2	3.7

Table 5: Misoperation rate of gestures 1, 2, and 3.

Scenario	Gesture 1	Gesture 2	Gesture 3
Running	0	0	0
Walking	0	0	0
Standing up and sitting down	0	0	0.21

We can further improve our work in the following directions. First, for recognition algorithms, only two basic algorithms were used for gesture recognition in this paper. We need to test other algorithms, e.g., recognition algorithms based on the hidden Markov model. Second, for experimental design, this paper only designed six candidate gestures for experiments, and there may be other more suitable defined gestures. Third, participants can be selected from different ages, genders, and occupations. More user data and wider coverage of subjects help to draw more accurate conclusions. Fourth, for the obtained defined gestures, the experiments in this paper were conducted under lab conditions. Considering that the defined gestures are often used together with common action gestures, their practical applications in motion gesture interaction need to be further investigated. Finally, gesture data collection is susceptible to the environment. The sensors that collect the data also produce a certain amount of errors, and even the way the user wears smartwatches could affect collected data. Advances in wearable devices can mitigate the impact of these problems and improve the usability of gesture interaction, and the development of gesture interaction can also promote the progress and popularity of wearable devices.

7. Conclusion

This paper is aimed at selecting suitable gestures as the delimiter for smartwatch motion gesture interaction. To this end, this study firstly selected six candidate gestures (gesture 1: shaking wrist left and right; gesture 2: shaking wrist up and down; gesture 3: holding fist and opening; gesture 4: turning wrist clockwise; gesture 5: turning wrist anticlockwise; gesture 6: shaking wrist up). We conducted two experiments to evaluate the performance of the above six candidate delimiters. We used DTW and feature extraction with KNN to recognize these delimiters. Results showed that gestures 1, 2, and 3 achieved high recognition rate. In the second experiment, during the common scenes in our life, the misoperation rate of gestures 1 and 2 is 0, but the misoperation rate of gesture 3 is 21%. Therefore, gesture 1 and gesture 2 are suitable as motion gesture delimiters for smartwatch interaction.

Data Availability

We collected gesture data from ten participants, including eight men and two women, through sensors in smartwatches.

Additional Points

Research Highlights. (i) A study was conducted to investigate motion gesture delimiters for smartwatch interaction. (ii) “shaking wrist left and right” and “shaking wrist up and down” can serve as delimiters for motion gesture interaction with smartwatches. (iii) Feature extraction with KNN provided higher recognition accuracy than the DTW algorithm. (iv) The study provides insights into designing gesture-based delimiters for smartwatch interaction.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported in part by the National Key Research and Development Program of China under Grant (2019YFB2102000), in part by the National Natural Science Foundation of China under Grant (62172215), and in part by the Natural Science Foundation of Jiangsu Province (No. BK20200067).

References

[1] S. Pizza, B. Brown, D. McMillan, and A. Lampinen, “Smartwatch in vivo,” in *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, pp. 5456–5469, New York, 2016.

[2] D. Pradhan and N. Sujatmiko, *Can smartwatch help users save time by making processes efficient and easier, [M.S. thesis]*, vol. 18, University of Oslo, 2014.

[3] E. Årsand, M. Muzny, M. Bradway, J. Muzik, and G. Hartvigsen, “Performance of the first combined smartwatch and smartphone diabetes diary application study,” *Journal of Diabetes Science and Technology*, vol. 9, no. 3, pp. 556–563, 2015.

[4] D. J. Wile, R. Ranawaya, and Z. H. T. Kiss, “Smart watch accelerometry for analysis and diagnosis of tremor,” *Journal of Neuroscience Methods*, vol. 230, pp. 1–4, 2014.

[5] L. Heng, “Smartwatch interaction design research,” *Technology and Innovation*, vol. 8, p. 73, 2015.

[6] W. J. Hou and W. U. Chun-Jing, “Gestures interaction research based on the data analysis for smart watch,” *Packaging Engineering*, vol. 36, no. 22, pp. 13–16, 2015.

[7] C. Xu, P. H. Pathak, and P. Mohapatra, “Finger-writing with smartwatch: a case for finger and hand gesture recognition using smartwatch,” in *Proceedings of the 16th International Workshop on Mobile Computing Systems and Applications*, pp. 9–14, New York, 2015.

[8] L. Huang, F. Qiaobo, M. He, D. Jiang, and Z. Hao, “Detection algorithm of safety helmet wearing based on deep learning,” *Concurrency and Computation: Practice and Experience*, vol. 33, no. 13, article e6234, 2021.

[9] Z. Lu, X. Chen, Q. Li, X. Zhang, and P. A. Zhou, “A hand gesture recognition framework and wearable gesturebased interaction prototype for mobile devices,” *IEEE Transactions on Human-Machine Systems*, vol. 44, no. 2, pp. 293–299, 2014.

[10] Y. Weng, D. Ying Sun, B. T. Jiang, Y. Liu, J. Yun, and D. Zhou, “Enhancement of real-time grasp detection by cascaded deep convolutional neural networks,” *Concurrency and Computation: Practice and Experience*, vol. 33, no. 5, article e5976, 2021.

[11] P. Zhang and Z. S. Liu, “Gesture recognition method based on inertial sensor mpu6050,” *Transducer and Microsystem Technologies*, vol. 37, no. 1, pp. 46–53, 2018.

[12] J. Ruiz and Y. Li, “Doubleflip: a motion gesture delimiter for mobile interaction,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2717–2720, New York, 2011.

[13] S.-J. Cho, J. K. Oh, W.-C. Bang et al., “Magic wand: a handdrawn gesture input device in 3-d space with inertial sensors,” in *Ninth International Workshop on Frontiers in Handwriting Recognition*, pp. 106–111, IEEE, Kokubunji, Tokyo, Japan, 2004.

[14] H. Wang and Z. Li, “Accelerometer-based gesture recognition using dynamic time warping and sparse representation,” *Multimedia Tools and Applications*, vol. 75, no. 14, pp. 8637–8655, 2016.

[15] P. Hart, “The condensed nearest neighbor rule (corresp),” *IEEE Transactions on Information Theory*, vol. 14, no. 3, pp. 515-516, 1968.

[16] M. Hirota, A. Tsuboi, M. Yokoyama, and M. Yanagisawa, “Gesture recognition of air-tapping and its application to character input in vr space,” in *SIGGRAPH Asia 2018 Posters*, pp. 1-2, New York, 2018.

[17] X. Xie and Z. Liu, “Electromyography hand gesture recognition method based on dtw,” *Computer Engineering and Applications*, vol. 54, no. 5, pp. 132–137, 2018.

[18] J.-W. Lin and C. Wang, “Backhand: sensing hand gestures via back of the hand,” in *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology*, pp. 557–564, New York, 2015.

[19] S. Agarwal, A. Mondal, G. Joshi, and G. Gupta, “Gestglove: a wearable device with gesture based touchless interaction,” in *Proceedings of the 8th Augmented Human International Conference*, pp. 1–8, New York, 2017.

[20] M. Baglioni, E. Lecolinet, and Y. Guiard, “Jerktilts: using accelerometers for eight-choice selection on mobile devices,” in *Proceedings of the 13th international conference on multimodal interfaces*, pp. 121–128, New York, 2011.

[21] Y. Chen, P. Yang, and X. Chen, “A gesture recognition method based on acceleration feature extraction,” *Chinese Journal of Sensors and Actuators*, vol. 25, no. 8, pp. 1073–1078, 2012.

[22] H. Duan, Y. Sun, W. Cheng et al., “Gesture recognition based on multi-modal feature weight,” *Concurrency and Computation: Practice and Experience*, vol. 33, no. 5, article e5991, 2021.

[23] X. L. Guo, T. T. Yang, and Y. C. Zhang, “Gesture recognition based on kinect depth information,” *Journal of Northeast Dianli University*, vol. 36, pp. 90–94, 2016.

[24] S. Ji, W. Xu, M. Yang, and K. Yu, “3d convolutional neural networks for human action recognition,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 1, pp. 221–231, 2013.

[25] J.-S. Wang and F.-C. Chuang, “An accelerometer-based digital pen with a trajectory recognition algorithm for handwritten digit and gesture recognition,” *IEEE Transactions on Industrial Electronics*, vol. 59, no. 7, pp. 2998–3007, 2012.

[26] J. Y.-H. Ng, M. Hausknecht, S. Vijayanarasimhan, O. Vinyals, R. Monga, and G. Toderici, Eds., “Beyond short snippets: deep networks for video classification,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4694–4702, Boston, 2015.

[27] D. Ashbrook and T. Starner, “Magic: a motion gesture design tool,” in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, pp. 2159–2168, New York, 2010.

[28] J. Ruiz, L. Yang, and E. Lank, “User-defined motion gestures for mobile interaction,” in *Proceedings of the SIGCHI conference on human factors in computing systems*, pp. 197–206, New York, 2011.

[29] K. Hinckley, J. Pierce, M. Sinclair, and E. Horvitz, “Sensing techniques for mobile interaction,” in *Proceedings of the 13th annual ACM symposium on User interface software and technology*, pp. 91–100, New York, 2000.

[30] L. Weberg, T. Brange, and Å. W. Hansson, “A piece of butter on the pda display,” in *In CHI’01 Extended Abstracts on Human Factors in Computing Systems*, pp. 435-436, New York, 2001.

[31] K. Katsuragawa, J. R. Wallace, and E. Lank, “Gestural text input using a smartwatch,” in *Proceedings of the International Working Conference on Advanced Visual Interfaces*, pp. 220–223, New York, 2016.

[32] J. Liu, Z. Lin, J. Wickramasuriya, and V. Vasudevan, “User evaluation of lightweight user authentication with a single tri-axis accelerometer,” in *Proceedings of the 11th International Conference on Human- Computer Interaction with Mobile Devices and Services*, pp. 1–10, New York, 2009.

[33] F. C. Y. Li, D. Dearman, and K. N. Truong, “Virtual shelves: interactions with orientation aware devices,” in *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, pp. 125–128, New York, 2009.

[34] S. Kratz and M. Back, “Towards accurate automatic segmentation of imu-tracked motion gestures,” in *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems*, pp. 1337–1342, New York, 2015.

[35] J. Gong, X.-D. Yang, and P. Irani, “Wristwhirl: One-handed continuous smartwatch input using wrist gestures,” in *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, pp. 861–872, New York, 2016.

[36] F. Kerber, P. Schardt, and M. Löchtefeld, “Wristrotate: a personalized motion gesture delimiter for wristworn devices,” in *In Proceedings of the 14th international conference on mobile and ubiquitous multimedia*, pp. 218–222, New York, 2015.