Realtime Recognition of Dynamic Hand Gestures in Practical Applications

YI XIAO, TONG LIU, and YU HAN, Beijing Institute of Technology, China
YUE LIU, Beijing Institute of Technology, AICFVE of Beijing Film Academy, China
YONGTIAN WANG, Beijing Institute of Technology, China

Dynamic hand gesture acting as a semaphoric gesture is a practical and intuitive mid-air gesture interface. Nowadays benefiting from the development of deep convolutional networks, the gesture recognition has already achieved a high accuracy, however, when performing a dynamic hand gesture such as gestures of direction commands, some unintentional actions are easily misrecognized due to the similarity of the hand poses. This hinders the application of dynamic hand gestures and cannot be solved by just improving the accuracy of the applied algorithm on public datasets, thus it is necessary to study such problems from the perspective of human-computer interaction. In this article, two methods are proposed to avoid misrecognition by introducing activation delay and using asymmetric gesture design. First the temporal process of a dynamic hand gesture is decomposed and redefined, then a realtime dynamic hand gesture recognition system is built through a two-dimensional convolutional neural network. In order to investigate the influence of activation delay and asymmetric gesture design on system performance, a user study is conducted and experimental results show that the two proposed methods can effectively avoid misrecognition. The two methods proposed in this article can provide valuable guidance for researchers when designing realtime recognition system in practical applications.

CCS Concepts: • **Human-centered computing** → **User interface management systems**; **Gestural input;**

Additional Key Words and Phrases: Dynamic gesture recognition, activation delay, asymmetric gesture design, convolutional neural network, human-computer interaction

ACM Reference format:
Yi Xiao, Tong Liu, Yu Han, Yue Liu, and Yongtian Wang. 2023. Realtime Recognition of Dynamic Hand Gestures in Practical Applications. *ACM Trans. Multimedia Comput. Commun. Appl.* 20, 2, Article 50 (September 2023), 17 pages.
https://doi.org/10.1145/3561822

1 INTRODUCTION

Computers and intelligent devices such as mobile phones [16] and **virtual reality (VR)/ augmented reality (AR)** glasses are gradually becoming an essential part of our life.

Authors' addresses: Y. Xiao and Y. Wang, Beijing Institute of Technology, No. 5, South Street, Zhongguancun, Haidian District, Beijing, China, 100081; emails: yixiao0202@gmail.com, wyt@bit.edu.cn; T. Liu and Y. Han, Beijing Institute of Technology, Beijing, China; emails: lt_leonard@163.com, han.yu@outlook.com; Y. Liu (corresponding author), Beijing Institute of Technology, AICFVE of Beijing Film Academy, No. 5, South Street, Zhongguancun, Haidian District, Beijing, China and AICFVE of Beijing Film Academy, No. 4, Xitucheng Rd, Haidian, Beijing, China, 100088; email: liuyue@bit.edu.cn.

1551-6857/2023/09-ART50 $15.00
https://doi.org/10.1145/3561822

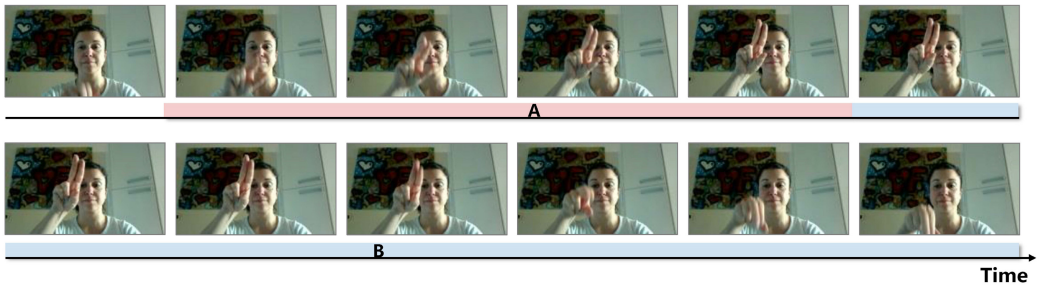

Fig. 1. "Sliding two fingers down": An example in the jester dataset.

Increasing applications of such intelligent devices makes it particularly important to design a simple and comfortable human-computer interface. Vision-based gestural interaction is a good choice with the development of computer vision technology. Compared with the traditional interaction methods such as the mouse and keyboard, gestural interaction is more natural, convenient, and intuitive [4]. In particular, as a non-touch, safe, and hygienic way of interaction, the mid-air hand gesture is becoming more important due to the coronavirus disease 2019 pandemic.

To recognize dynamic hand gestures, various methods have been proposed either based on hand skeleton key points or raw RGB data [8]. Benefiting from the rapid development of deep learning, the recognition of dynamic hand gestures has achieved a high accuracy. Take the Jester dataset [18] for an example, the accuracy of the state-of-the-art method has reached over 95% [24]. However, as a result of the temporal characteristics of dynamic hand gestures, the context information of time dimension must be considered in practical applications, which makes it prone to misrecognition. In particular, when performing dynamic gestures such as swiping left and swiping right, certain unintentional actions are easily misrecognized due to the similarity of hand poses. Figure 1 shows an example of "Sliding Two Fingers Down" in the Jester dataset [18]. When the user slides two fingers down, it is inevitable to encounter certain actions of sliding two fingers up as shown in Part *A* of Figure 1, which makes it prone to misrecognition. It is hard to solve such a problem simply by improving accuracy of the algorithm on the dataset and designing the recognition system from the perspective of human-computer interaction may be a good direction.

In this article, a realtime dynamic gesture recognition system with an RGB camera is built. The main contributions of this article are summarized as follows:

- A lightweight and realtime dynamic hand gesture recognition system based on RGB camera is built and evaluated in practical applications. To the best of our knowledge, this is the first work to study the application of dynamic hand gestures for continuous gesture recognition using just an RGB camera.
- The temporal process of a dynamic hand gesture is decomposed and redefined, and two methods are proposed to avoid misrecognition by introducing activation delay and using asymmetric gesture design.
- A user study is conducted to investigate the influence of introducing activation delay and using asymmetric gesture design on system performance. Experimental results show that the two proposed methods can effectively avoid misrecognition. The design considerations for practical applications are also summarized.

The rest of this article is organized as follows: Section 2 introduces related works. Section 3 describes the dynamic hand gesture recognition system. Section 4 evaluates accuracy and realtime performance of the system. Section 5 evaluates the two proposed methods through a user study. Section 6 presents the experimental results. Section 7 shows the discussions about the work. Finally,

Section 8 draws a conclusion of this work and presents several design considerations for practical applications.

2 RELATED WORK

This section first introduces the gestural interaction as well as the related works, then summarizes the progress of deep learning based gesture recognition method.

2.1 Gestural Interaction

Gestures of gestural interaction can be classified as manipulative gestures and semaphoric gestures [25]. Manipulative gesture is a common interface applied in AR/VR [2], which is defined in Quek et al.'s work [25] as "those whose intended purpose is to control some entity by applying a tight relationship between the actual movements of the gesturing hand/arm with the entity being manipulated". It is essential for manipulative gestures to obtain the key points of hand skeleton, which is computationally intensive and needs extra devices. Methods such as using a depth camera [22], glove-based wearable devices [31], and even RGB cameras [35] have been proposed to obtain the hand skeleton. The main challenge when applying manipulative gestures is the lack of feedback during interaction [25]. Many researches have been conducted such as using electrical feedback, force feedback, and so on. For example, Zhao et al. [37] explored the touch-based interaction using electrovibration haptic feedback to enhance the system interactivity and improve user experience in virtual environments.

In contrast, there are no such constraints in semaphoric gestures because semaphoric gestures convey semantic meanings through predefined hand gestures. However, semaphoric gestures are subject to individual and cultural differences [19]. To solve such issues, many researches have been conducted. Nielsen et al. proposed a procedure to develop intuitive and ergonomic gestures from a user-centered viewpoint [20]. Wobbrock et al. presented an approach to design tabletop gestures that relies on eliciting gestures from non-technical users [32]. Similar works are also conducted in other scenarios such as driving [33], handheld objects [26], smart rings [29], microgestures [6]. For example, Xiao et al. [33] presented a set of micro-gestures on the steering wheel which are designed for intuitively commanding the in-car information system. Sharma et al. [26] conducted a user elicitation study of microgestures that are performed while the user is holding an object.

It should be noted that gestures in datasets such as Jester dataset [18] and EgoGesture dataset [36] belong to semaphoric gestures, thus all gestures of this work refer to semaphoric gestures.

2.2 Deep Learning Based Gesture Recognition

Various deep learning based methods [9] have been proposed for gesture recognition and can be generally classified into **two-dimensional convolutional neural network (2D CNN)** based methods and **three-dimensional convolutional neural network (3D CNN)** based methods. In view of the success of deep convolutional networks in image classification, researchers began to apply it in gesture recognition [13]. The biggest challenge for 2D CNN based gesture recognition methods is to capture motion information from still frames. Karen et al. [27] proposed a two-stream convolutional network architecture that incorporated spatial and temporal networks, in which color data and dense optical flow were used. Wang et al. [30] proposed the **Temporal Segment Networks (TSN)** and they divided a video into several snippets, in which each snippet would produce its own preliminary prediction of the gesture classes and a consensus among the snippets would be finally derived as the video-level prediction. However, all these methods capture temporal information with the help of handcrafted features such as optical flow. To capture temporal relationships from raw data, Lin et al. [15] proposed the **Temporal Shift Module (TSM)**, in

Fig. 2. Temporal process of a dynamic hand gesture ("Sliding Two Fingers Up").

which part of the channels along the temporal dimension were shifted, which means that it could achieve the performance of 3D CNN but maintain 2D CNN's complexity.

In contrast, 3D CNN based methods can model the time dimension directly. Ji et al. [11] attempted to develop a 3D CNN model for gesture recognition, which extracted features from both the spatial and the temporal dimensions by performing 3D convolutions. Tran et al. [28] proposed a **Convolutional 3D (C3D)** model and investigated the best performing architectures for 3D CNN. Recently, various 3D CNN based methods are proposed such as Inflated 3D ConvNet (I3D) [5], Pseudo-3D Residual Net (P3D ResNet) [23], SlowFast Networks [7], and so on. 3D CNN based methods can achieve temporal modeling well but are computationally intensive, whereas 2D CNN based methods are computationally cheap but are difficult to model the temporal relationship. To build a lightweight and realtime system, a 2D CNN network with TSM is adopted in this article.

3 GESTURE RECOGNITION SYSTEM

This section presents the details of the proposed system. First, the dynamic hand gesture is modeled in the time dimension, which decomposes and redefines the temporal process of a dynamic hand gesture. Then the difficulties and corresponding solutions of building a realtime dynamic hand gesture recognition system are proposed. Afterwards, the two proposed methods of the activation delay and asymmetric gesture design are introduced and the network as well as the whole pipeline of the system are presented.

3.1 Temporal Modeling of Dynamic Hand Gestures

Pavlovic et al. [21] divided the temporal process of a dynamic gesture into a preparation phase, a nucleus phase, and a retraction phase. In order to better utilize the temporal characteristics of gestures, in this article, the temporal process of a dynamic gesture is further divided into Part I (t_a-t_b), Part II (t_b-t_c), Part III (t_c-t_d), and Part IV (t_d-t_e), as shown in Figure 2, which are continuous frames of a dynamic hand gesture recorded by an RGB camera. The realtime recognition results of each frame are shown in Figure 2.

- *Part I* corresponds to the preparation phase such as Part *A* shown in Figure 1, which consists of a preparatory movement that sets the hand in motion from some resting position.
- *Part II and Part III* correspond to the nucleus phase. Because a dynamic gesture is an action over a period of time, the recognition system can only perform classification after detecting gesture actions for a certain period of time. Thus, Part II is defined as the least video frames needed for the system to recognize a dynamic hand gesture.

- *Part IV* corresponds to the retraction phase and the hand either returns to the resting position or repositions for a new gesture in this phase.

3.2 Challenges and Solutions

To build a realtime gesture recognition system, there are several challenges as follows:

- *How to Determine the Start and End of a Dynamic Hand Gesture.* Since the dynamic hand gesture is an action that changes over time as shown in Figure 2, there is no indication that clearly defines its start and end, which implies that there is no need to accurately locate the exact time of the gesture's start and end. In this work, a time window is adopted to slide on the video stream to determine whether the gesture occurs, and the start and end of the gesture are determined by the system recognition results, such as the time points t_c and t_d shown in Figure 2.
- *How to Reduce Memory and Power Consumption.* Generally, the time window slides over the video stream in a very small stride such as 2 in the system, which is computationally intensive. Some works use a two-model hierarchical architecture [14], in which a lightweight model is used for detection and a heavy model is used for classification. In order to reduce memory and power consumption, a lightweight 2D CNN model is adopted in the system, which is computationally cheap and has a good performance on gesture recognition.
- *How to Recognize a Hand Gesture Only Once.* The dynamic hand gesture is a temporal process as shown in Figure 2, various number of frames belong to the same gesture. In order to activate the command only once for a gesture, gestures will not be recognized during the current time window after the command is activated. In considering that the lengths of dynamic gestures are generally less than 1 second, the time window in the system is set at about 0.67 seconds, which has been proven to be effective and stable in practical test.
- *How to Avoid Misrecognition.* Misrecognition is a critical issue for the system performance, thus two methods are proposed by introducing activation delay and using asymmetric gesture design. The details are introduced in Section 3.3.

3.3 Activation Delay and Asymmetric Gesture Design

In order to avoid misrecognition, the following two methods are proposed in the system:

- **Activation delay.** After decomposing a dynamic gesture as shown in Figure 2, a deeper understanding of the temporal process of a dynamic hand gesture can be obtained . However, when the dynamic hand gesture is used as an interactive interface, it is hard to determine the stage in a dynamic hand gesture process to activate the corresponding command. In the proposed method, the time point when the system recognizes the gesture is taken as the origin, such as the time point t_c shown in Figure 2, and then the command is activated after some frames, which is called the *activation delay*. Details about the activation delay are introduced in Algorithm 1.
- **Asymmetric gesture design.** Previous analysis shows that dynamic hand gestures especially for gestures of direction commands are prone to misrecognition because of the similarity of hand poses of different gestures. Thus, asymmetric gesture design is proposed, i.e., different hand poses for direction commands in the opposite directions are utilized. For example, the "sliding two fingers right" can be used as the right command while the "swiping left" can be used as the left command. Details about the asymmetric gesture design are introduced in Section 4.1.

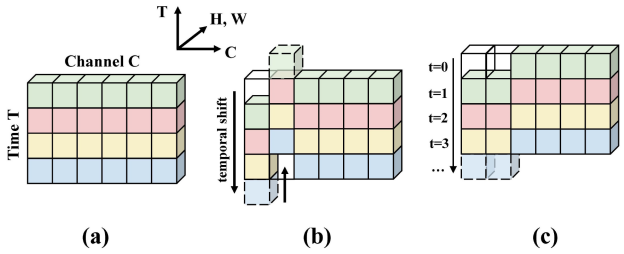

Fig. 3. Temporal shift module [15]: (a) Original tensor without temporal shift. (b) Offline temporal shift. (c) Online temporal shift.

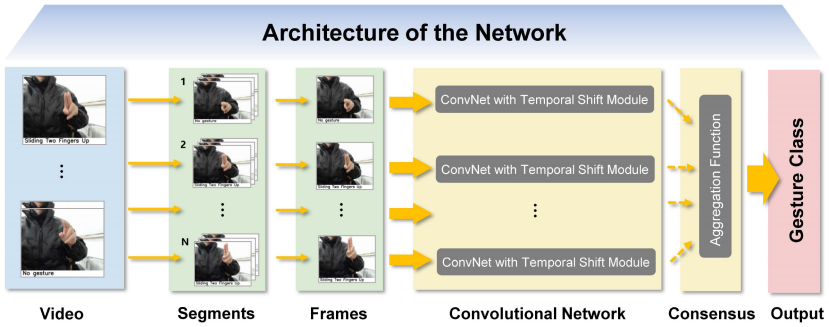

Fig. 4. Architecture of the network used in system.

3.4 Architecture of the Network Used in the System

In order to build a lightweight system, a 2D CNN network with online temporal shift module (TSM) is adopted. By shifting part of the channels along the temporal dimension, TSM can help capture temporal relationships and can be inserted into 2D CNNs to achieve temporal modeling at zero computation and zero parameters [15]. TSM can be divided into offline temporal shift and online temporal shift. Figure 3(a) shows the original tensor without shift. Offline temporal shift is designed for offline action recognition, which mingles both past and future frames with the current frame as shown in Figure 3(b). For an online video stream, since the future frames cannot be obtained in advance, the online temporal shift is designed, which mingles only the past frame with the current frame as shown in Figure 3(c).

The whole 2D CNN based network for dynamic hand gesture recognition is shown in Figure 4. First, a video is divided into N segments of equal durations. Then, one video frame is randomly sampled from its corresponding segment and each video frame is input to the 2D CNN network with TSM insertion. MobileNet-V2 [17] is adopted here to make the model lightweight. Finally, the class scores of different video frames are fused by an averaging aggregation function.

The online TSM framework [15] is used and pipeline of the proposed system is shown in Algorithm 1. In the system, video frames are obtained from the video stream by continuously sliding a time window. s indicates the sliding stride and 2 is adopted in the system. n_{delay} indicates the number of video frames of activation delay. It should be noted that the subscript "delay" indicates that the system with such variables adopts the activation delay. *Scores* indicates the classification scores of each video frame. $Buffer$ indicates the list that stores *Scores* of N historical frames. When the system gets a frame from the video stream, the new video frame is input to the classification model and the classification scores are obtained. Here MobileNet-V2 is adopted in the online network. Then, the classification scores are stored in $Buffer$ and the average

classification scores in $Buffer$ are calculated. $Scores_{total}$ indicates the average value of classification scores stored in $Buffer$. If the maximum value in $Scores_{total}$ is greater than 0.8, the index of gesture class is equal to the index of the maximum value in $Scores_{total}$. Finally, if the gesture class of the output keeps the same in n_{delay} frames, corresponding semaphore is activated and no gesture is recognized during this period.

ALGORITHM 1: Pipeline of the online system

Input:
$Video$: video stream;
$Frame$: current frame;
s: stride of the time window;
n_{delay}: video frames of activation delay;
Buffer: a list that stores historical frames' *Scores*;
Output:
$Gesture$: gesture classification of current frame;
activation of corresponding semaphore;

```
while true do
    update the current frame from video stream every s frames;
    Scores = model(Frame);
    append Scores into Buffer;
    Scores_total = Average(Buffer[−N :]);
    if (Max(Scores_total) ≥ 0.8) then
        Gesture = Argmax(Scores_total);
    if (Gesture keeps the same in n_delay frames) then
        activate corresponding semaphore;
        stop recognizing new gestures until the current gesture ends;
```

4 EXPERIMENTS ON OFFLINE MODEL AND REALTIME PERFORMANCE

This section first introduces the evaluation of the model used in the system, then discusses the realtime performance of the system.

4.1 Evaluation of the Offline Model

In order to obtain a model that can be used in practical applications, the model is trained on a large-scale public dataset, i.e., the Jester dataset. Details about the dataset, experimental settings, experimental results and analysis are presented below.

The Jester Dataset. The Jester dataset [18] is a large-scale, real-world dataset for dynamic gesture recognition. It includes 148,092 video gesture clips, which are split into train, validation and test set in the ratio of 8:1:1. To the best of our knowledge, the Jester dataset is the largest dynamic gesture dataset to date. Gestures of direction commands are common in such dynamic gesture datasets as the Jester dataset. Considering that the Jester dataset is a large-scale and challenging video dataset of human gestures [18], gestures used in the system are selected from this dataset, which can help to train a robust model in practical applications.

Experimental Settings. The implementation of this network is based on the public PyTorch platform. The training and testing bed is Ubuntu 20.04 system with Nvidia GeForce 3090 graphics cards. During training, mini-batch stochastic gradient descent (SGD) is adopted with batch size 64, momentum 0.9 and weight decay 0.0005. The initial learning rate is 0.001 and is decayed by 10 every 50 epochs.

Table 1. Comparison to the State-of-the-Art on the Jester Dataset

Model	top-1 (%)	top-5 (%)	FLOPs (G)	Params (M)
I3D [5]	92.57	99.61	108.0	12.0
STM [12]	96.70	99.90	66.5	22.4
SlowFast-ResNet50 [7]	93.36	99.55	36.1	34.5
PAN-ResNet101 [34]	97.40	99.90	251.7	-
LIGAR [10]	95.43	99.45	4.7	4.5
TRN-Multiscale [38]	95.31	-	16.0	18.3
C3D [14]	94.14	90.57	71.8	65.5
ResNeXt-101 [14]	96.99	93.75	13.9	47.6
TSM-ResNet50-offline	95.77	99.76	65.9	23.6
TSM-MobileNet-V2-offline	94.97	99.74	5.1	2.3
TSM-MobileNet-V2-online	95.01	99.66	5.1	2.3

"-" indicates the numbers are not available for us.

Exprerimental Results. To verify that the model used in the system is lightweight and has a high accuracy, the TSM model is compared with some competitive models in gesture recognition. For each model, Table 1 reports the top-1 and top-5 accuracy on the Jester dataset, floating point operations (FLOPs) and parameters (Params). Experimental results show that TSM model has achieved a high level of accuracy, with the highest accuracy of TSM-ResNet50 reaching 95.77%. It can be found that I3D [5], SpatioTemporal and Motion Encoding (STM) [12], SlowFast [7], Persistent Appearance Network (PAN) [34], Temporal Relation Network (TRN) [38] and TSM-ResNet50 all have a large amount of FLOPs and parameters, however, the TSM-MobileNet-V2 model has a very small amount of FLOPs and parameters, and maintains a high accuracy at the same time.

The TSM-MobileNet-V2-offline model is further compared with TSM-MobileNet-V2-online model and it is found that both of them have similar accuracy, computation and parameters. It should be noted that the TSM-MobileNet-V2-online used in the system has only 5.1 GFLOPs and 2.3 M parameters, which proves that the system is lightweight. The accuracy of TSM-MobileNet-V2-online model of each gesture class on the Jester dataset is also analyzed, and a confusion matrix is plotted. Figure 5 shows that the accuracy of TSM model can reach more than 90% for most gestures, and gestures used in the system are selected from these gestures. Details about the selected gestures are introduced in Section 4.1.

In addition, the model used in the system is compared with those used in other online systems. In their work, C3D and ResNeXt-101 are used for gesture classification [14]. Table 1 shows that our system has the virtue of fewer FLOPs and parameters with a high accuracy.

Experimental Analysis. It can be seen from the experimental results that 3D CNN based models such as I3D and C3D are not suitable for lightweight system due to the large amount of FLOPs and parameters. As for 2D CNN based models, a deep backbone also leads to large amount of FLOPs and parameters. Thus the MobileNet is used as the backbone in the proposed system. In order to capture the motion information in the video stream, the TSM module is inserted which will not increase the amount of FLOPs and parameters. Finally, a lightweight system with high accuracy is obtained. In addition, when comparing with others' online system, the system in this article has the advantage of being lightweight, and the online performance is discussed in Section 7.

4.2 Realtime Performance of the System

The experimental task in this work is built by unity and runs on a high-performance computer (i7-8700k, GTX 1080Ti). Details about the experimental task are introduced in Section 4.2. The

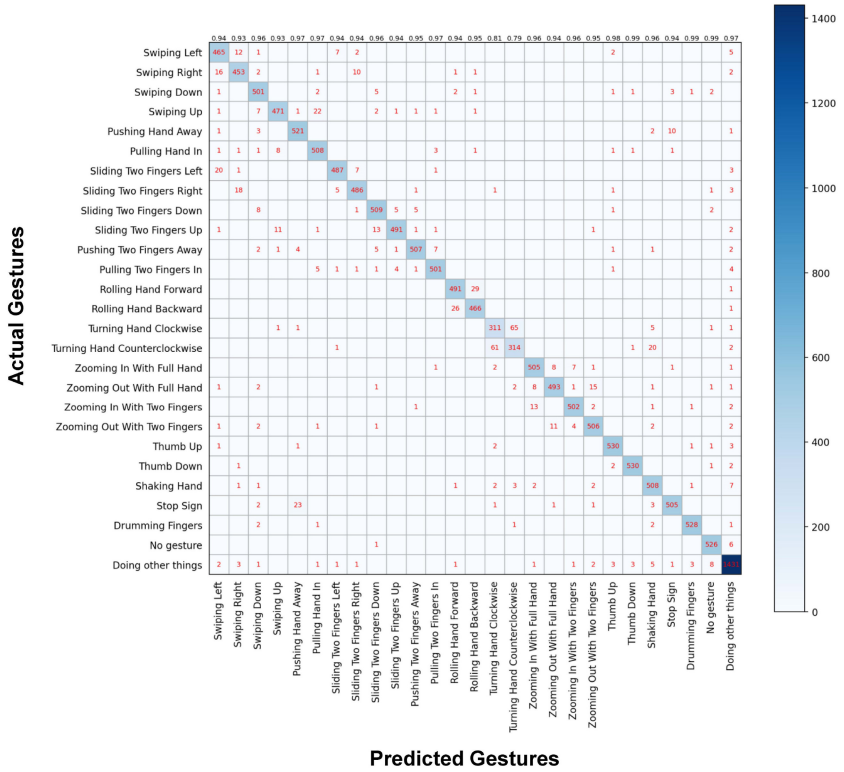

Fig. 5. Confusion matrix of TSM-MobileNet-V2-online model on Jester dataset.

Table 2. Results of Realtime Performance Test

	Total time of refreshing 360 frames	Average time of refreshing 1 frame	Frame rate
RGB camera	11.9940 s	33.32 ms	30 FPS
Our system	11.9962 s	33.32 ms	30 FPS

camera used in the gesture recognition system is an HD Pro webcam c920. Multiprocess technology is adopted in the experiment as shown in Figure 6(a). The gesture recognition system continuously recognizes the gesture, once a gesture is recognized, the information about the recognized gesture will be transmitted to the unity project through **transmission control protocol (TCP)** communication.

To verify the realtime performance of the system, the realtime frame rate of the camera and the gesture recognition system are calculated respectively. The test method is to record the time spent on refreshing 360 frames, then calculate the average time spent in a frame and the frame rate. Test results as shown in Table 2 indicate that the system has almost the same frame rate as the camera, which proves that the gesture recognition system has achieved realtime operation.

5 USER STUDY OF THE SYSTEM IN THE PRACTICAL APPLICATION

This section first introduces the participants and study design, then presents the procedure of the experiment. Afterwards, the metrics and hypotheses about the study are introduced. Finally, the statistical analysis used in this study is described.

Table 3. Four Groups of Experiments and Corresponding Gestures

Group	Gestures
A (Symmetrical gesture design with no activation delay)	"Sliding Two Fingers Up" "Sliding Two Fingers Down" "Sliding Two Fingers Left" "Sliding Two Fingers Right"
A_{delay} (Symmetrical gesture design with activation delay)	"Sliding Two Fingers Up" "Sliding Two Fingers Down" "Sliding Two Fingers Left" "Sliding Two Fingers Right"
B (Asymmetric gesture design with no activation delay)	"Sliding Two Fingers Up" "Swiping Down" "Swiping Left" "Sliding Two Fingers Right"
B_{delay} (Asymmetric gesture design with activation delay)	"Sliding Two Fingers Up" "Swiping Down" "Swiping Left" "Sliding Two Fingers Right"

5.1 Participants and Study Design

To investigate the influence of introducing activation delay and asymmetric gesture design, a user study was conducted with 20 participants (10 males, 10 females, aged 21–30). The participants are all right-handed. The entire experiment lasts about 45 minutes for each participant.

Two considerations must be taken into account before the experiment. On the one hand, too many and too complex gestures may cause too much cognitive as well as learning burden to users, which affects the reliability of the experimental results. On the other hand, as a result of the similarity of the hand poses, gestures of directional commands are more prone to misrecognition than other gestures when performing continuous gestures. Thus, eight gestures of direction commands as shown in Table 3 are selected in the experiment, and these eight gestures are divided into four groups. Group A and A_{delay} adopt symmetrical gesture design while group B and B_{delay} adopt asymmetric gesture design, group A and B have no activation delay while group A_{delay} and B_{delay} have activation delay. The activation delay used in the experiment is 16 video frames, which has been proven to be an appropriate activation delay in practical test. Participants are asked to perform these four groups of experiments in the random order. It should be noted that it is not completely random, which means that participants need to complete two groups of symmetrical gestures (A and A_{delay}) or two groups of asymmetrical gestures (B and B_{delay}) first, then complete the remaining two groups.

5.2 Procedure

Upon arrival, participants need to give their verbal consent to participate in the experiment and complete an interview to record their background information. Then, the experimenter verbally introduces the experimental task to the participants and ensures that they understand the task.

In the study, participants are asked to complete four groups of experiments in a random order as mentioned above, and they can rest for 5 minutes between each group of experiments. In each experiment, participants need to perform the dynamic gestures corresponding to the instructions according to the prompts on the computer screen. As shown in Figure 6(b), if the dynamic gesture

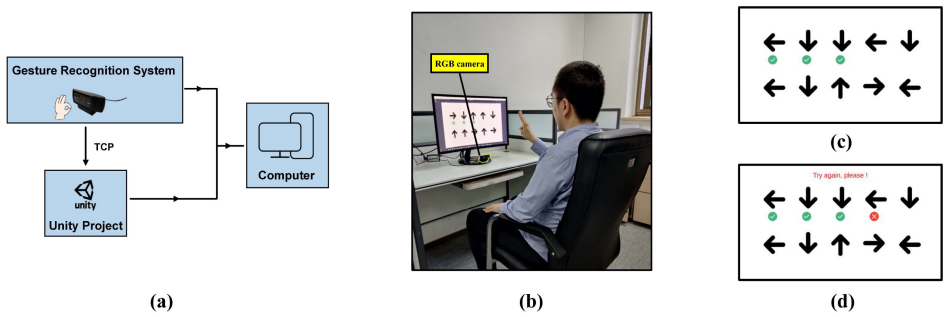

Fig. 6. (a) Architecture of the system. (b) An annotated photo showing a participant in the experiment. (c) The check mark on the screen. (d) The cross and prompt on the screen.

conducted by the participant is correct, a check mark will be displayed on the screen, and then the participant needs to complete the next dynamic gesture. If the participant conducts the wrong dynamic gesture, a cross and prompt will be displayed on the screen as shown in Figure 6(c), and the participant needs to conduct the dynamic gesture again until it is correct. Participants are required to perform ten groups of dynamic gestures according to the screen prompts in each experiment and there are ten dynamic gestures in each group. Therefore, participants must correctly complete a total of 100 dynamic gestures in one experiment, among which four groups of gestures are single instructions, such as upward instructions, and the other six groups of gestures are randomly generated from upward, downward, leftward and rightward instructions. Participants can rest between each group of dynamic gestures. After participants complete a group of experiments, they need to fill in the questionnaire. Participants have 5 minutes of training before each experiment.

5.3 Metrics

In order to evaluate the impact of activation delay and asymmetric gesture design on the recognition system, the recognition accuracy, misrecognition number, completion time and scores of the **System Usability Scale (SUS)** [3] are recorded.

- *Accuracy.* Accuracy is used to reflect the recognition accuracy of the system and it is calculated according to the following equation:

$$Accuracy = \frac{N_{correct}}{N_{total}} \times 100\%, \tag{1}$$

  here $N_{correct}$ refers to the number of successful recognition at the first time among the N_{total} dynamic gestures performed by each participant.
- *Misrecognition Number.* It should be noted that there may be several times of misrecognition when the participant performs one gesture, especially in the experiment of group A (no activation delay and no asymmetric gesture design); therefore, the total number of misrecognition is also recorded. Misrecognition number is used to reflect the recognition stability of the system, larger misrecognition number means that the system is unstable and is more likely to misrecognize gestures.
- *Completion Time.* Time efficiency is an important index for evaluating a system. In particular, the operation of introducing activation delay may have a great impact on time efficiency. In order to evaluate the time efficiency of the system, the average completion time of performing ten dynamic hand gestures is recorded.
- *SUS Scores.* The user experience of the system is very important, in order to obtain the subjective evaluation of the participants on the system and the usability of the system, the SUS

Table 4. Results of Different Groups

	Group				
	A	A_{delay}	B	B_{delay}	p-Value
Accuracy (%)	66.20 (11.16)^{abc}	85.20 (7.63)^{ad}	86.55 (6.40)^{be}	95.55 (2.65)^{cde}	<0.001(***)
Misrecognition number	70.75 (54.46)^{abc}	24.35 (16.66)^{ad}	18.10 (11.21)^{be}	6.80 (5.06)^{cde}	<0.001(***)
Completion time (s)	24.43 (8.47)^{ab}	29.04 (4.59)^{ac}	17.61 (5.28)^{bcd}	26.79 (6.36)^{d}	<0.001(***)
SUS scores	66.88 (16.00)	67.25 (18.17)	77.38 (13.14)	73 (17.29)	0.087

Values are presented as mean (standard deviation). Values in a row with the same superscript are significantly different in the follow-up test. *** $p < 0.001$, ** $p < 0.01$, * $p < 0.05$.

questionnaire is used. SUS is an inexpensive, yet effective tool for assessing the usability of a system, which provides an easy-to-understand score from 0 (negative) to 100 (positive) [1]. In our study, the SUS questionnaire is used as subjective data to evaluate the usability of the system.

5.4 Hypotheses

To better understand the user study, two hypotheses are established:

- **H1: "the introduction of an activation delay can significantly improve recognition accuracy, reduce misrecognition number, but may cause more completion time and lower SUS scores".**
  An activation delay gives the system more time to recognize the gesture; however, more time means worse time efficiency, thus this hypothesis is inferred.
- **H2: "the asymmetric gesture design can significantly improve recognition accuracy, reduce misrecognition number, but may cause more completion time and lower SUS scores".**
  This hypothesis is inferred because the asymmetric gesture design can almost completely avoid misrecognition due to the similarity of hand poses of direction command gestures in different directions, but the asymmetric gesture design may make users feel uncomfortable because they must spend more time.

5.5 Statistical Analysis

The recognition accuracy, misrecognition number, completion time and SUS scores were recorded to evaluate the system. Nonparametric test was used in the analysis since the data did not pass the variance homogeneity test. Considering that the study was a within-subjects design, the Friedman test was used to estimate the overall p-Values, and if significant, pair-wise comparisons were performed using the Wilcoxion signed-rank test.

6 RESULTS

The results of recognition accuracy, misrecognition number, completion time and SUS scores for different groups are summarized in Table 4. To better visualize and analyze the data, the interval plots showing the results of different groups are plotted in Figures 7 and 8.

For the accuracy, there are significant differences between different groups as shown in Figure 7(a). The accuracy of group B_{delay} is significantly higher than that of other groups, with an average accuracy of 95.55%. The average accuracy of groups A_{delay} and B are 24.35 and 18.10, respectively, and there is no significant difference between them, which indicates that accuracies of these two groups are at the same level. Group A has the lowest accuracy, with an average accuracy of 66.20%.

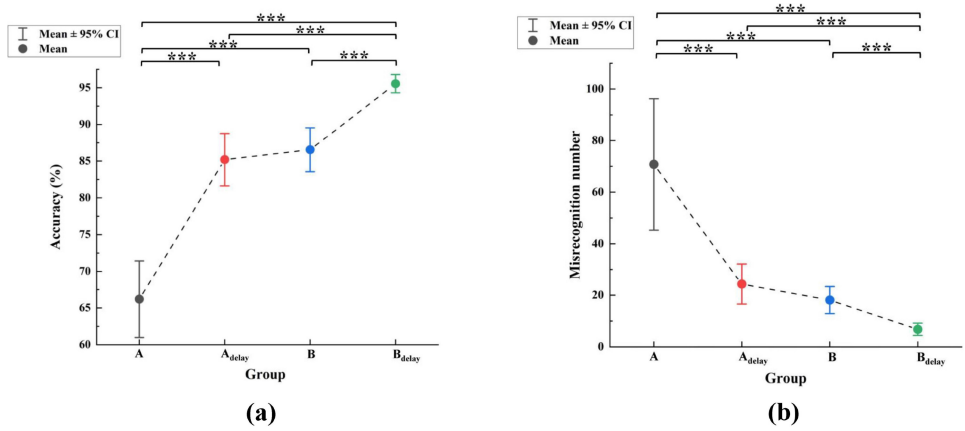

Fig. 7. Experimental results. (a) Accuracy of different groups. (b) Misrecognition number of different groups. *Error bars* are 95% confidence intervals (CI). *** $p < 0.001$, ** $p < 0.01$, * $p < 0.05$.

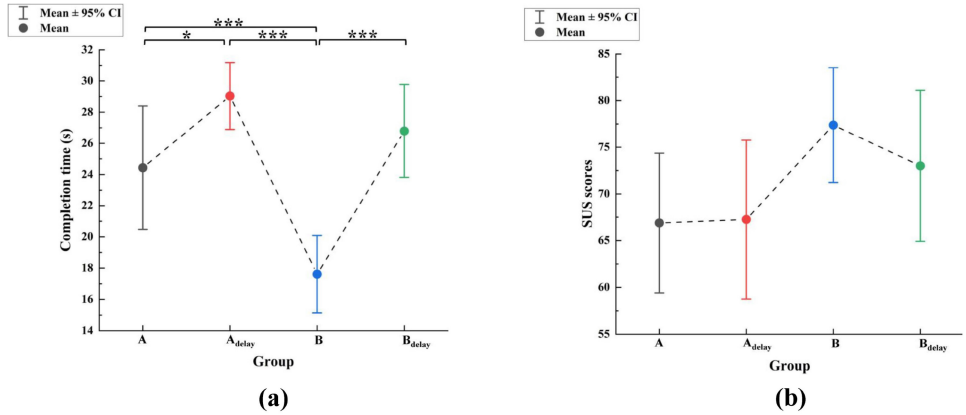

Fig. 8. Experimental results. (a) Completion time of different groups. (b) SUS scores of different groups. *Error bars* are 95% confidence intervals (CI). *** $p < 0.001$, ** $p < 0.01$, * $p < 0.05$.

For the misrecognition number, there are significant differences between different groups as shown in Figure 7(b). The misrecognition number of group B_{delay} is significantly smaller than that of other groups, with an average number of 6.8. The average misrecognition number of groups A_{delay} and B are 24.35 and 18.10, respectively, and there is no significant difference between them, which indicates that the misrecognition number of these two groups are at the same level. Group A has the largest misrecognition number, with an average misrecognition number of 70.75.

For the completion time, there are significant differences between different groups as shown in Figure 8(a). Group B has the least completion time, with an average completion time of 17.61 s. The average completion time of the symmetrical gesture group A_{delay} with delay is 29.04 s, which is significantly higher than that of the symmetrical gesture group A without delay, with an average completion time of 24.43 s. The average completion time of asymmetric gesture group B_{delay} with delay is 26.79 s, which is significantly higher than that of asymmetric gesture group B without delay, with an average completion time of 17.61 s.

For the SUS scores, no significant differences are found between different groups as shown in Figure 8(b). From the perspective of the average value, the SUS scores of the four groups are all above 65, and the average value of SUS scores of group B is the highest, which reaches about 77.38.

7 DISCUSSION

The evaluation of the system of different groups find differences in accuracy, misrecognition number and completion time. For accuracy, the groups with activation delay tend to have higher accuracy than that of groups without activation delay, the groups with asymmetric gesture design tend to have higher accuracy than that of groups with symmetric gesture design. This indicates that the two methods proposed in this article can significantly improve the accuracy of the system, which is consistent with **H1** and **H2**. However, there is no significant difference between A_{delay} and B, which implies that introducing an activation delay and using the asymmetric gesture design have the same effect on the improvement of accuracy. In addition, the accuracy of the group that introduces an activation delay and uses the asymmetric gesture design has the highest accuracy, which is about 95.55%. An interesting finding is that this accuracy is close to the accuracy of the TSM model used in this system. It should also be noted that the gesture data in Jester dataset are all trimmed videos, while the gesture data in the realtime recognition system is obtained by sliding time windows on the video stream, thus it is not an easy task for the system to have the same accuracy level as the model has. These two nearly equal accuracies indicate that the accuracy of the recognition system can achieve the same level as what the model has using the methods proposed in this work, which provides valuable guidance for designers who design practical systems.

For the misrecognition number, the groups with activation delay tend to have smaller misrecognition number than that of groups without activation delay, the groups with asymmetric gesture design tend to have smaller misrecognition number than that of groups with symmetric gesture design. This suggests that the two methods proposed in this article can significantly reduce the misrecognition number, which is consistent with **H1** and **H2**. The performance index reflected by the misrecognition number is somewhat similar to the concept of accuracy; however, there may be several times of misrecognition when the participant performs one gesture, thus misrecognition number can also be used to reflect the recognition stability of the system. What's more, a larger misrecognition number costs more time in the task; thus, it can also report the time efficiency of the system and this is discussed further in the following paragraph.

As can be seen from the previous discussion, the accuracy of the system has achieved a high level, but some other aspects need to be considered, such as time efficiency and usability. Completion time is recorded to evaluate the time efficiency of the system. Results of completion time show that the groups with activation delay takes significantly more time than that of groups without activation delay, which is consistent with **H1**. However, the groups with asymmetric gesture design takes significantly less time than that of groups with symmetric gesture design, which is contrary to **H2**. We believe that asymmetric gesture design may increase the cognitive and learning burden of users; for example, users may spend some reflection time thinking about what gesture to use before the task, thus we infer that groups with the asymmetric gesture design may spend more time. But the time efficiency becomes better in the groups with the asymmetric gesture design and we attribute this to the small misrecognition number in the groups with the asymmetric gesture design. Thus, we can infer that the asymmetric gesture design does not cause too much cognitive and learning burden to users.

When it comes to the SUS scores, statistical results show that there is no significant difference in SUS scores, which implies that introducing an activation delay and using the asymmetric gesture design do not cause too much discomfort to users. From the distribution of the SUS scores as shown in Figure 8(b), the group with the activation delay but without the asymmetric gesture design has

a relatively high SUS score. This suggests that the activation delay has caused a bad use experience for the system, while the asymmetric gesture design has not caused too much cognitive burden to the user, and the user is relatively easy to accept this way of interaction to a certain extent, which is contrary to **H2**.

The comparison with other online system proves that the proposed system can achieve the accuracy of the offline model while their system achieves 91.04% online accuracy with 93.75% offline model [14]. It should be noted that their online evaluation is conducted on videos of the dataset; however, our system is evaluated through the practical user study. The practical application is more complex and changeable, thus we argue that our evaluation is more challenging, which proves the effectiveness of the two proposed methods to avoid misrecognition.

On the whole, both the activation delay and asymmetric gesture design deal with how to determine the start of the gesture to distinguish different gestures. The hand gesture is a dynamic process; thus, how to define the start and end of a hand gesture has always been a difficult problem [14]. Especially for continuous recognition of gestures in practical applications, the problem of how to avoid misrecognition is actually how to determine the beginning of each gesture. The activation delay can give the system more time to recognize as well as give the user more time to react. It's actually the work to distinguish the beginning of different gestures by introducing a time lag in activation, which is to exchange time efficiency for less misrecognition. As to the asymmetric gesture design, it directly defines the beginning of different gestures by designing easily distinguishable gestures in the task. In fact, it works by exchanging the user's certain cognitive burden for less misrecognition. In the specific system design, we believe that we should consider the time efficiency, accuracy, and user cognitive burden, and formulate a reasonable system scheme.

8 CONCLUSIONS AND CONSIDERATIONS

In this article, a realtime dynamic hand gesture recognition system based on RGB camera is built. To avoid misrecognition, two methods are proposed by introducing activation delay and using asymmetric gesture design. Experimental results demonstrate that the proposed methods can effectively improve accuracy and reduce misrecognition number, but special attention should be paid when introducing the activation delay if the time efficiency is an important factor of the specific application. In future work, we will further investigate whether there is a threshold range of activation delay in the system and study the mappings of dynamic hand gestures in asymmetric gesture design.

For applications that using gesture recognition following similar principles as the study presented in this article, the following design considerations could be taken into account:

- In terms of interactive actions with temporal characteristics, it is a good choice to optimize the design and improve the accuracy from the time dimension.
- For classes of gestures that are prone to misrecognition, the practical system can be designed from the perspective of human-computer interaction.
- When designing the gesture recognition system, designer can introduce the activation delay and use asymmetric gesture design to avoid misrecognition in practical applications.

REFERENCES

[1] Aaron Bangor, Philip Kortum, and James Miller. 2009. Determining what individual SUS scores mean: Adding an adjective rating scale. *J. Usability Studies* 4, 3 (May 2009), 114–123.

[2] Evren Bozgeyikli and Lal Lila Bozgeyikli. 2021. Evaluating object manipulation interaction techniques in mixed reality: Tangible user interfaces and gesture. In *Proceedings of the 2021 IEEE Virtual Reality and 3D User Interfaces Conference*. 778–787. https://doi.org/10.1109/VR50410.2021.00105

[3] John Brooke. 1995. SUS: A quick and dirty usability scale. *Usability Eval. Ind.* 189 (11 1995).

[4] Marcio C. Cabral, Carlos H. Morimoto, and Marcelo K. Zuffo. 2005. On the usability of gesture interfaces in virtual reality environments. In *Proceedings of the 2005 Latin American Conference on Human-Computer Interaction (CLIHC'05)*. ACM, New York, 100–108. https://doi.org/10.1145/1111360.1111370

[5] J. Carreira and Andrew Zisserman. 2017. Quo vadis, action recognition? A new model and the kinetics dataset. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4724–4733. https://doi.org/10.1109/CVPR.2017.502

[6] Edwin Chan, Teddy Seyed, Wolfgang Stuerzlinger, Xing-Dong Yang, and Frank Maurer. 2016. User elicitation on single-hand microgestures. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems (CHI'16)*. ACM, New York, 3403–3414. https://doi.org/10.1145/2858036.2858589

[7] Christoph Feichtenhofer, Haoqi Fan, Jitendra Malik, and Kaiming He. 2019. SlowFast networks for video recognition. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision*. 6201–6210. https://doi.org/10.1109/ICCV.2019.00630

[8] Shao Huang, Weiqiang Wang, Shengfeng He, and Rynson W. H. Lau. 2017. Egocentric hand detection via dynamic region growing. *ACM Trans. Multimedia Comput. Commun. Appl.* 14, 1, Article 10 (Dec. 2017), 17 pages. https://doi.org/10.1145/3152129

[9] Matthew S. Hutchinson and Vijay N. Gadepally. 2021. Video action understanding. *IEEE Access* 9 (2021), 134611–134637. https://doi.org/10.1109/ACCESS.2021.3115476

[10] Evgeny Izutov. 2021. LIGAR: Lightweight general-purpose action recognition. *arXiv preprint arXiv:2108.13153* (2021).

[11] Shuiwang Ji, Wei Xu, Ming Yang, and Kai Yu. 2013. 3D convolutional neural networks for human action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35, 1 (2013), 221–231. https://doi.org/10.1109/TPAMI.2012.59

[12] Boyuan Jiang, MengMeng Wang, Weihao Gan, Wei Wu, and Junjie Yan. 2019. STM: Spatiotemporal and motion encoding for action recognition. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2000–2009.

[13] A. Karpathy, G. Toderici, S. Shetty, T. Leung, R. Sukthankar, and L. Fei-Fei. 2014. Large-scale video classification with convolutional neural networks. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, Los Alamitos, CA, 1725–1732. https://doi.org/10.1109/CVPR.2014.223

[14] Okan Köpüklü, Ahmet Gunduz, Neslihan Kose, and Gerhard Rigoll. 2019. Real-time hand gesture detection and classification using convolutional neural networks. In *Proceedings of the 2019 14th IEEE International Conference on Automatic Face Gesture Recognition*. 1–8. https://doi.org/10.1109/FG.2019.8756576

[15] Ji Lin, Chuang Gan, and Song Han. 2019. TSM: Temporal shift module for efficient video understanding. In *Proceedings of the IEEE International Conference on Computer Vision*.

[16] Zhihan Lv, Alaa Halawani, Shengzhong Feng, Haibo Li, and Shafiq Ur Réhman. 2014. Multimodal hand and foot gesture interaction for handheld devices. *ACM Trans. Multimedia Comput. Commun. Appl.* 11, 1s, Article 10 (Oct. 2014), 19 pages. https://doi.org/10.1145/2645860

[17] Menglong Zhu, Andrey Zhmoginov, Liang-Chieh Chen, Mark Sandler, and Andrew Howard. 2018. MobileNetV2: Inverted residuals and linear bottlenecks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 4510–4520.

[18] Joanna Materzynska, Guillaume Berger, Ingo Bax, and Roland Memisevic. 2019. The Jester Dataset: A large-scale video dataset of human gestures. In *Proceedings of the 2019 IEEE/CVF International Conference on Computer Vision Workshop*. 2874–2882. https://doi.org/10.1109/ICCVW.2019.00349

[19] P. Morrel-Samuels. 1990. Clarifying the distinction between lexical and gestural commands. *Int. J. Man-Mach. Stud.* 32, 5 (May 1990), 581–590. https://doi.org/10.1016/S0020-7373(05)80034-3

[20] Michael Nielsen, Moritz Störring, Thomas B. Moeslund, and Erik Granum. 2004. A procedure for developing intuitive and ergonomic gesture interfaces for HCI. In *Gesture-Based Communication in Human-Computer Interaction*, Antonio Camurri and Gualtiero Volpe (Eds.). Springer, Berlin, Berlin, 409–420.

[21] V. I. Pavlovic, R. Sharma, and T. S. Huang. 1997. Visual interpretation of hand gestures for human-computer interaction: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 19, 7 (1997), 677–695. https://doi.org/10.1109/34.598226

[22] Chen Qian, Xiao Sun, Yichen Wei, Xiaoou Tang, and Jian Sun. 2014. Realtime and robust hand tracking from depth. In *Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition*. 1106–1113. https://doi.org/10.1109/CVPR.2014.145

[23] Zhaofan Qiu, Ting Yao, and Tao Mei. 2017. Learning spatio-temporal representation with pseudo-3D residual networks. In *Proceedings of the 2017 IEEE International Conference on Computer Vision*. 5534–5542. https://doi.org/10.1109/ICCV.2017.590

[24] Niamul Quader, Juwei Lu, Peng Dai, and Wei Li. 2020. Towards efficient coarse-to-fine networks for action and gesture recognition. In *Computer Vision – ECCV 2020*, Andrea Vedaldi, Horst Bischof, Thomas Brox, and Jan-Michael Frahm (Eds.). Springer International Publishing, Cham, 35–51.

[25] Francis Quek, David McNeill, Robert Bryll, Susan Duncan, Xin-Feng Ma, Cemil Kirbas, Karl E. McCullough, and Rashid Ansari. 2002. Multimodal human discourse: Gesture and speech. *ACM Trans. Comput.-Hum. Interact.* 9, 3 (Sep. 2002), 171–193. https://doi.org/10.1145/568513.568514

[26] Adwait Sharma, Joan Sol Roo, and Jürgen Steimle. 2019. *Grasping Microgestures: Eliciting Single-Hand Microgestures for Handheld Objects.* ACM, New York, 1–13. https://doi.org/10.1145/3290605.3300632

[27] Karen Simonyan and Andrew Zisserman. 2014. Two-stream convolutional networks for action recognition in videos. In *Proceedings of the 27th International Conference on Neural Information Processing Systems (NIPS'14).* MIT Press, Cambridge, MA, 568–576.

[28] Du Tran, Lubomir Bourdev, Rob Fergus, Lorenzo Torresani, and Manohar Paluri. 2015. Learning spatiotemporal features with 3D convolutional networks. In *Proceedings of the 2015 IEEE International Conference on Computer Vision.* 4489–4497. https://doi.org/10.1109/ICCV.2015.510

[29] Radu-Daniel Vatavu and Laura-Bianca Bilius. 2021. *GestuRING: A Web-Based Tool for Designing Gesture Input with Rings, Ring-Like, and Ring-Ready Devices.* ACM, New York, 710–723. https://doi.org/10.1145/3472749.3474780

[30] Limin Wang, Yuanjun Xiong, Zhe Wang, Yu Qiao, Dahua Lin, Xiaoou Tang, and Luc Van Gool. 2019. Temporal segment networks for action recognition in videos. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41, 11 (2019), 2740–2755. https://doi.org/10.1109/TPAMI.2018.2868668

[31] Robert Y. Wang and Jovan Popović. 2009. Real-time hand-tracking with a color glove. In *Proceedings of ACM SIGGRAPH 2009 Papers (SIGGRAPH'09).* ACM, New York, Article 63, 8 pages. https://doi.org/10.1145/1576246.1531369

[32] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. 2009. User-defined gestures for surface computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI'09).* ACM, New York, 1083–1092. https://doi.org/10.1145/1518701.1518866

[33] Yiqi Xiao and Renke He. 2019. The intuitive grasp interface: Design and evaluation of micro-gestures on the steering wheel for driving scenario. *Universal Access in the Information Society* 19, 2 (2019), 433–450. https://doi.org/10.1080/10447318.2019.1571783

[34] Can Zhang, Yuexian Zou, Guang Chen, and Lei Gan. 2020. Pan: Towards fast action recognition via learning persistence of appearance. *arXiv preprint arXiv:2008.03462* (2020).

[35] Fan Zhang, Valentin Bazarevsky, Andrey Vakunov, Andrei Tkachenka, George Sung, Chuo-Ling Chang, and Matthias Grundmann. 2020. MediaPipe hands: On-device real-time hand tracking. In *Proceedings of the 2020 CVPR Workshop on Computer Vision for Augmented and Virtual Reality.*

[36] Yifan Zhang, Congqi Cao, Jian Cheng, and Hanqing Lu. 2018. EgoGesture: A new dataset and benchmark for egocentric hand gesture recognition. *IEEE Transactions on Multimedia* 20, 5 (2018), 1038–1050. https://doi.org/10.1109/TMM.2018.2808769

[37] Lu Zhao, Yue Liu, Dejiang Ye, Zhuoluo Ma, and Weitao Song. 2020. Implementation and evaluation of touch-based interaction using electrovibration haptic feedback in virtual environments. In *Proceedings of the 2020 IEEE Conference on Virtual Reality and 3D User Interfaces.* 239–247. https://doi.org/10.1109/VR46266.2020.00043

[38] Bolei Zhou, Alex Andonian, Aude Oliva, and Antonio Torralba. 2018. Temporal relational reasoning in videos. In *Proceedings of the European Conference on Computer Vision (ECCV'18).* 803–818.

Received 15 December 2021; revised 29 June 2022; accepted 1 September 2022