Replace Your Mouse with Your Hand! HandMouse: A Gesture-Based Virtual Mouse System

Qiujiao Wang[1], Zhijie Xie[2*]
Department of Foundation, Southwest Jiaotong University Hope College, Chengdu, China[1]
Chengdu Transportation + Tourism Big Data Application Technology Research Base, Chengdu, China[1]
School of Computing and Artificial Intelligence, Southwest Jiaotong University, Chengdu, China[2]
Chengdu Office, Shanghai Tecwin Software Technology Co., Ltd, Chengdu, China[2]

Abstract*—The existing gesture-based operating systems can only simply operate a single piece of software or a specific system, and are not compatible with other applications of mainstream operating systems. In this paper, based on the MediaPipe gesture recognition framework, we design HandMouse, a virtual mouse system that operates using hand gestures. It has the following characteristics: 1. The user does not have contact with the computer hardware when using the system; 2. It requires only one hand to operate, and the design of the gesture considers the ergonomics of the hand; 3. It has most of the functions commonly used in a physical mouse; 4. It can locate the operating area with relative precision. We invited 27 participants to use and evaluate the virtual mouse and then conducted an experiment to compare the performance of the virtual mouse with the physical mouse. The results show that this virtual mouse has a good learning effect and is a great alternative to the physical mouse in public places. The demonstrated operation video is available on *https://github.com/wanzhuxie/HandMouse-IJACSA.

***Keywords*—Virtual mouse; ergonomics; gesture; MediaPipe**

I. Introduction

Gesture recognition has been the subject of research for several decades. As early as 1997, paper [1] has provided a comprehensive summary of the accumulated technology of gesture recognition at that time, and it has experienced rapid development over the past 20 years. There are approximately four research directions for gesture recognition. The first direction involves the use of physical sensors to transmit information. For example, papers [2, 3] described sensor gloves with gesture recognition capabilities, while the system proposed in paper [4] processes finger pressure signals, and a virtual keyboard was designed in paper [5]. The second direction involves the use of cameras to extract finger information. Although physical sensors are not in direct contact with the computer, the recognition process requires the assistance of gloves of different colours to help the computer identify gestures, as described in papers [6, 7]. The third direction focuses on processing images without external elements to assist the computer. Advanced algorithms are used to extract hand images from gestures and infer their meaning, as demonstrated in papers [8-10]. The fourth direction involves inferring the coordinates of the key points of the hand and leaving the recognition of gestures to the specific gesture designers. This allows designers to consider more details and design a wider range of gestures. Deep learning techniques are commonly used for this purpose, for example, in 2016, paper [11] presented a gesture recognition system based on recursive 3D convolutional neural networks, achieving an accuracy of 83.8%. In 2017, a paper [12] trained a hand recognition model using 18,000 stereo hand images and the 3D key points in each image in different scenes, demonstrating a good tracking performance. In 2019, Google Artificial Intelligence Laboratory released an open-source gesture recognition framework called MediaPipe Hands [13], which continues to be maintained. MediaPipe Hands was trained using 300 million real-life hand gesture images with the corresponding 3D key point coordinates of the hand, achieving an overall recognition accuracy of 95.7%.

Thanks to the increasing efficiency and accuracy of gesture recognition technology, it has a wide range of applications in areas such as sign language recognition, remote-controlled robots, and human-computer interaction (HCI) [14-16]. Compared with the traditional mouse click or touch screen operation, it is more popular to add gesture operation into human-computer interaction system [17]. Gesture operation has significant appeal to customers due to its novelty, and the potential purchasing power of customers may be stimulated by the exploration or trial of gesture control. Gesture operation enables humans to have no physical contact with the machine, allowing users to be at a greater distance from the device, which enables a better layout planning for venue managers, prevents accidental damage caused by customers touching the device, and prevents the spread of diseases caused by touching. In paper [18], preliminary research on gesture output of Chinese was conducted. In paper [19], a gesture-based image viewer software was designed and applied to touchless operations in surgical room scenarios. The study in [20] proposed a gesture recognition method for controlling in-car devices. Furthermore, gesture recognition has been applied in gaming interactions as discussed in papers [21, 22], and an interesting study conducted in paper [23] explored the use of gestures for simple coding purposes.

Since the gesture commands have been used to realize the simple control of specific software, can they be further developed by using gestures to realise the functions of the physical mouse, such as the commonly used left-clicking, left-double-clicking, right-clicking, etc.? The answer is yes. At present, the gesture-based system can realize all or some of the

*Corresponding author
This work was supported by the Chengdu Philosophy and Social Science Key Research Base: Chengdu Transportation + Tourism Big Data Application Technology Research Base, China (20231003).

functions of the physical mouse and allow the operator to have no direct contact with the machine hardware, for what we can call it a virtual mouse. The virtual mouse has been the subject of extensive research by many researchers over the last decade. Although the result presented in paper [24] focused on the virtual keyboard input, its techniques are still relevant to the virtual mouse and serve as a valuable reference. Paper [25] employed a two-layered Bayesian network technique for gesture recognition and designed a virtual mouse system. Paper [26] designed a gesture-based virtual controller for manipulating 3D objects, taking into account the 3D data of finger movements. In paper [27], used convolutional neural network technology to recognise finger and fist gestures, and then developed a simple virtual mouse control system.

However, current gesture operations are specific to particular systems or software, where each gesture represents a specific command for a particular software. Unlike a physical mouse, these gestures are tailored to specific software and lack universality.

Based on existing gesture recognition technology, combined with the physiological structure of the human hand, this paper designs a set of simple and easy-to-use mouse operation gestures. On this theoretical basis, this paper presents an absolutely contactless, entirely gesture recognition-based virtual mouse system. The virtual mouse is independent of the software that is being operated and has a high degree of versatility. With relatively comprehensive functions, the virtual mouse is like a physical mouse, and most of the functions that can be achieved with a physical mouse can also be achieved with the virtual mouse. The general flow of the system is shown in Fig. 1, where the video frame image is captured by the camera, and the key points of the hand are detected by MediaPipe. The gesture recognition module analyses the coordinates of the key points and identifies the current gesture. When a pre-defined gesture is made, the mouse operation module immediately releases the mouse signals to operate any software.

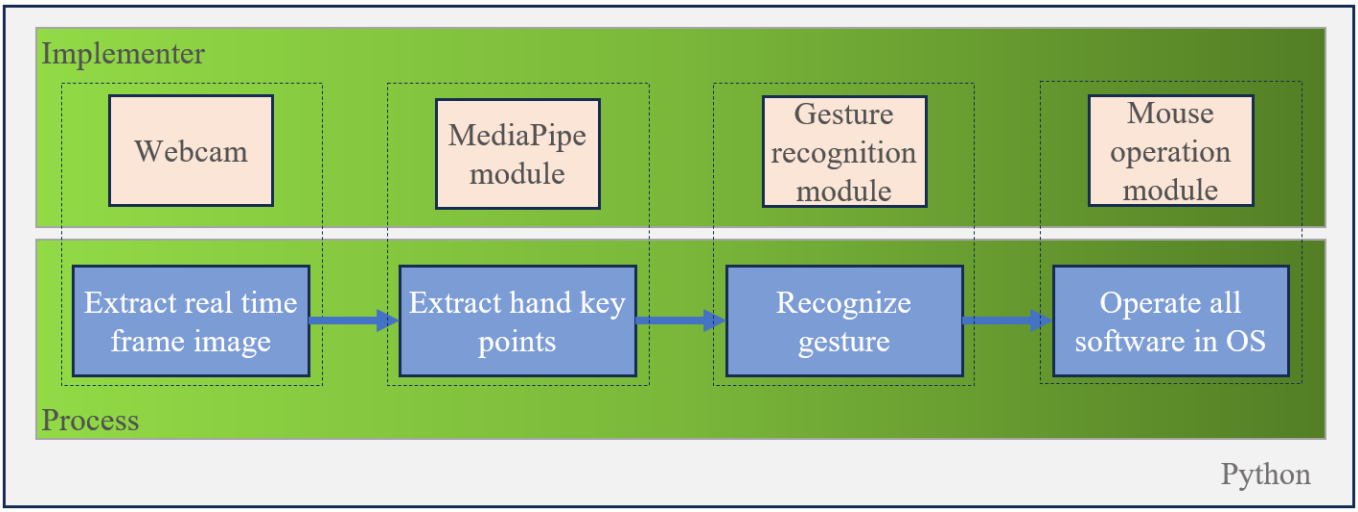

Fig. 1. The system architecture of the virtual mouse system.

II. FOUNDATIONS

A. Key Points of a Hand

There are many ways to recognize various gestures, such as feature matching, extraction of hand key points, and so on. Feature matching can only recognize the pre-defined hand postures, while key points extraction supports all kinds of hand posture, which has stronger speculation ability. As shown in Fig. 2, the 21 key points can basically describe the gesture information of a hand, and the recognition of gesture can be transformed into the analysis of the key points and further into the conversion of abstract image information into specific mathematical information. In this paper, P_n is used to represent the key points at a specific position, where $n \in [0, 20]$, and the sequence number of the starting key point is 0.

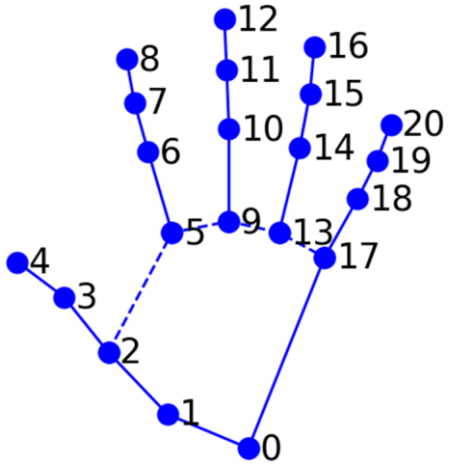

Fig. 2. Key points of a hand.

B. MediaPipe

MediaPipe is a cross-platform machine learning framework [28] that includes a number of sub-frameworks such as gesture recognition, face recognition, whole-body posture recognition, 3D object coordinate inference and so on. It has been widely used in many areas of life and industry. Papers [29-31] apply MediaPipe to home sports equipment, gesture language expression system, and human posture simulator. With the assistance of MediaPipe, papers [32, 33] developed the measurement system for some human rehabilitation movements and the abnormal gesture detection system for patients with the nerve injury. Paper in [34] developed a human emotion detection system using MediaPipe.

MediaPipe Hands is one of the frameworks of MediaPipe, which supports five fingers and gesture tracking, and can infer 21 stereo nodes of one hand from a frame image. Even if the palm is partially displayed or the hands are self-occluded, it can achieve high robustness, high performance, and low time consumption, so it has been applied in a variety of fields. Paper [35] designed a sign language expression system for Japanese; paper [36] captured the motion trajectory of fingers by analyzing the 3D coordinates of the key points of hand, and then designed an air-writing system; paper [37] used MediaPipe to identify the driver's hand information in a driver distraction warning system.

The system described in this paper also uses MediaPipe Hands to extract the coordinate information from the 21 key points of the hand. It reads each frame image captured by the camera and provides three coordinate values of the key points. Since the image can be scaled to meet the needs of image analysis, information transmission, image display and so on, when it is processed, the original coordinate values of the key points are normalised values, i.e. each coordinate value is a proportional value relative to the image size. The normalized datum of the X and Z coordinate value is the width of the image, and the normalized datum of Y coordinate value is the height of the image. Therefore, before using the coordinate value of key points, it is necessary to calculate the pixel coordinates of key points according to the frame image size obtained by the camera.

Although the performance of the MediaPipe Hands was excellent, there is a drop in recognition accuracy when the background colour is similar to the hand colour, or when the overall lighting is poor. The user may also make errors. The system therefore optimises the recognition reliability of MediaPipe. When recognising the current gesture, it is only considered to have changed if the camera captures the same gesture three times in a row. Otherwise, it is considered a single misrecognition by MediaPipe or the user, and the recognition result of the previous frame is returned.

C. Coordinate System

The default coordinate system of the display screen of the system takes the upper left corner as the origin, the horizontal right direction along the screen is the positive direction of the X axis, and the vertical downward direction along the screen is the positive direction of the Y axis. The coordinates (X_{max}, Y_{max}) of the bottom right corner of the screen are related to the screen resolution. When analyzing gestures, the larger the Y coordinate of the hand key point, the lower the position of the point.

III. Basic Design of Gesture Operations

A. Design Principles

First of all, we give three design principles:

1) Single-handed operating. Although two-handed operation can express more information, as in the application scenarios in [38-40], it also increases the demands on the operator. Compared with single-handed operation, it has two disadvantages. Firstly, the operator has to raise both hands at the same time to make gestures during the operation and may feel tired after a short time. Secondly, when two hands are operating together, they may be required to make different gestures, so the error rate of two-handed operation may be higher than that of single-handed operation or the operation of two hands making the same gesture.

2) Simple gestures considering. Due to the physiological structure of the hand, most people can't stretch out their ring finger alone as easily as their index finger. They need to work with other adjacent fingers to make gestures quickly and accurately. Therefore, the system does not use the ring finger alone as an indication signal, which reduces the probability of error gestures.

3) Multiple postures of hand supporting. In the process of specific operation, many scholars have studied the palms parallel to the display screen as a condition for gesture recognition [8, 25, 39, 41]. Meeting this condition can indeed improve the accuracy of detection, but it is not a small challenge to the user's physical strength and patience. In fact, if the palm plane is kept parallel to the screen all the time, the user's arm and wrist will feel tired in a short time and will not be able to perform gesture operations, which will reduce the user's use experience and may also cause the user's resistance. What can be determined is that people would prefer the system to be compatible with multiple postures of the same gesture, so that different postures can be changed during a long period of operation to alleviate the fatigue caused by gesture operation. Therefore, the system supports any angle between the palm plane and the screen plane, and users can make gesture signals according to their customary posture. All postures shown in Fig. 3 indicate that only the index finger is extended. It should be noted that, for ease of expression and understanding, all the gesture images in this paper are generated from the perspective of the operator. The gesture image captured by the camera should be the image observed in the opposite direction. However, this does not affect the design of the algorithm. As mentioned above, the system supports multi-position operation.

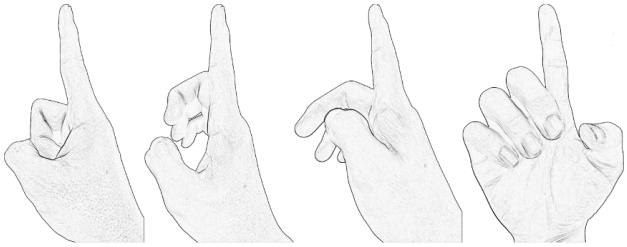

Fig. 3. Multiple gestures express that only the index finger is extended.

B. Gesture Unit Judgment

In this paper, the gesture of a single finger is called a gesture unit. The gesture unit has two states, the extension and closed of the finger. Each gesture signal is a combination of the states of each finger. To improve versatility and fault tolerance, this system does not consider the angle between fingers, but only whether each finger extends. We use five binary numbers to represent gestures for the convenience of display. Each number represents the extension and closed of the corresponding finger, with 1 representing the extension and 0 representing the closed. The numbers from left to right represent the status of the thumb, index finger, middle finger, ring finger, and little thumb. For example, [01000] means that only the index finger is extended.

For the other four fingers except the thumb, the extended condition is that the fingertip's key point is above its corresponding three other key points as shown in Fig. 2, as

$$Y\left(P_i\right) < Y\left(P_j\right) \tag{1}$$

where, $i \in \{8,12,16,20\}$ is the key point indexes of the fingertip and $j \in [i-3, i)$ is one of three other key points indexes.

This determination method is suitable for the scenario where multiple fine-tuned gestures are used to express the same signal, reducing the fatigue caused by the user making the same gesture for a long time.

For the thumb, the relationship between the thumb and the other four fingers is determined first, and then its extension and closed are determined based on the coordinate relationship between the thumb tip key point and the other key points of the thumb. If $X\left(P_2\right) > X\left(P_{17}\right)$, we define the handedness to be the right, and if $X\left(P_3\right) > X\left(P_4\right)$, then the thumb is closed, otherwise it is extended. Similarly, if $X\left(P_2\right) \le X\left(P_{17}\right)$, we define the handedness to be the left, and if $X\left(P_3\right) < X\left(P_4\right)$, then the thumb is closed, otherwise it is extended.

C. Gesture Operation Area

Mouse movement is the most common operation, and in terms of the difficulty of making gestures, it is easier to extend the index finger alone. In addition, using the index finger to guide the mouse pointer is also in line with the public's understanding of the habit. Therefore, this system uses the extension of the index finger as the signal to set the mouse position, and the mouse pointer moves with the movement of the tip of the index finger.

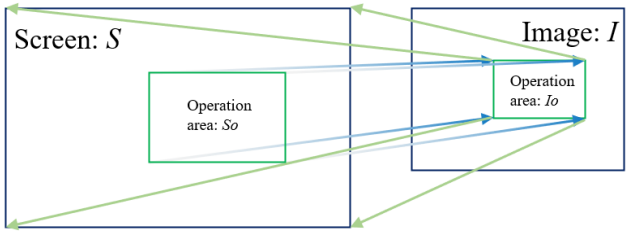

Fig. 4. The correspondence between the screen area and image area.

As shown in Fig. 4, *S* indicates the display screen area, *So* indicates the active area of the index fingertip on the screen *S*, which should be significantly smaller than *S*, otherwise it will cause two problems. Firstly, the components at the bottom of the screen cannot be operated because when the index fingertip is at the bottom of the screen, the hand is outside the screen area, and the recognition rate of a hand is very low, or even impossible. Similarly, when using the right hand, the components on the right side of the screen cannot be manipulated, and when using the left hand, the components on the left side of the screen also cannot be manipulated. Secondly, if the active area of the index fingertip is larger, the active area of the human hand will also be larger, and the range of activity of the arm will increase, which not only tires the arm, but also takes time to position the pointer. In fact, when the physical mouse is positioned, it will move much less than its pointer. Therefore, the size of *So* should be consistent with the motion range of the index fingertip when only the wrist is active. It is therefore necessary to map the fingertip coordinates in *So* to *S* in order to set the pointer coordinates.

In fact, there is another intermediate area between *So* and *S*, which is the corresponding area *Io* in the image of *So* on the screen, as shown in Fig. 4, where *I* is the area of the image captured by the camera. It should be noted that the height-width ratio of the image may not be the same as that of the screen. The coordinates of each key point of the hand are captured from the image, so the operation area in the image should be considered initially. But when the system is working, it is not necessary to display the image area, shown at the top right of Fig. 5, on the screen. The operator is not obliged to calculate the mapping relationship between the image and the screen, but only needs to customize the operation area on the screen. Depending on the size of the image, the system first maps the user-defined operation area So onto the operation area *I*o and then maps *I*o onto the screen, so that the index fingertip can wander around the entire screen and operate on target objects at any position. The above mapping process is also shown in Fig. 4.

It needs to consider multiple factors when setting the area of *S*o. If the area is too large, the swing range of the user's arm needs to increase, which will increase the user's fatigue. If the area is too small, because the area is mapped to the entire screen area, the slight movement of the fingers in this area may produce a large mouse pointer movement effect on the entire screen, which will affect the positioning accuracy. By default, the system sets the size of *S*o to a quarter of the screen size and places it in the centre of the screen. In fact, the size and

position of the operation area depend on the relative position of people and cameras. In order to adapt to cameras with different parameters or different installation positions of cameras with the same parameters, the system supports the operators to adjust the operation area.

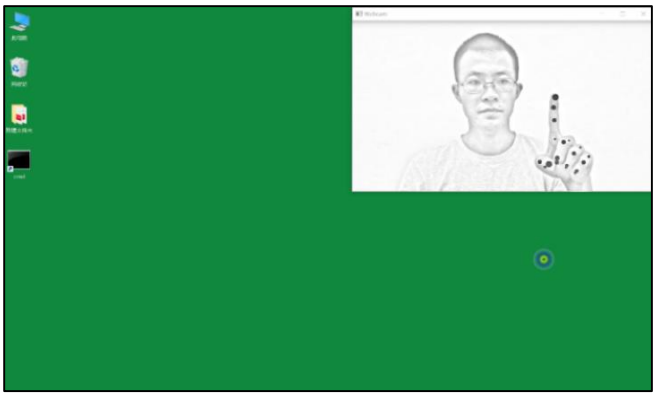

Fig. 5. The real screen area and the image area.

D. Mouse Position Calculation

Before performing mouse operations, the operator needs to use the tip of their index finger to locate the target position on the screen. We use $f(x,y)_s$ representing the coordinates of the index fingertip on the screen, and then we will derive and calculate it.

If R_{io} indicates the size of the operation area in the image, R_i indicates the size of the image, R_{so} indicates the size of the operation area in the screen, and R_s indicates the size of the screen. They have the following corresponding relationship.

$$R_{io} / R_i = R_{so} / R_s \tag{2}$$

The width and height of each of the four areas satisfy the above relationship, i.e., the four dimensions above can represent both width and height.

From the mapping relationship between S_o and I_o we can see that any position in S_o (the image operation area) $f(x,y)_{io}$ and the corresponding position $f(x,y)_{so}$ in I_o (the screen operation area) have the relationship.

$$f(x,y)_{io} = \frac{R_{io}}{R_{so}} f(x,y)_{so} \tag{3}$$

From the mapping relationship between I_o and S, the position of the fingertip in S (the screen) $f(x,y)_s$ has the following relationship with the corresponding position $f(x,y)_i$ in I (the image).

$$f(x,y)_s = \frac{R_s}{R_{io}} \left(f(x,y)_i - f(x,y)_{iom} \right) \tag{4}$$

Where, $f(x,y)_{iom}$ is the upper left corner of I_o (the image operation area), which can be obtained by the following equation.

$$f(x,y)_{iom} = \frac{R_{io}}{R_{so}} f(x,y)_{som} = \frac{R_i}{R_s} f(x,y)_{som} \tag{5}$$

Where, $f(x,y)_{som}$ is the upper left corner of S_o (the screen operation area).

From above equations, we can express the relationship between $f(x,y)_s$ and $f(x,y)_i$ as

$$f(x,y)_s = \frac{R_s^{\,2}}{R_i R_{so}} \left(f(x,y)_i - \frac{R_i}{R_s} f(x,y)_{som} \right) \tag{6}$$

Where, the screen size R_s is fixed; the image size R_i is depends on the camera settings and is also known. The size of the screen operation area R_{so} and its upper left corner $f(x,y)_{iom}$ are user defined or system default given values, which are also known. So $f(x,y)_s$ varies with the variable $f(x,y)_i$, and the latter can be calculated by the logic of the image processing section described above, up to this point we obtain an expression for the coordinates of the index fingertip.

In order to ensure that the value of the index fingertip's coordinate $f(x,y)_s$ does not have a negative value or a value that exceeds the screen area, we place a restriction on the final mouse pointer coordinate value, and the X-coordinate of the mouse pointer satisfies the following equation.

$$P_x = \begin{cases} 0 & , f(x,y)_s|_x \in (-\infty, 0) \\ R_s|_x & , f(x,y)_s|_x \in (R_s|_x, +\infty) \\ f(x,y)_s & , f(x,y)_s|_x \in [0, R_s|_x] \end{cases} \tag{7}$$

Similarly, the Y-coordinate of the mouse pointer satisfies

$$P_y = \begin{cases} 0 & , f(x,y)_s|_y \in (-\infty, 0) \\ R_s|_y & , f(x,y)_s|_y \in (R_s|_y, +\infty) \\ f(x,y)_s & , f(x,y)_s|_y \in [0, R_s|_y] \end{cases} \tag{8}$$

where, $R_s|_x$ is the width in pixels of the screen, $R_s|_y$ is the height in pixels of the screen, $f(x,y)_s|_x$ is the theoretical X-coordinate of the mouse pointer in the screen and $f(x,y)_s|_y$ is the theoretical Y-coordinate of the mouse pointer in the screen.

E. Mouse Sensitivity Design

Due to the hand inevitable tremor in front of the camera, the mouse pointer on the screen often frequently jump caused by the small movement of the hand. Hand tremor is unavoidable, and we can only minimise the effect caused by it [42]. The position can be recorded and compared with the last recorded position, and if the difference is within a certain error range, it is considered to be an invalid signal caused by hand tremor. In fact, this method filters out the effect of hand tremor, but it also ignores the signals that occur when the user actually

intends to move the mouse to another near position. Additionally, when the mouse is moved using gesture signals, the movement of the mouse pointer is not smooth, but rather jumps in discrete steps, with the step size depending on the error value. This may result in a less smooth user experience.

For mouse pointer positioning, this system detects the coordinates of the index fingertip on the screen and calculates a weighted value by combining the fingertip coordinates with the current mouse pointer position as the new pointer position.

The specific algorithm is as follows:

Step 1: Record the last mouse pointer coordinates $P(x_l, y_l)$.

Step 2: Detect the coordinates of the index fingertip in real time and calculate the weighted result of the fingertip coordinates and the last mouse pointer coordinates.

$$P(x_r, y_r) = P(x_l, y_l) + P(a(x_c - x_l), b(y_c - y_l)) \quad (9)$$

where, $P(x_r, y_r)$ is the weighted coordinates, x_c, y_c are the X and Y coordinates of the current index fingertip point respectively, a and b are constants, their range is all (0,1], when $a = b = 1$, it is equivalent to using the current fingertip coordinates as the coordinates of the mouse pointer.

The system correlates the value with the screen resolution, i.e.

$$\frac{a}{b} = \frac{R_s|_x}{R_s|_y} \quad (10)$$

Step 3: The weighted result of the fingertip coordinates and the last mouse pointer coordinates may have values that are outside the screen area, so the weighted result is constrained similarly to the "(8)" and "(9)" to ensure that the mouse pointer coordinate values are not outside the screen area.

IV. Gesture Design

A. Signal Categories and States

The Fig. 6 is the logic flow chart of the system operation, the blue rectangle indicates the system state, we define three states for the system, respectively, the *Ready* state, which is the first state after the system initialisation, the *Pressing* state during the mouse pressing process, and the *Adjusting* state during the operation area adjustment process.

The solid rounded rectangle represents the operation signals. Considering the mouse operations commonly used in the operating system, the operation signals of this system are left click, left double click, right click, scroll wheel up, scroll wheel down, left button press, and left button release, which are seven signals in total. Except for the two operations of scroll wheel, the other operations need to accurately locate the target area before the operation actions, so we call them locating operations, and the two operations of scroll wheel are called non-locating operations. In addition to mouse operations, the system supports customisation of the operation area, including setting its size and position. After initialization, if the system detects that only the index finger is extended [01000], it enters the *Ready* state, in which the mouse pointer follows the tip of the index finger.

The dashed rectangular rectangles indicate the indication signals, which are used to support the state switching and operation signaling.

The design of the gesture signals is the core of this system. For the Locating operations, the first step is to move the mouse pointer to the target object. Once the mouse pointer is placed, it should be held still while the corresponding gesture signal is performed. When making a gesture signal, the index finger should be extended as it plays a role in the composition of the gesture signal. If the index finger needs to be closed during this time, the spatial position of the index finger would change from extended to closed, making it difficult for the system to recognise the actual purpose of the index finger movement. It could be interpreted as a signal for an operation or simply as a movement of the mouse pointer. Therefore, the index finger should remain extended for positioning operations.

Indeed, if someone wants to keep the index finger away from the next action signal, an intermediate state can be introduced before the Locating operation. In this state, the mouse pointer no longer moves with the movement of the index fingertip, but remains stationary on the target object until the next signal action. During this time, the index finger can move freely to perform gestures. However, using this approach would require at least two actions to simulate one operation of the physical mouse, including a state transition action and a mouse operation action, which may introduce inconvenience in the operation process. Therefore, this system adopts the method that the index finger participates in the gesture signal composition for the locating operations.

Furthermore, it should avoid making gestures that may cause significant changes in the position of the index fingertip for the next operation. For example, extending the ring finger may cause an obvious change in the position of the index fingertip. As mentioned earlier in the discussion of mouse sensitivity, any change in the position of the index fingertip will have a larger impact on the screen. Due to this change, the mouse pointer may have moved beyond the intended target area. In fact, the extension and closing of the thumb have minimal impact on the other four fingers. Therefore, the system does not utilize an intermediate state but directly uses the thumb to send the operation signal.

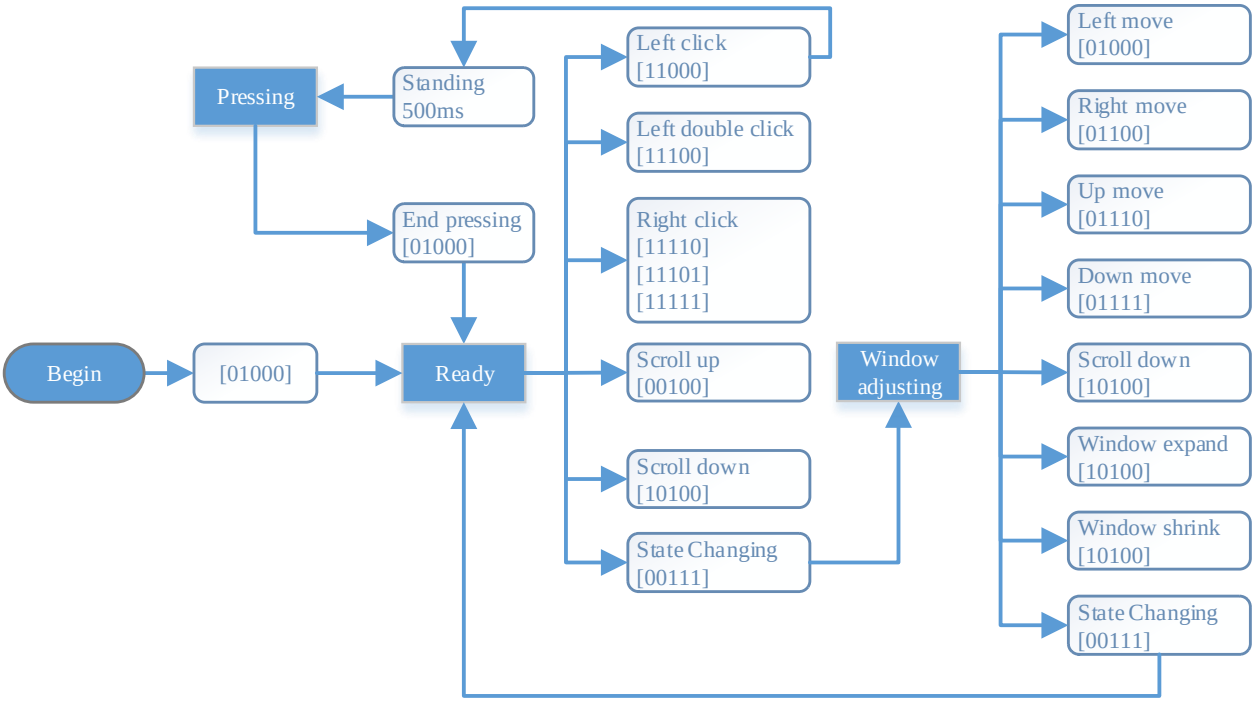

Fig. 6. System flow chart.

B. Operation Signal Design

1) Left clicking, pressing and releasing: Precision control and user comfort are two essential considerations in gesture operations [42, 43]. When designing the signals, the frequency of use of each operation is an important consideration. The more frequently an operation is used, the simpler the corresponding gesture should be. Left-clicking is the most frequently used mouse operation, and we use the index finger as the corresponding gesture for it, similar to the approach used in the references [44, 45]. When users want to left-clicking on an object, they first extend their index finger [01000] and place the mouse pointer over the target area. Keep the index finger stationary, the thumb can be extended. Then the system will detect the [11000] gesture, which represents the execution of a left-clicking. After the click, the thumb should immediately close, maintaining the [01000] gesture to guide the mouse pointer for the next operation. If the thumb remains extended in the target position for more than 0.5 seconds, the system performs a pressing operation on the target position.

The Pressing state is a specific state designed for dragging files. In the Ready state, if the gesture [11000] is held for 0.5 seconds, the system switches from the Ready state to the Pressing state. Similar to the logic of a physical mouse, the pressing action is executed by the system regardless of whether there is a target object under the mouse pointer. In the Pressing state, the mouse pointer drags the file along with the movement of the index finger until the thumb is retracted, at which point the system releases the left button and stops dragging, and the system immediately return to the Ready state.

To provide clearer feedback on the current state of the system, a semi-transparent coloured halo can be displayed around the mouse pointer in the Ready state. Fig. 7 shows the effect of the halo on the target position with four different background colours. This halo not only serves as an indicator but also helps visualize the position of the mouse pointer. When the system transitions to the Pressing state, the halo is hidden to prompt the user for the current drag operation. This is, of course, optional, and unless the scene has a complex background, it is recommended not to use a coloured halo as it may affect the activity of the pointer.

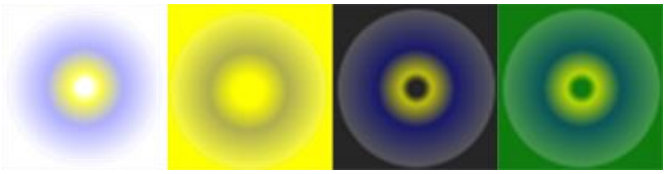

Fig. 7. The halo effect with different background colours.

2) Left-double-clicking: The left-double-clicking operation is also commonly used. From a conventional standpoint, a double-clicking consists of two single-clicks operations, as is the case for the physical mouse. However, when sending a signal, gesture-operated systems generally have a much larger range of motion than the physical mouse. As a result, the time required to perform a double-clicking operation is significantly longer than that of a physical mouse operation. In addition, performing a left-clicking gesture in this system

requires the thumb to be extended and closed once respectively, which can lead to hand fatigue after multiple operations. Therefore, our system uses a combination of the index and middle fingers as the gesture signal for the left double-click. In this case, the user only needs to extend and close the thumb once. When the user wants to make a left double-click on a target area, they can extend both the index finger and middle finger, and the mouse pointer will still follow the movement of the index fingertip. Once the pointer is over the target area, extending the thumb completes the double-click. This approach is similar to the method described in study [45].

3) Right-clicking: Similarly, the gesture corresponding to the right click is the combination of the index, middle and ring fingers. However, due to the physiological structure of the human hand, the ring finger has less dexterity than the other fingers, so the system accepts that the little finger can also act as the ring finger, and the ring finger can of course act on its own. Therefore, there are three gestures corresponding to the right-clicking, respectively [01110], [01101] and [01111].

4) Wheel scrolling and pausing: In most scenarios, scrolling operations with the mouse wheel do not require precise positioning of the mouse pointer. This system uses the gesture [00100] as the signal for scrolling the mouse wheel forward, and uses the gesture [10100] for scrolling the mouse wheel backward. In the *Ready* state, any gesture other than those mentioned above can temporarily pause the current scrolling operation.

C. Operation Area Adjusting

As mentioned above, the size and position of the work area can be adjusted. In order to still be able to operate with single hand, some of the gestures used to adjust the work area will inevitably be the same as those used to operate the mouse pointor, which is why the Adjusting state is raised. The states between Adjusting and others can be switched using the [00111] gesture (OK gesture). In order to minimise the risk of incorrect operations caused by hand movements or tremors, static gestures are used to adjust the operating area instead of dragging. In the Adjusting state, users can easily perform adjusting gestures without worrying about unintended operations. The corresponding gestures for these adjustment operations are listed in Fig. 6.

V. System Test and Discussion

We invited participants to test this virtual mouse system to measure its performance, learning curve and user acceptance. We first determined the unit of measurement of the target area for mouse clicks during the test. The size of the target area operated by the mouse is determined by the desktop application to which the operated object belongs. The smaller the target, the longer it takes to locate it with the mouse. Most UI controls are sized in pixels, so resizing a UI control requires adjusting the pixel values of its width or height accordingly. For mouse controls, however, the physical size of the target area is more important than its pixel size. For example, at resolutions of 1920×1080 and 800×600, an 80×80 target area at the former resolution is smaller than a 60×60 target area at the latter resolution, making it slightly more difficult to focus on using the mouse. Therefore, the target size here refers to the actual physical length, not the number of pixels.

A. User Experience Test

In order to test the learning effect of this virtual mouse system and its acceptance by new users, 27 participants were invited to take part in an experience and learning evaluation of the system. The test was conducted with 2 randomly varying items, the target area to be operated by the mouse and the specific action for each mouse operation. We randomly placed a square target area with sides in the range [30,80] on a 340×195 monitor screen, and randomly appeared an instruction in the square area that required the participant to operate the area. We set up only some of the most commonly used left-clicking, left-double-clicking and right-clicking instructions. The user operated on these target areas according to the instructions, and the next target area to be operated appeared only if the current operation was correct; otherwise the current target area was kept waiting for the user's correct operation. Each test consisted of 20 mouse operations. The operator first performed two tests with the physical mouse to use its average time as a comparison with the virtual mouse, then five tests were performed with the virtual mouse.

Based on the temporal data of the 27 participants' tests, we plotted the learning curve of the system, as shown in Fig. 8, which mainly expresses the mean value of the participants' time for each operation as well as its standard deviation.

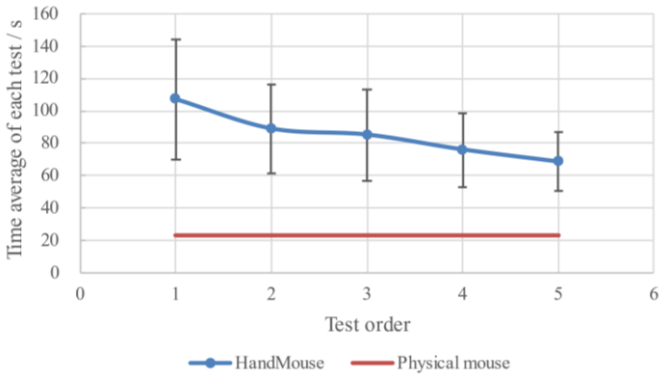

Fig. 8. The learning curve for the system. The blue curve is the learning curve based on the average time taken by the participant to perform the actions using HandMouse, and the red line is the referenced time taken to perform the actions using a physical mouse.

The time curves of the five virtual mouse tests for most participants did not always show a downward trend, but rather significant fluctuations. However, as we can see from Fig. 8, the average value of the test time has an overall decreasing trend, indicating that it still has a relatively positive learning effect. During the testing process, most of the new users would gradually understand and master the use of the system, although not very skillfully. At the same time, however, we can see that the standard deviation of each group's operation is relatively large, indicating that there are relatively large individual differences among the participants. A small number of participants experience a brief period of confusion and disorientation.

B. Subjective User Evaluation

After two physical mouse tests and five virtual mouse tests, each participant was asked to complete a survey to obtain a subjective evaluation of the gesture mouse system from new users. The survey questions were divided into two parts, one based on the NASA-TLX (NASA Task Load Index, https://humansystems.arc.nasa.gov/groups/TLX/) evaluation method, which asked questions about the feel of the operation. The second part compared the virtual mouse with the physical mouse to determine the user's preference between the two and the acceptance of the virtual mouse.

We collected the subjects' feelings from 6 aspects of Mental Demand, Physical Demand, Temporal Demand, Effort, Frustration Level, and Performance, and each of them has a corresponding question. Each item of the NASA-TLX is rated with a score of 20 points, and for each question, we counted the average of the 27 values that subjects rated, and the results are shown in Fig. 9.

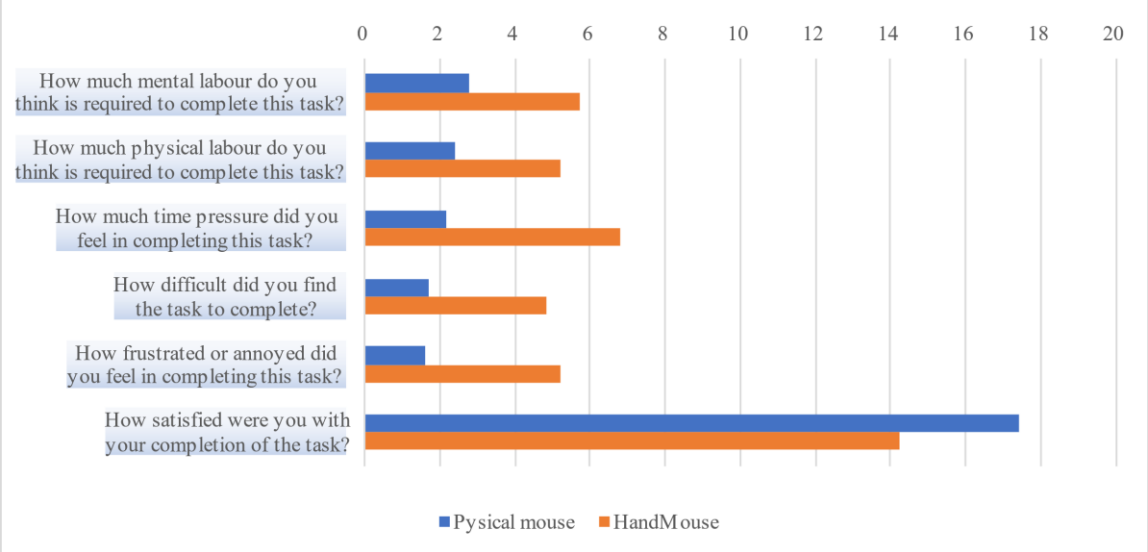

Fig. 9. Evaluation results on NASA-TLX.

The results of the second part of the survey content are shown in Table I, where 22 participants indicated that they would prefer to use this virtual mouse in an HCI system in a public environment, but unfortunately none of them preferred to use it in a work environment. Although this system has tried its best to perform as well as it can, it still has more mental and physical work in using it than a physical mouse. With the need for more precise results in the workplace, this virtual mouse system really isn't a good choice for new users. However, considering its performance, interesting, and technological effects, most of the participants gave it a relatively good rating as a whole.

TABLE I. AVERAGE TIME SPENT ON DIFFERENT OPERATIONS. THE UNIT OF ELAPSED TIME IS MILLISECONDS

Questions	Results
Do you prefer to use a physical mouse or this HandMouse during work or study?	0/27 body selected by HandMouse
Do you prefer to use a physical mouse or this HandMouse in public places such as shopping malls and museums?	22/27 bodies selected by HandMouse
How reasonable do you think the design of the gesture mouse is? (Score [0,20])	15.5 / 20
How satisfied are you with the gesture mouse compared to the physical mouse? (Score [0,20])	15.4 / 20
What is your overall score for this gesture mouse? (Considering rationality of design, technology, interesting, difficulty of operation, etc., Score [0,20])	16.8 / 20

C. Expert Test

To see how the HandMouse is used by experienced users and to explore the limits of its operation, we tested it with smaller target areas. We set the side lengths of the test squares to [6, 8, 12, 16, 20, 25, 30, 40, 50, 60]. Each square was randomly placed on the screen and appeared 20 times in succession. We performed 20 left-clicking operations on each square area and recorded the time taken to perform each operation. We obtained the average test results for different target areas, as shown in Table II and Table III. The open source code repository also contains the test program and the source data.

As can be seen from Table II and Table III, the efficiency of completing a mouse operation is related to the size of the target area and has an overall negative correlation. For the physical mouse, the maximum value of the time spent is about three times the minimum value. The median, mean, and average of Q1-Q3 (values in the middle 50%) are also close to each other, indicating that the physical mouse has great stability when operating on the target area. However, for the gesture mouse, when the target area is less than 12mm, the maximum value is more than 10 times the minimum value because it has one or two obviously large outliers. The overall average is significantly larger than the median and average of Q1-Q3 for the gesture mouse, and the median or average of Q1-Q3 is more representative, but at the same time, the instability of the gesture mouse's functionality should not be ignored.

TABLE II. TIME TAKEN OF PHYSICAL MOUSE OPERATIONS IN DIFFERENT AREAS

Object width / mm	Time taken / ms				
	Max	Min	Average	Median	Average of Q1-Q3
6	1,399	530	1,104	1,092	1,106
8	2,192	639	1,204	1,103	1,111
10	1,423	792	1,020	931	951
12	1,443	712	1,021	1,016	1,017
16	1,483	633	875	823	814
20	1,947	583	868	784	776
25	999	546	717	688	690
30	2,274	450	838	776	777
40	971	527	724	702	706
50	937	454	635	620	617
60	1,208	430	591	516	519

TABLE III. TIME TAKEN OF HANDMOUSE OPERATIONS IN DIFFERENT AREAS

Object width / mm	Time taken / ms				
	Max	Min	Average	Median	Average of Q1-Q3
6	35,464	2,690	7,278	3,991	4,114
8	17,009	1,624	4,884	4,034	4,037
10	18,390	1,876	5,271	3,952	3,889
12	18,526	1,799	4,873	2,957	2,964
16	3,822	1,369	2,412	2,374	2,330
20	8,997	1,512	3,094	2,465	2,516
25	6,693	1,203	2,538	2,289	2,345
30	6,513	1,323	2,391	2,280	2,198
40	2,803	930	1,838	1,876	1,832
50	35,464	2,690	7,278	3,991	4,114
60	17,009	1,624	4,884	4,034	4,037

When the target area is small, the standard deviation of the overall mean is larger for both the physical mouse and the gesture mouse, especially for the gesture mouse, and there are even 1-2 obvious outliers. There are two main reasons for the outliers: firstly, when the target area is small, it is really not easy to position the mouse pointer, and at this point the disadvantage of the gesture mouse is more obvious. Secondly, when the target area appears randomly, the user probably does not know where the target area is, and has to spend a certain amount of time to locate the target area, and then go to locate it.

The total interaction time with the gesture mouse is almost 3.2 times that of the physical mouse. The performance of the virtual mouse is significantly lower than that of the physical mouse. However, these differences may not be as pronounced when using the virtual mouse. The above data was obtained under the premise of assessing mouse performance, where the subject's attention was focused solely on the mouse operations and they aimed to perform the corresponding operations as quickly as possible to achieve the maximum performance of the mouse. In actual use, the mouse is merely a tool for performing specific tasks, and the mental effort required for these tasks may far outweigh the attention devoted to clicking on specific locations. Users often think about the logical steps or considerations in moving the mouse to the target position, without paying excessive attention specifically to mouse operations. However, speculation may not be accurate when the target area is smaller than 12-20 mm. In the course of testing the virtual mouse, subjects may have noticeable difficulty in focusing on the small target area, requiring considerable concentration on the target and repeated attempts to focus. In such cases, the efficiency of using the virtual mouse will be noticeably lower than that of a physical mouse, which may cause some anxiety.

Fortunately, as shown in Table III, the efficiency gap between the virtual mouse and the physical mouse decreases as the size of the target area increases. When operating on larger target areas, it becomes easier to achieve the same ease of use as with a physical mouse. Scenarios in which smaller target objects are manipulated are typically found in work-related environments, where people generally choose to use a physical mouse. The virtual gesture mouse is more suitable for HCI scenarios such as tourist attractions, shopping mall navigation and electronic exhibits in museums. In these cases, the size of the target objects should be larger, even up to 200mm, and then the efficiency gap between using a virtual mouse and a physical mouse or touch screen operation will be smaller. Therefore, the actual efficiency of this gesture mouse in practical use will be higher than in the test.

VI. CONCLUSIONS

Although AI has rapidly developed, gesture-based operations, as one of its applications, have not yet permeated various aspects of people's daily lives. They are only found in specific software or systems, such as intelligent car controls or sign language systems, with limited applications. Moreover, a universal gesture mouse applicable to all software is even rarer. This paper presents a set of gestures designed to replace the physical mouse, resulting in a gesture mouse system that achieves the basic functionality of a physical mouse. On personal computers, it can partially replace the physical mouse, but its performance is significantly lower than a physical mouse. While in public places with larger displays, it can serve as a viable alternative to physical mouse and touch screen operations.

However, the system also has noticeable disadvantages. Firstly, the efficiency of operating smaller target areas is significantly lower compared to a physical mouse, limiting its practical use in work scenarios. Additionally, the system currently only supports recognition of a single hand and does not consider the allocation of operating privileges in multi-user scenarios. For example, in situations where multiple users simultaneously give different control commands, the system does not know whose instructions to follow. While optimizing the former disadvantage may be difficult due to inherent human physiological characteristics, as hand tremors on the screen might already exceed the size of the target object, optimization is not currently prioritized. Nevertheless, there is

great potential for optimizing the latter disadvantage. In the future work, facial recognition of the operators may be implemented to determine the owner of the hand currently in control, thereby automatically allocating operating privileges.

DATA AVAILABILITY STATEMENT

This work has some supporting materials available on https://github.com/wanzhuxie/HandMouse-IJACSA.

REFERENCES

[1] Pavlovic V, Sharma R, Huang TS (1997): Visual Interpretation of Hand Gestures for Human-Computer Interaction: A Review. IEEE T PATTERN ANAL 19, 677-695

[2] Han X, Miao X, Liu Q, Li Y, Wan A (2022): A Fabric-Based Integrated Sensor Glove System Recognizing Hand Gesture. AUTEX RES J 22, 458-465

[3] Ozioko O, Dahiya R (2022): Smart Tactile Gloves for Haptic Interaction, Communication, and Rehabilitation. ADV INTELL SYST-GER 4

[4] Zhang YF, Liu B, Liu ZQ (2019): Recognizing Hand Gestures With Pressure-Sensor-Based Motion Sensing. IEEE T BIOMED CIRC S 13, 1425-1436

[5] Zhao Y, Lian C, Ren X, Xin L, Zhang X, Sha X, Li WJ (2021): High-Precision and Customized Ring-Type Virtual Keyboard Based on Layout Redesign. IEEE SENS J 21, 25891-25900

[6] Grif H, Farcas CC (2016): Mouse Cursor Control System Based on Hand Gesture. In: Moldovan L (Hrsg.), 9TH INTERNATIONAL CONFERENCE INTERDISCIPLINARITY IN ENGINEERING, INTER-ENG 2015, 9th International Conference on Interdisciplinarity in Engineering (INTER-ENG), pp. 657-661

[7] Maleki B, Ebrahimnezhad H (2015): Intelligent visual mouse system based on hand pose trajectory recognition in video sequences. MULTIMEDIA SYST 21, 581-601

[8] Wang Y, Jung C, Yun I, Kim J, IEEE (2019): SPFEMD: SUPER-PIXEL BASED FINGER EARTH MOVER'S DISTANCE FOR HAND GESTURE RECOGNITION, 2019 IEEE INTERNATIONAL CONFERENCE ON ACOUSTICS, SPEECH AND SIGNAL PROCESSING (ICASSP), 44th IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), pp. 4085-4089

[9] Ren Z, Yuan J, Meng J, Zhang Z (2013): Robust Part-Based Hand Gesture Recognition Using Kinect Sensor. IEEE T MULTIMEDIA 15, 1110-1120

[10] Li G, Tang H, Sun Y, Kong J, Jiang G, Jiang D, Tao B, Xu S, Liu H (2019): Hand gesture recognition based on convolution neural network. CLUSTER COMPUT 22, S2719-S2729

[11] Molchanov P, Yang X, Gupta S, Kim K, Tyree S, Kautz J (2016): Online Detection and Classification of Dynamic Hand Gestures with Recurrent 3D Convolutional Neural Networks. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 4207-4215

[12] Zhang J, Jiao J, Chen M, Qu L, Xu X, Yang Q (2017): A hand pose tracking benchmark from stereo matching. 2017 IEEE International Conference on Image Processing (ICIP), 982-986

[13] Zhang F, Bazarevsky V, Vakunov A, Tkachenka A, Sung G, Chang C, Grundmann M (2020): MediaPipe Hands: On-device Real-time Hand Tracking. ArXiv abs/2006.10214

[14] Cheok MJ, Omar Z, Jaward MH (2019): A review of hand gesture and sign language recognition techniques. INT J MACH LEARN CYB 10, 131-153

[15] Jiang S, Kang P, Song X, Lo BPL, Shull PB (2022): Emerging Wearable Interfaces and Algorithms for Hand Gesture Recognition: A Survey. IEEE REV BIOMED ENG 15, 85-102

[16] Xia P, You H, Ye Y, Du J (2023): ROV teleoperation via human body motion mapping: Design and experiment. COMPUT IND 150, 103959

[17] Sagayam KM, Hemanth DJ (2018): ABC algorithm based optimization of 1-D hidden Markov model for hand gesture recognition applications. COMPUT IND 99, 313-323

[18] Lee YH, Tsai CY (2009): Taiwan sign language (TSL) recognition based on 3D data and neural networks. EXPERT SYST APPL 36, 1123-1128

[19] Gobhiran A, Wongjunda D, Kiatsoontorn K, Charoenpong T (2022): Hand Movement-Controlled Image Viewer in an Operating Room by Using Hand Movement Pattern Code. WIRELESS PERS COMMUN 123, 103-121

[20] Benitez-Garcia G, Haris M, Tsuda Y, Ukita N (2022): Continuous Finger Gesture Spotting and Recognition Based on Similarities Between Start and End Frames. IEEE T INTELL TRANSP 23, 296-307

[21] Van Thanh T, Lee J, Kim D, Jeong Y (2016): Easy-to-use virtual brick manipulation techniques using hand gestures. J SUPERCOMPUT 72, 2752-2766

[22] Zhang W, Zhu F, Lu P, Li P, Sheng B, Mao L (2020): 3D Geology Scene Exploring Base on Hand-Track Somatic Interaction. In: Magnenat-Thalmann N et al. (Hrsg.), ADVANCES IN COMPUTER GRAPHICS, CGI 2020, 37th Computer Graphics International (CGI) Conference, pp. 364-373

[23] Toro-Guajardo S, Lizama E, Gutierrez FJ (2023): Gesture Coding: Easing the Introduction to Block-Based Programming Languages with Motion Controls. In: Bravo J, Ochoa S , Favela J (Hrsg.), PROCEEDINGS OF THE INTERNATIONAL CONFERENCE ON UBIQUITOUS COMPUTING & AMBIENT INTELLIGENCE (UCAMI 2022), Proceedings of the International Conference on Ubiquitous Computing & Ambient Intelligence (UCAmI), pp. 840-851

[24] Yang CK, Chen YC (2015): A HCI interface based on hand gestures. SIGNAL IMAGE VIDEO P 9, 451-462

[25] Roh M, Kang D, Huh S, Lee S (2017): A virtual mouse interface with a two-layered Bayesian network. MULTIMED TOOLS APPL 76, 1615-1638

[26] Mahdikhanlou K, Ebrahimnezhad H (2021): Object manipulation and deformation using hand gestures. J AMB INTEL HUM COMP

[27] Bush IJ, Abiyev R, Arslan M (2019): Impact of machine learning techniques on hand gesture recognition. J INTELL FUZZY SYST 37, 4241-4252

[28] Lugaresi C, Tang J, Nash H, McClanahan C, Uboweja E, Hays M, Zhang F, Chang C, Yong MG, Lee J, Chang W, Hua W, Georg M, Grundmann M (2019): MediaPipe: A Framework for Building Perception Pipelines. ArXiv abs/1906.08172

[29] Wu Y, Lin S, Lin J, Han C, Chang C, Jiang J (2022): Development of AI Algorithm for Weight Training Using Inertial Measurement Units. APPL SCI-BASEL 12

[30] Ramirez Sanchez JE, Anguiano Rodriguez A, Gonzalez Mendoza M (2021): Real-Time Mexican Sign Language Interpretation Using CNN and HMM. ADVANCES IN COMPUTATIONAL INTELLIGENCE (MICAI 2021), PT I 13067, 55-68

[31] Radmehr A, Asgari M, Masouleh MT (2021): Experimental Study on the Imitation of the Human Head-and-Eye Pose Using the 3-DOF Agile Eye Parallel Robot with ROS and Mediapipe Framework. 2021 9TH RSI INTERNATIONAL CONFERENCE ON ROBOTICS AND MECHATRONICS (ICROM), 472-478

[32] Latreche A, Kelaiaia R, Chemori A, Kerboua A (2023): Reliability and validity analysis of MediaPipe-based measurement system for some human rehabilitation motions. MEASUREMENT 214

[33] Gu F, Fan J, Cai C, Wang Z, Liu X, Yang J, Zhu Q (2022): Automatic detection of abnormal hand gestures in patients with radial, ulnar, or median nerve injury using hand pose estimation. FRONT NEUROL 13

[34] Siam AI, Soliman NF, Algarni AD, Abd El-Samie FE, Sedik A (2022): Deploying Machine Learning Techniques for Human Emotion Detection. COMPUT INTEL NEUROSC 2022

[35] Yasumuro M, Jin'No K (2022): Japanese fingerspelling identification by using MediaPipe. IEICE NONLINEAR THEO 13, 288-293

[36] Watanabe T, Maniruzzaman M, Al Mehedi Hasan M, Lee H, Jang S, Shin J (2023): 2D Camera-Based Air-Writing Recognition Using Hand Pose Estimation and Hybrid Deep Learning Model. ELECTRONICS 12

[37] Zhao X, Li Z, Zhao C, Wang C (2022): Real-Time Multistep Time-Series Prediction of Driver's Head Pose During IVIS Secondary Tasks

for Human-Machine Codriving and Distraction Warning Systems. IEEE SENS J 22, 24364-24379

[38] Dang TL, Nguyen HT, Dao DM, Nguyen HV, Luong DL, Nguyen BT, Kim S, Monet N (2022): SHAPE: a dataset for hand gesture recognition. NEURAL COMPUT APPL 34, 21849-21862

[39] Lopes DS, Parreira P, Paulo SF, Nunes V, Rego PA, Neves MC, Rodrigues PS, Jorge JA (2017): On the utility of 3D hand cursors to explore medical volume datasets with a touchless interface. J BIOMED INFORM 72, 140-149

[40] Chua SND, Chin KYR, Lim SF, Jain P (2022): Hand Gesture Control for Human-Computer Interaction with Deep Learning. J ELECTR ENG TECHNOL 17, 1961-1970

[41] Jia Y, Ding R, Ren W, Shu J, Jin A (2021): Gesture Recognition of Somatosensory Interactive Acupoint Massage Based on Image Feature Deep Learning Model. TRAIT SIGNAL 38, 565-572

[42] Szeghalmy S, Zichar M, Fazekas A, IEEE (2013): Comfortable mouse control using 3D depth sensor, 2013 IEEE 4TH INTERNATIONAL CONFERENCE ON COGNITIVE INFOCOMMUNICATIONS (COGINFOCOM), 4th IEEE International Conference on Cognitive Infocommunications (CogInfoCom), pp. 219-222

[43] Wang S, Hu H, Long H, Liu L, Chen Y, Wu Y, IEEE (2023): A Generalized Model for Non-Contact Gesture Interaction with Function Application Independence, 2023 IEEE CONFERENCE ON VIRTUAL REALITY AND 3D USER INTERFACES ABSTRACTS AND WORKSHOPS, VRW, 30th IEEE Conference Virtual Reality and 3D User Interfaces (IEEE VR), pp. 346-352

[44] Ko BK, Yang HS (1997): Finger mouse and gesture recognition system as a new human computer interface. COMPUTERS & GRAPHICS 21, 555-561

[45] Dogan RO, Dogan H, Kose C (2015): Virtual Mouse Control with Hand Gesture Information Extraction and Tracking. 2015 23RD SIGNAL PROCESSING AND COMMUNICATIONS APPLICATIONS CONFERENCE (SIU), 1893-1896