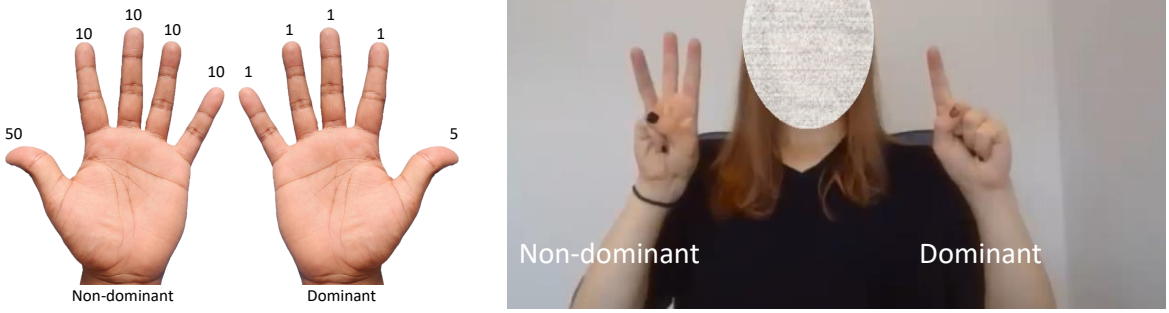
Abacus Gestures: A Large Set of Math-Based Usable Finger-Counting Gestures for Mid-Air Interactions

MD EHTESHAM-UL-HAQUE, Pennsylvania State University, USA
SYED MASUM BILLAH, Pennsylvania State University, USA



(a) (b)

Fig. 1. Illustrations of the proposed Abacus Gestures—a set of 100 mid-air gestures, each representing a number between 0 to 99—that users can make with ten fingers. (a) shows the mapping of base numbers (1, 5, 10, and 50) to different fingers: the worth of the *index*, the *middle*, the *ring*, or the *little* finger is *1* on the *dominant* hand and *10* on *the non-dominant* hand; and the worth of thumb is *5* and *50*, respectively. (b) shows how a user makes a series of Abacus Gestures in our system. For example, to make an Abacus Gesture, *31*, the user is facing a webcam and opening their index finger on the dominant hand (worth 1) and the index, middle, and ring fingers on the non-dominant hand (each worth 10), thus summing to 31 (=1+10+10+10).

Designing an extensive set of mid-air gestures that are both easy to learn and perform quickly presents a significant challenge. Further complicating this challenge is achieving high-accuracy detection of such gestures using commonly available hardware, like a 2D commodity camera. Previous work often proposed smaller, application-specific gesture sets, requiring specialized hardware and struggling with adaptability across diverse environments. Addressing these limitations, this paper introduces Abacus Gestures, a comprehensive collection of 100 mid-air gestures. Drawing on the metaphor of Finger Abacus counting, gestures are formed from various combinations of open and closed fingers, each assigned different values. We developed an

Authors' addresses: Md Ehtesham-Ul-Haque, Pennsylvania State University, University Park, Pennsylvania, USA, mfe5232@psu.edu; Syed Masum Billah, Pennsylvania State University, University Park, Pennsylvania, USA, sbillah@psu.edu.

2474-9567/2023/9-ART93 $15.00
https://doi.org/10.1145/3610898

algorithm using an off-the-shelf computer vision library capable of detecting these gestures from a 2D commodity camera feed with an accuracy exceeding 98% for palms facing the camera and 95% for palms facing the body. We assessed the detection accuracy, ease of learning, and usability of these gestures in a user study involving 20 participants. The study found that participants could learn Abacus Gestures within five minutes after executing just 15 gestures and could recall them after a four-month interval. Additionally, most participants developed motor memory for these gestures after performing 100 gestures. Most of the gestures were easy to execute with the designated finger combinations, and the flexibility in executing the gestures using multiple finger combinations further enhanced the usability. Based on these findings, we created a taxonomy that categorizes Abacus Gestures into five groups based on motor memory development and three difficulty levels according to their ease of execution. Finally, we provided design guidelines and proposed potential use cases for Abacus Gestures in the realm of mid-air interaction.

CCS Concepts: • **Human-centered computing → Gestural input**.

Additional Key Words and Phrases: Mid-air, gesture interaction, math, finger counting, abacus.

ACM Reference Format:
Md Ehtesham-Ul-Haque and Syed Masum Billah. 2023. Abacus Gestures: A Large Set of Math-Based Usable Finger-Counting Gestures for Mid-Air Interactions. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 7, 3, Article 93 (September 2023), 30 pages. https://doi.org/10.1145/3610898

1 INTRODUCTION

Recent technological advancements in virtual reality, augmented reality, smart glasses, and large displays have propelled mid-air interactions into the spotlight [39]. These interactions hinge on gestures that the system can detect accurately, and that users can learn, remember, and execute easily. Moreover, to support various tasks and actions, like navigation, selection, and activation, these gestures must possess sufficient expressiveness [39, 56]. However, the creation of a large repertoire of such gestures presents significant challenges from *technical*, *usability*, and *design* perspectives. Technically, the system must recognize these gestures with high accuracy and reduce the incidence of false positives [65]. In terms of usability, the gestures must be simple, socially acceptable, and allow users to learn, remember, and execute them with ease [61]. From a design perspective, the limited availability of relevant real-world metaphors complicates the creation of a large set of such gestures [46, 70]. Assessing these gestures' effectiveness—for instance, recognition accuracy, ease of use, and learnability—without reliance on specific applications also presents a design challenge [56].

Prior work sought to improve the recognition accuracy of mid-air gestures by employing specialized hardware, such as 3D hand trackers and body-mounted devices [39]. However, the need for such hardware could limit their broad use, as not all users might have access to these devices, thereby potentially hampering the practicality and usability of mid-air gestures in everyday applications. To tackle usability and design challenges linked to mid-air gestures, researchers have primarily utilized gesture elicitation studies [56]. In these studies, users envision new gestures in response to a shown action without considering how a system could recognize these gestures. An agreement score is then used to amalgamate all user-envisioned gestures into a final set [85]. Such a process tends to yield gestures that are generally easy to learn, remember, and execute; however, it often results in sets that are not only impractical but also limited—typically, to around 20 gestures. This constraint narrows the range of expressiveness, thereby limiting the the practicality and usability of mid-air gestures in real-world applications [39, 56].

This paper aims to address the aforementioned challenges. We introduce ***Abacus Gestures***, a comprehensive set of 100 mid-air gestures, which are easily detectable by a single commodity camera. This set, based on the opening and closing of fingers, draws its metaphor from the classical finger abacus counting technique, known as Chisanbop or Fingermath [45, 81]. Users can represent numbers from 0 to 99 by various combinations of open and closed fingers, with each number interpreted as a distinct gesture. As illustrated in Figure 1a, Finger Abacus assigns specific values to each of the ten fingers. On the dominant hand, every finger is worth 1, except

for the thumb, which is valued at 5. Conversely, each finger on the non-dominant hand has a value of 10, with the exception of the thumb, which is worth 50. An open finger contributes its value to the count; a closed finger does not. The final number, i.e., an abacus gesture, is the sum of the values of all open fingers.

Finger Abacus, as a mathematical counting technique, possesses several distinctive and beneficial properties. *First*, counting and performing arithmetic operations with Finger Abacus is inherently a bimanual mid-air process. Users hold their hands aloft, open or close their fingers, and count [45]. This bimanual process naturally suits hands-free interaction domains, such as virtual reality. *Second*, this technique permits users to perform simple arithmetic operations ubiquitously, without any equipment [2, 49]. These gestures can be detected using a single camera, for example, using palm gestures to take selfies on smartphones [4]. This detection can serve as an input to camera-based mid-air interaction systems, a possibility we examine in this work. *Third*, Finger Abacus is simple and easy to learn, even before the formal introduction of arithmetic notations [2, 53, 55, 71]. Studies have revealed that children who master finger-based number representation from an early age tend to develop significantly advanced mathematical skills over time [8, 24]. *Finally*, Finger Abacus enables users to perform rapid arithmetic operations, such as the manipulation of two-digit numbers simultaneously [3] and the addition of 20 single-digit numbers [1].

To detect Abacus Gestures, we designed an algorithm (Section 4) that uses 21 keypoints for each hand—one for the wrist and four for each finger—provided by an off-the-shelf computer vision library [91]. We evaluated the detection accuracy and usability of Abacus Gestures by conducting a user study with 20 participants (Section 5).

Our results revealed that this algorithm could detect Abacus Gestures with an accuracy surpassing 98% when the palms face the camera and 95% when the backs of the hands face the camera. Additionally, the algorithm is rotation invariant (along the X-Y plane) and can accommodate a variety of hand shapes and finger dexterity. Our study also discovered that participants were able to learn the necessary finger combinations to perform Abacus Gestures rapidly – in just five minutes after executing 15 gestures. They also developed motor memory for most of the Abacus Gestures shortly after performing 100 Abacus Gestures.

Drawing on the findings, we devised a taxonomy (Section 6) divided into five categories—visual match, mnemonic, unimanual, bimanual symmetric, and bimanual asymmetric (Table 2)—and three difficulty levels: easy, medium, and hard (Table 3). This taxonomy associates the ease with which users can develop motor memory for different Abacus Gestures and groups various gestures by their difficulty levels. Such categorization will inform HCI researchers and system designers in assigning Abacus Gestures with specific actions or tasks. For instance, users are more likely to quickly develop motor memory for gestures that visually match their corresponding finger combinations, such as 0, 1, 2, 3, and 4, which are also the simplest gestures to perform. Furthermore, we propose several scenarios in which Abacus Gestures can facilitate mid-air interaction (Section 7.2).

Finally, to assess the memorability of our gestures, we sent a follow-up online survey to the participants four months after the study (Section 6.1.3). The responses confirmed that all participants could vividly recall the Abacus Gestures, with some even adopting these gestures for their routine mental arithmetic operations. These results strongly indicate that the proposed gestures, despite comprising a large set, are easy to learn, remember, and execute.

In summary, this work contributes in the following ways:

- We proposed Finger Abacus as a metaphor to design a large-scale, math-based gesture set for mid-air interaction.
- We developed a rotation and scale invariant algorithm to detect Abacus Gestures with commodity cameras.
- We conducted an evaluation of Abacus Gestures with 20 participants and categorized the Abacus Gestures according to their ease of motor memory development and ease of execution.
- We provided design guidelines and potential use cases related to the application of Abacus Gestures for mid-air interaction.

2 BACKGROUND AND RELATED WORK

This section describes prior research on number-counting techniques, gesture design challenges, and the detection of mid-air gestures.

2.1 Number Counting Techniques

The ability to comprehend numbers is a basic human instinct, inherent even in the early stages of human development [26]. The human brain is naturally capable of processing numbers and understanding arithmetic operations, a faculty often referred to as 'Number Sense' [27]. Counting, a fundamental arithmetic operation, emerges naturally at a young age, even without formal numerical knowledge [31]. Given the ubiquity and necessity of number processing, various counting methods and tools have been proposed and utilized since ancient times.

2.1.1 Abacus. The abacus is a mathematical tool that predates the adoption of written numeral systems [80]. It comprises rows of movable beads on wires, with each bead representing a number for performing arithmetic operations such as counting, addition, and subtraction. Different versions of the abacus have been utilized worldwide for mathematical calculations.

Studies suggest that learning to calculate with an abacus can enhance mental calculation capacity. Even without a physical abacus, individuals can perform mental calculations by visualizing an abacus, a method known as Abacus-based mental calculation (AMC) [79, 80]. AMC is a high-level cognitive skill, and long-term AMC training has been associated with improved numerical memory capacity [36, 87] and effective memory retrieval for complex tasks [20, 43, 80].

In contemporary times, the abacus is often used in preschools and elementary schools as a teaching tool for numerical systems and arithmetic operations. Moreover, it remains in use as a scoring device for table games and as a counting tool for visually impaired individuals who cannot use calculators [16].

2.1.2 Finger Abacus. Finger Abacus, also known as Chisanbop or finger math, is a finger-counting technique originating from Korea that draws inspiration from the Abacus system [45, 81]. This method assigns specific values to each of the ten fingers, enabling users to count numbers from 0 to 99 and perform arithmetic operations [53].

While Finger Abacus only accommodates two-digit numbers, a limitation compared to a physical abacus, it allows users to perform simple arithmetic operations without any additional tools. This makes it sufficient for everyday calculations and beneficial for teaching children arithmetic operations. Moreover, finger counting has been found to be a powerful tool that aids in teaching complex numerical tasks to children [8, 24].

Experienced users of Finger Abacus can reach a level of speed and efficiency comparable to that of physical abacus users [3, 23]. These finger-counting techniques facilitate swift mental calculations and are utilized by participants in mental math competitions, such as the Mental Calculation World Cup [3, 83, 84].

In the present study, we propose the various finger combinations from the Finger Abacus technique as a large, usable gesture set for mid-air interaction.

2.2 Finger-Counting Gestures: A Primitive Finger Abacus Technique

Finger counting gestures, which can be considered a primitive version of the Finger Abacus, involve people counting numbers using their open fingers (the maximum number possible is 10).

2.2.1 Applications. The most prevalent application of finger-counting gestures is in selecting a menu item [10–13, 18, 42, 90]. Each menu item is assigned a number (for example, from 1 to 4), and users raise the corresponding number of fingers to select it.

Menus can contain up to 10 items in various forms. For instance, on distant TVs, the menu items may include play or pause media controls [13, 42]. In a car's dashboard, the menu items could be controls for media, temperature, fan speed, or navigation [90], while touchless kiosks offer a range of on-screen items [13].

In addition, users can select items from hierarchical menus. For example, users can first select one of the five top-level menu items by raising [13] or touching the display [10, 11] with 1 to 5 fingers of their non-dominant hand. After selecting the top-level menu item, users can then choose one of its five sub-menu items by raising or touching the display with 1 to 5 fingers of their dominant hand.

2.2.2 Gesture Recognition. Various techniques are employed to recognize the number of raised fingers in prior work, including i) multi-touch displays [10, 11]; ii) RGB cameras [42]; iii) RGB cameras with finger-mounted tags or augmentations, such as stickers [40] or LED lights [18]; iv) RGBD cameras (RGB cameras with depth sensors) [12], like the Microsoft Kinect [5]; and v) IR cameras with depth sensors [90], such as the Leap Motion [6].

However, previous research on finger-counting gestures has been limited to smaller gesture sets (10 or fewer gestures). Moreover, gesture recognition often poses practical difficulties, as some prototypes require special hardware or finger augmentations. Finally, gestures are typically tied to a specific application domain or form factor, leaving the potential for appropriating these gestures for other applications largely unexplored.

This paper expands on previous work by increasing the gesture set size tenfold (from 10 to 100), only requiring a commodity RGB camera, and studying gestures in isolation, without tying them to any specific application.

2.3 Mid-Air Gestures: A Superset of Abacus Gestures

As interactive computing continues to evolve, characterized by Virtual and Augmented Reality headsets, large displays, smartwatches, and smart glasses, traditional input methods like physical keyboards become less practical or even unusable. This has led to the emergence of mid-air gestures as a viable alternative. Mid-air gestures, involving a variety of finger, hand, and body configurations and orientations, form a set that is significantly broader than that of Abacus Gestures. Note that Abacus Gestures itself represents a superset of the more rudimentary finger counting gestures. It is not surprising that real-time detection of generic mid-air gestures demands robust hardware and software solutions [39, 63]. Such solutions include 3D external trackers with depth sensors, for example, Leap Motion [6] and Microsoft Kinect [5], as well as Motion-Capture Systems like Optitrack and Vicon [51, 52, 77, 78].

2.3.1 The Positions of Trackers and Cameras. Achieving high accuracy in mid-air gesture recognition often requires strategic positioning of trackers to avoid occlusion and cater to specific use cases. Although trackers are typically positioned vertically, either above or below a display, to face the user, they can also be mounted behind the user [28]. In some scenarios, trackers are attached directly to the user's body, such as on the head [48], hands (e.g., gloves [44, 47, 60]), forearms [38], fingertips [19, 21, 88, 89], thigh [46], pockets [46, 67], chest [54], shoulder [32], or even shoes [12].

However, these 3D trackers may not always be readily available and often require users to wear additional accessories, obstructing hands-free interaction. Our goal is to carve out a subset of mid-air gestures that are expressive yet simple enough to be detected using a single 2D commodity camera, such as a webcam, without the need for any additional markers or accessories. The widespread availability of such commodity cameras suggests significant potential for the implementation and use of Abacus Gestures across a variety of devices and environments.

2.4 Challenges in Gesture Design

Designing a gesture set presents two key challenges: i) ensuring new gestures are easy to use, learn, and remember; and ii) determining whether a set of gestures is suitable for a task. To address the first challenge, researchers often resort to metaphorical gesture design or gesture elicitation studies.

2.4.1 Metaphorical Gestures. Many past studies have focused on designing gestures based on familiar and natural metaphors. Typically, these metaphors are chosen for comfort, familiarity, and ease of use, helping users to learn and memorize the resulting gestures easily. For instance, Liu et al. [46] utilized the "Gunslinger" metaphor, where users keep their hands down and make comfortable micro-gestures with their fingers. The gestures are detected using a pair of thigh-mounted Leap Motion devices. Song et al. [70] used a handle-bar metaphor for 3D virtual object manipulation, where users manipulate 3D objects using an imaginary handlebar passing through the object. A Microsoft Kinect sensor was used to detect users' hand movements. In this work, we utilize the Finger Abacus as a mathematical metaphor for designing a large gesture set.

2.4.2 Gesture Elicitation Studies. A Gesture Elicitation Study [85] is a participatory design technique where end users are individually shown the desired effect of an action or the referent. Then, they are asked to propose an appropriate gesture for that effect, referred to as a symbol. The results from all participants are reconciled using an agreement metric to create a single gesture [56, 85]. This technique has been applied to a wide range of emerging interaction and sensing technologies [50, 56], such as surface computing [57, 85], mobile interaction [66], and smart television [29, 75].

Gestures are typically elicited in a specific physical environment for determined devices, such as the steering wheel in a car [30]. The location where gestures are issued also plays a significant role (e.g., social acceptance [62]) in the resulting gesture set. Therefore, migrating from one environment to another using the same gesture set with different devices may be challenging [50, 74]. Gesture sets resulting from elicitation studies can also suffer from legacy bias, where participants propose gestures based on their prior use of technologies. This bias can limit creativity and underutilize the capabilities of the system for which the study is being conducted [56]. Moreover, finding a common set of agreed-upon gestures can be challenging, reducing the size of the resulting gesture set [39, 50]. Finally, most gesture sets resulting from elicitation studies are limited in size (less than 30) and are designed for a specific application or device. We aim to go beyond this limit.

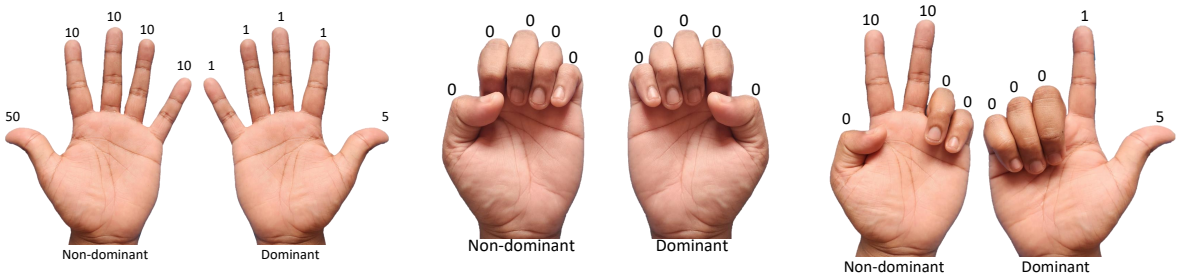
2.4.3 Evaluation of Gestures. Previous work has examined different evaluation metrics to determine whether a set of gestures is appropriate for a task. These metrics fall into two general categories: i) cognitive dimensions, which relate to the users' mental model; and ii) motor dimensions, which pertain to the ergonomic characteristics of the gestures [39]. We use these metrics to evaluate our proposed gestures.

Among the cognitive dimensions, one well-studied metric is the intuitiveness or naturalness of gestures. These are gestures that enable users to use an application with little or no instructions [58]. Jahani et al. [37] investigated the effect of natural and non-intrusive mid-air gestures for secondary tasks while the primary task was driving. Bostan et al. [15] showed that intuitive gestures were easily memorable by investigating hand-specific on-skin gestures. Researchers also explored discoverability [59, 64], learnability [22, 86], and gesture simplicity [17, 73] when designing gesture sets [39].

In terms of the motor dimension, prior work has investigated ergonomic characteristics [39]. For instance, Ruiz et al. [66] looked into the ease of repeating each gesture multiple times. Other well-explored ergonomic metrics include comfort [9, 15, 62, 86], perceived fatigue [15, 22, 68], body part suitability [69], the ability to perform concurrent gestures during intense gameplay [69], and the usage of unimanual or bimanual gestures [7].

3 OVERVIEW OF ABACUS GESTURES

The Finger Abacus technique assigns two states to each finger: *open* or *closed*. When closed, all fingers on both hands have a value of zero. When open, the thumb is worth five, and the remaining four fingers are worth one each on the dominant hand. On the non-dominant hand, the thumb is worth fifty when open, and the other four fingers are worth ten each. Users can create different numbers by opening and closing various combinations of fingers on both hands. The sum of the values of all open fingers corresponds to the represented number.



(a) Count = 99 (b) Count = 0 (c) Count = 26

Fig. 2. Counting different numbers with Finger Abacus. (a) The maximum count is 99 when all fingers in both hands are in the open state. (b) The minimum count is 0 when all fingers in both hands are in the closed state. (c) Counting the number 26 requires the non-dominant Index (worth 10) and Middle (worth 10) fingers, along with the dominant Thumb (worth 5) and Index (worth 1) fingers in the open state. All other fingers are kept in the closed state.

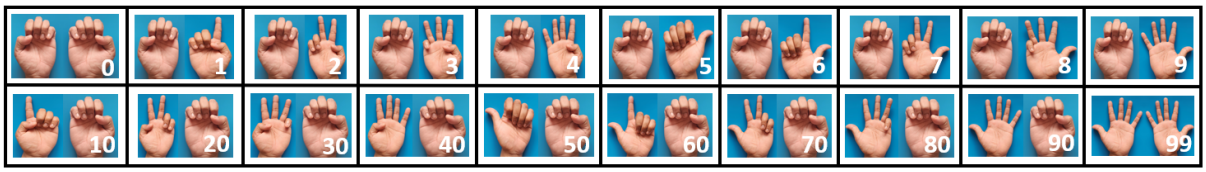


Fig. 3. A subset of Abacus Gestures and their corresponding finger combinations.

Figure 2a demonstrates the combined value of all fingers in the open state, which corresponds to the number '99'. Conversely, with all fingers in the closed state (Figure 2b), the represented number is '0'. Figure 2c illustrates how to represent the number 26 with the Finger Abacus. By opening the index (worth 10) and middle (worth 10) fingers on the non-dominant hand and the thumb (worth 5) and index (worth 1) fingers on the dominant hand, the resulting number is $10 + 10 + 5 + 1 = 26$.

In total, users can represent 100 numbers, ranging from 0 to 99, with different combinations of open and closed fingers. Note that multiple finger combinations can represent the same number as long as the sum of the values of the open fingers is equal to that number. For example, opening either the index or the little finger on the dominant hand while keeping all other fingers closed also represents the number '1'. Similarly, another way to represent 26 is to open the index (worth 10) and little (worth 10) fingers on the non-dominant hand and the thumb (worth 5) and ring (worth 1) fingers on the dominant hand. Figure 3 provides a subset of Abacus Gestures and their corresponding finger combinations.

4 ALGORITHM FOR DETECTING FINGER ABACUS GESTURES

Detecting Abacus Gestures poses several key challenges: (i) extracting reliable features or keypoints representing each hand; (ii) ensuring rotation and orientation invariance for the keypoints; (iii) discerning how keypoints interact in different finger states (e.g., open or closed); (iv) selecting a suitable gesture delimiter to differentiate intended gestures from unintended ones; and (v) accommodating the variability in users' finger dexterity.

In this section, we describe the challenges in detail and outline our algorithm's approach to address these challenges.

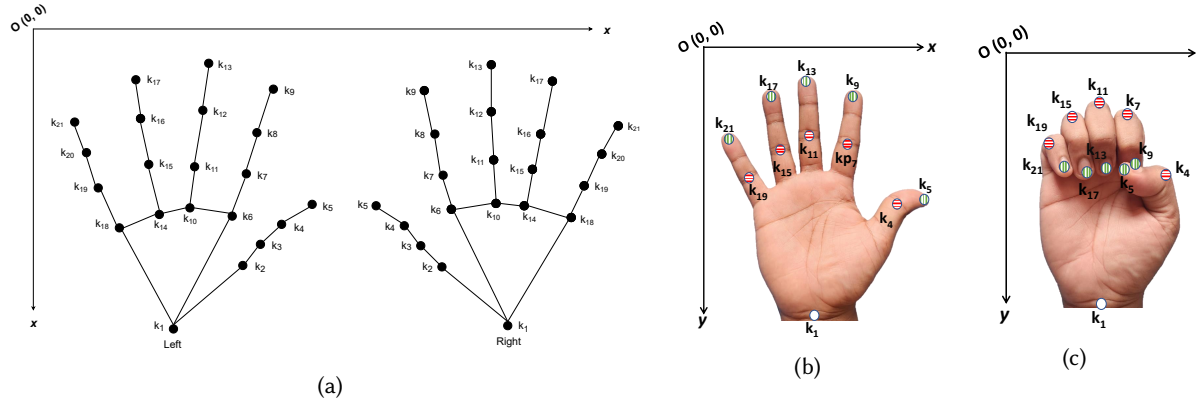


Fig. 4. (a) The Mediapipe Hands module [91] detects 21 keypoints (k) for both the left and right hands. (b) The image depicts the position of keypoints in the open state. The five keypoints marked by circles with red horizontal lines - k_4, k_7, k_{11}, k_{15}, and k_{19} - serve as reference points for the thumb, index, middle, ring, and little fingers respectively. The five keypoints marked by circles with green vertical lines - k_5, k_9, k_{13}, k_{17}, and k_{21} - signify the tips of the respective fingers. The keypoint k_1 positioned at the wrist functions as the base of the hand. In this open state, the tip keypoints maintain a greater distance from the base compared to the reference keypoints. (c) The illustration depicts the position of the same five reference points and the five tip points in the closed state. Here, the tip keypoints are positioned closer to the base than the reference keypoints.

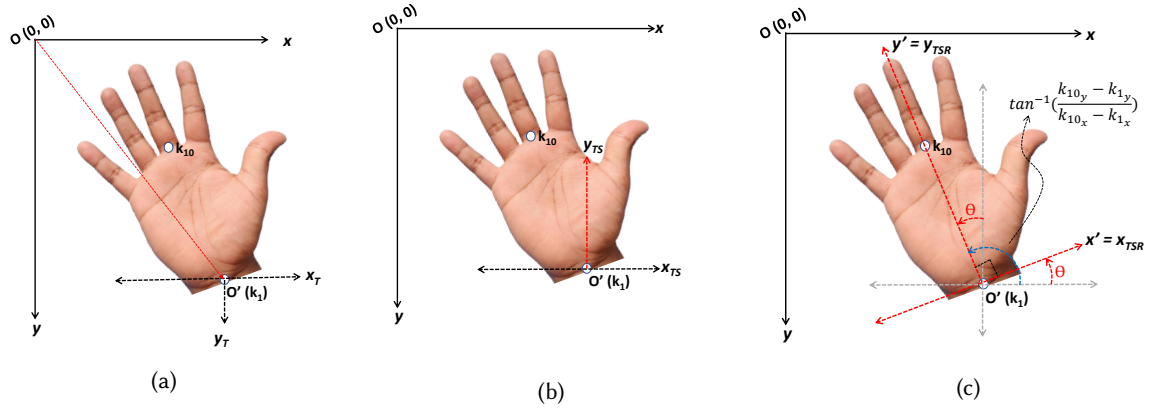


Fig. 5. Transformation of the camera coordinate system to the hand-relative coordinate system. (a) A translation operation that moves the origin to the new origin at the wrist (k_1). (b) An scale operation that shifts the Y-axis. (c) A rotation operation that aligns both the X- and Y-axes passing through the points k_1 and k_{10}.

4.1 Reliable Feature Extraction and Keypoint Representation

Our initial challenge involves extracting a robust set of features and keypoints to offer an accurate representation of each hand. For this purpose, we utilize the Mediapipe Hand Landmarks model [91], a widely available, pre-trained computer vision library. This model can extract 21 keypoints from each hand in real-time from a continuous 2D camera feed, with a precision of over 95%. It can detect users' palms, as well as discern between left and right hands.

Figure 4a depicts these 21 keypoints relative to the origin O, positioned at the top-left corner. We represent these keypoints as vectors $\boldsymbol{k_1}$ through $\boldsymbol{k_{21}}$. These keypoints correspond to various anatomical landmarks such as the wrist, fingertips, finger bases, and middle finger joints. For instance, $\boldsymbol{k_1}$ represents the wrist, while $\boldsymbol{k_5}$, $\boldsymbol{k_9}$, $\boldsymbol{k_{13}}$, $\boldsymbol{k_{17}}$, and $\boldsymbol{k_{21}}$ represent the tips of the thumb, index, middle, ring, and little fingers, respectively.

Each keypoint vector $\boldsymbol{k_i}$ manifests as a three-dimensional point $\left(k_{i_x}, k_{i_y}, k_{i_z}\right)$ within the current camera frame, relative to the origin O. Here, k_{i_x} indicates the value along the X-axis, while k_{i_y} indicates the value along the Y-axis. Additionally, k_{i_z} denotes the distance of $\boldsymbol{k_i}$ from the X-Y plane.

However, as the depth information from a standard camera tends to be unreliable, our algorithm currently dismisses k_{i_z}, replacing it with 1, thereby making $\boldsymbol{k_i} = \left(k_{i_x}, k_{i_y}, 1\right)$. This adjustment allows us to represent $\boldsymbol{k_i}$ as a 2D affine vector, which is required to transform keypoint vectors into a rotation-invariant, hand-relative coordinate system, as described in the next section.

4.2 Transforming Keypoints to a Rotation Invariant Hand-Relative Coordinate System

The next challenge is to ensure the rotation invariance of the keypoints on the X-Y plane. We accomplish this through an affine transformation comprising a translation, scaling, and rotation of all keypoints.

First, we represent 21 keypoint vectors in a $\boldsymbol{K}$ matrix (shown below). Next, we calculate the transformation matrices as follows. The *translation* matrix $\boldsymbol{T}$ initially shifts the origin from O to the wrist keypoint vector, $\boldsymbol{k_1}$ as shown in Figure 5a. The *scale* matrix $\boldsymbol{S}$ inverts the direction of the Y-axis (Figure 5b). If the back of the hand faces the camera, it also inverts the direction of the X-axis, as indicated by the -1 value of λ. The *rotation* matrix $\boldsymbol{R}$ rotates all keypoint vectors by an angle, $\theta = \tan^{-1}(\frac{k_{10_y} - k_{1_y}}{k_{10_x} - k_{1_x}}) - \frac{\pi}{2}$, such that the line passing through the center of the hand aligns with the Y-axis (Figure 5c). This sequence of transformations results in a new keypoint matrix, $\boldsymbol{K'} = \boldsymbol{K} \cdot \boldsymbol{T}^T \cdot \boldsymbol{S}^T \cdot \boldsymbol{R}^T$, where $\boldsymbol{k_1'}$ to $\boldsymbol{k_{21}'}$ vectors are now defined with respect to a hand-relative coordinate system, originating at O′. Note that $\boldsymbol{T}^T$, $\boldsymbol{S}^T$, and $\boldsymbol{R}^T$ denote the transpose of matrices $\boldsymbol{T}$, $\boldsymbol{S}$, $\boldsymbol{R}$, respectively.

$$\boldsymbol{K} = \begin{bmatrix} \boldsymbol{k_1} \\ \boldsymbol{k_2} \\ \boldsymbol{k_3} \\ \vdots \\ \boldsymbol{k_{21}} \end{bmatrix}; \quad \boldsymbol{T} = \begin{bmatrix} 1 & 0 & -k_{1_x} \\ 0 & 1 & -k_{1_y} \\ 0 & 0 & 1 \end{bmatrix}; \quad \boldsymbol{S} = \begin{bmatrix} \lambda & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 1 \end{bmatrix}; \quad \boldsymbol{R} = \begin{bmatrix} cos\theta & sin\theta & 0 \\ -sin\theta & cos\theta & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

Here, λ is 1 if the palm faces the camera and -1 if the back of the hand faces the camera.

4.3 Keypoint Interaction and Finger State Recognition

To understand the interaction of keypoints under different finger states (open or closed), we scrutinize the spatial relationship between the keypoints corresponding to each finger. This relationship is demonstrated in Figure 6a and Figure 6c for the palm facing the camera. Figure 6b and 6d illustrate the same relationship when the back of the hand faces the camera. For both of the camera perspectives, the distances of the keypoints from their reference axes can be measured in the same way. We compare the distance between the tip of the finger and its reference following the rules detailed in Table 1, further elaborated in Algorithm 2.

Regardless of the perspective of the hand, a positive distance between the tip and the reference point from the reference axis denotes an open state, while a negative distance signifies a closed state. As these distances are calculated in the transformed, hand-relative coordinate system that invariably rotates with the hand in the X-Y plane, this detection system is effective across any hand rotation in the X-Y plane and any hand perspective. This system also accurately identifies gestures under changing hand perspectives—for instance, when one palm faces the camera while the other faces the body. It maintains invariance even under dynamic changes in perspective.

For example, the system can successfully detect a sequence where a user presents a 'thumbs up' gesture for the number 5, initially with the palm facing away from the camera and then reorienting it to face the camera.

4.4 Gesture Delimiter Selection

Our fourth challenge involves determining an appropriate gesture delimiter to distinguish between intended and unintended gestures. We introduce a "neutral pose" as a delimiter, where all fingers must be closed both before and after making a gesture. This neutral pose ensures that any Abacus Gesture stands out distinctly.

For example, to input the number 56 using Abacus gestures, users would first form the corresponding finger combination, then close all fingers to delimit the gesture. Users can alter their finger combinations until they reach the intended number, with the neutral pose acting as a delimiter.

Selecting the neutral pose as a delimiter is advantageous for three reasons. Firstly, gesturing in mid-air can be physically taxing. By allowing users to close all their fingers and rest their hands between gestures, we help minimize fatigue [46]. Secondly, as Finger Abacus is essentially a finger-counting method, there should be a clear distinction between counting fingers and finalizing the count. Users should be able to count their fingers incrementally and adjust the total until they reach the desired number. By closing all fingers and making the neutral pose, users can finalize their count. Lastly, returning to the neutral pose resets the count to zero, allowing users to start the next gesture naturally from zero without carrying over the previous count.

Despite these advantages, we noted that the neutral pose can occasionally lead to incorrect gesture recognition. This can occur when the fingers of both hands do not close synchronously, causing the system to detect an

Table 1. The open condition of 10 fingers for Finger Abacus gestures, including the reference and tip keypoints, reference axis, and the fingers' worth in open condition. Given the mutually exclusive nature of finger states, a finger that is not open is considered closed.

Hand	Finger	Reference Keypoint (after transformation)	Tip Keypoint (after transformation	Reference Axis (after transformation	Open Condition	Finger Worth
Right	Thumb	$\boldsymbol{k}_{4}^{'}$	$\boldsymbol{k}_{5}^{'}$	Y-axis	$k_{5_x}^{'} - k_{4_x}^{'} > 0$	5
	Index	$\boldsymbol{k}_{7}^{'}$	$\boldsymbol{k}_{9}^{'}$	X-axis	$k_{9_y}^{'} - k_{7_y}^{'} > 0$	1
	Middle	$\boldsymbol{k}_{11}^{'}$	$\boldsymbol{k}_{13}^{'}$	X-axis	$k_{13_y}^{'} - k_{11_y}^{'} > 0$	1
	Ring	$\boldsymbol{k}_{15}^{'}$	$\boldsymbol{k}_{17}^{'}$	X-axis	$k_{17_y}^{'} - k_{15_y}^{'} > 0$	1
	Little	$\boldsymbol{k}_{19}^{'}$	$\boldsymbol{k}_{21}^{'}$	X-axis	$k_{21_y}^{'} - k_{19_y}^{'} > 0$	1
Left	Thumb	$\boldsymbol{k}_{4}^{'}$	$\boldsymbol{k}_{5}^{'}$	Y-axis	$k_{5_x}^{'} - k_{4_x}^{'} > 0$	50
	Index	$\boldsymbol{k}_{7}^{'}$	$\boldsymbol{k}_{9}^{'}$	X-axis	$k_{9_y}^{'} - k_{7_y}^{'} > 0$	10
	Middle	$\boldsymbol{k}_{11}^{'}$	$\boldsymbol{k}_{13}^{'}$	X-axis	$k_{13_y}^{'} - k_{11_y}^{'} > 0$	10
	Ring	kp_{15}	$\boldsymbol{k}_{17}^{'}$	X-axis	$k_{17_y}^{'} - k_{15_y}^{'} > 0$	10
	Little	$\boldsymbol{k}_{19}^{'}$	$\boldsymbol{k}_{21}^{'}$	X-axis	$k_{21_y}^{'} - k_{19_y}^{'} > 0$	10

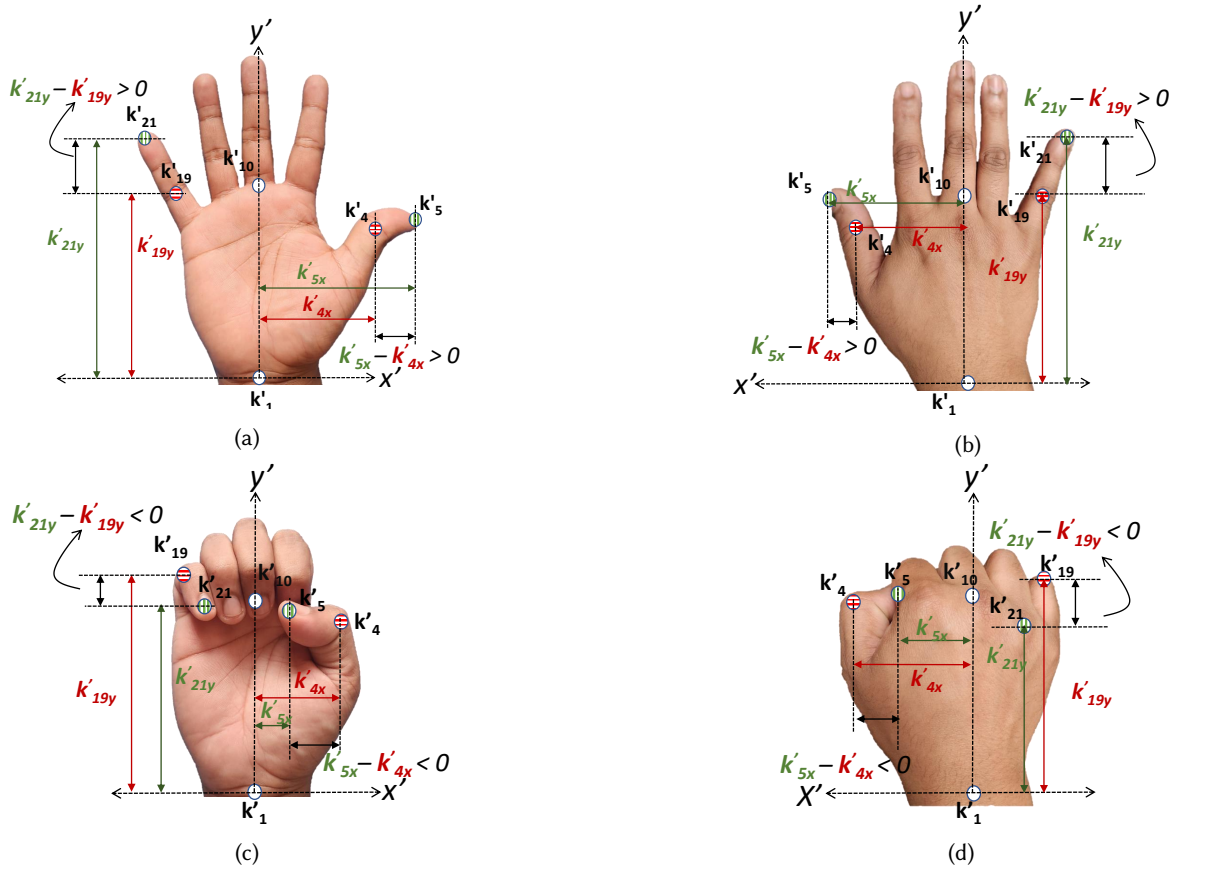


Fig. 6. Determining the state of a finger as either open or closed. (a)-(b) Illustrations show the positions of the tip and reference keypoints when all fingers are open for both hand perspectives - palm facing the camera and palm facing the body. Regardless of hand perspective, the distance to the tip is consistently greater than the distance to the reference for all five fingers. (c)-(d) The figures depict the positions of the tip and reference keypoints when all fingers are closed, again for both hand perspectives. Irrespective of the perspective of the hand, the distance to the tip is consistently less than the distance to the reference for all five fingers.

unintended gesture just before reaching the neutral pose. To mitigate this, we introduce a stabilization threshold – a short time interval during which a gesture must remain consistent to be considered stable. This threshold should be small enough to allow for rapid response to intentional gesture changes but large enough to prevent accidental changes. The appropriate threshold will depend on the frame rate of the system processing the camera feed.

Algorithm 1 (in the Appendix) outlines the process of gesture delimitation, using the camera feed CF and a stabilizer threshold ΔT as inputs. The algorithm continuously captures frames from the camera feed, and each frame at time t ($frame_t$) is processed by the computer vision algorithm to generate 21 keypoints for each hand (Figure 4a). These keypoints are then transformed to the hand-relative coordinates and passed to Algorithm 2 to calculate the Abacus Gesture count. When the neutral pose is detected, the last stable gesture is passed to the system. For a gesture to be considered stable, it must remain consistent for at least ΔT frames. To detect whether

the $abacus\_gesture_t$ at timestamp t is stable, we always keep track of all the gestures detected in the last ΔT frames in the list denoted as $last\_\Delta t\_gestures$. If all the gestures in $last\_\Delta t\_gestures$ are the same as $abacus\_gesture_t$, we make it the stable gesture, assign it to the variable $stable\_gesture$, and update the $last\_\Delta t\_gestures$ accordingly.

4.5 Adapting to Varying Finger Dexterity and Camera Positions

Finally, we designed our algorithm to accommodate variations in users' finger dexterity and camera positions. To do this, we incorporated per-finger thresholds that can handle minor differences in finger dexterity and orientations.

Our algorithm includes a calibration stage, where users perform a series of predefined gestures like fully extending all fingers, closing all fingers, and creating numbers from 1 to 8. Based on these steps, the algorithm calculates threshold values for each finger, if necessary. By adjusting the reference keypoints according to these thresholds, our system tailors itself to the unique movements and capabilities of each user. This calibration phase can also adjust the stabilization threshold ΔT in Algorithm 1 to account for user expertise and interaction speed.

We conducted a series of lab experiments to gauge the impact of rotation along the Y-axis on gesture detection. Our tests involved rotating the hand along the Y-axis in 15-degree increments until the detection accuracy of the 21 keypoints fell by more than 30%. We found this drop in accuracy occurred at 75 degrees when the palm faced the camera (as illustrated in Figure 7a through Figure 7f) and at 50 degrees when the palm faced away from the camera (as illustrated in Figure 7g through Figure 7l). This result makes sense because as the hand rotates around the Y-axis, the fingers start to occlude each other. Particularly the thumb gets occluded by the other fingers and is hard to detect by the algorithm. Overall, we found an accuracy of approximately 95% when the palm was away from the camera.

5 USER STUDY: EVALUATION OF ABACUS GESTURES

We conducted a user study to evaluate the learnability, ease of use, and detection accuracy of Abacus Gestures. More specifically, we wanted to evaluate the following hypotheses–

- **H1**: Participants will be quick to learn the mapping of a number to the finger combinations.
- **H2**: Participants will make different numbers using Abacus Gestures easily.
- **H3**: A single camera-based system can easily detect Abacus Gestures with high accuracy.

5.1 Participants

We recruited 20 participants, referred to as P1 through P20, in our study through university mailing lists and personal outreach. We set our selection criteria to include individuals capable of forming diverse finger combinations and executing basic arithmetic tasks, such as counting, addition, and subtraction. The participants' ages ranged from 21 to 35 years (mean: 24.95, SD: 3.25), and the group comprised 15 males and 5 females. Notably, only one participant (P12) was left-handed, with the remainder being right-handed.

5.2 Apparatus

We constructed the Abacus Gestures detection algorithm utilizing the Python API of the MediaPipe-Hands module. We executed the study on a Dell XPS laptop equipped with Windows 10, utilizing its built-in webcam to capture the video feed necessary for gesture detection. Our system's interface is displayed in Figure 1b. This interface provides users with a prompt on the first line, followed by the Abacus Gestures they've input on the second line. The interface then displays the currently stable Abacus Gestures on the next line, as seen with the '31' in blue font color in Figure 1b. Users maintain the flexibility to adjust their finger combinations until they transition to gesture 0 by closing all fingers. Concurrently, the system updates the stable gesture in accordance

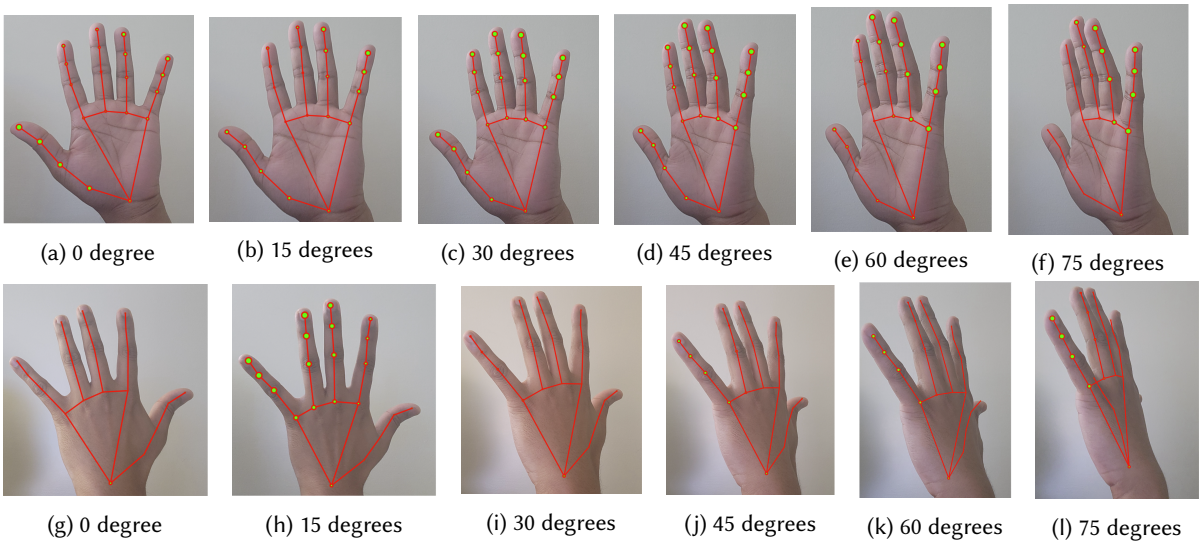


Fig. 7. The effect of rotation along the X-axis. (a)-(f) Shows the 21 keypoints detected by Mediapipe when the palm faces the camera and creates angles with the X-Y plane from 0 degrees to 75 degrees with approximately 15 degrees apart. For smaller angles, the fingers are clearly visible to the camera. As the angle increases, the fingers start to occlude each other. (g)-(l) Shows the 21 keypoints detected by Mediapipe when the palm is away from the camera and creates angles with the X-Y plane from 0 degrees to 75 degrees with approximately 15 degrees apart. Again, for larger angles, the occlusion is more prominent. However, compared to the earlier perspective, here, the thumb gets occluded more by the palm and the other fingers. As a result, at larger angles, the thumb is barely visible. In (l), the thumb is completely missed by the model, making it very difficult to detect the gestures.

with the users' finger combinations. Once users see their desired gesture stabilized, they can close their fingers to register the Abacus Gestures in the system.

5.3 Task and Design

We structured the study into three distinct phases. In the initial phase, we employed a think-aloud protocol [14]. Here, participants were asked to vocalize their thought process as they strategized how to represent a given number with Abacus Gestures. This phase aimed to ascertain the duration it took for participants to accurately devise the finger combination for a random number.

During the second phase, participants were required to produce various Abacus Gestures in response to prompts displayed on the screen. We employed the Wizard of Oz protocol [25] for this part of the study. A number was presented on the computer screen, and when participants made the corresponding Abacus Gesture, the interviewer (acting as the wizard) pressed a button to display the next number. The purpose of this phase was to examine how participants developed motor memory while generating various numbers using Abacus Gestures. This phase was divided into 5 task blocks, each containing 20 gestures. Therefore, each participant generated a total of 5 $task\ blocks \times 20\ gestures = 100\ gestures$ during this phase.

In the third phase, we employed a Repeated Measures protocol [76]. During this phase, participants were instructed to generate a sequence of Abacus Gestures representing numerical values using our system, which was running the gesture detection algorithm. The aim of this phase was to evaluate the accuracy and usability of a commodity camera-based system for mid-air interaction with Abacus Gestures. This phase also consisted of

multiple trials. In each trial, participants created a total of five sequences of Abacus Gestures, each sequence consisting of ten gestures. Thus, participants completed $5\ trials \times 5\ sequences \times 10\ gestures = 250\ gestures$ during the third phase. It's noteworthy that instead of prompting one gesture at a time, as we did in the second phase, we prompted a sequence of gestures. This approach allowed us to better understand how participants reacted to creating a sequence of gestures, a skill that might be required for specific tasks such as navigating hierarchical menus or entering text.

To assess the long-term learnability and memorability of Abacus Gestures, we distributed a brief survey to all 20 participants. This survey included questions about the value of different fingers, the maximum number achievable with Abacus Gestures, and the finger combinations necessary to represent a provided list of numbers. We instructed the participants to respond to these questions solely from memory, without resorting to any external resources for assistance.

5.4 Procedure

The study took place in a quiet, well-lit room. Initially, we obtained consent and demographic details from the participants. Next, we introduced the concept of the Finger Abacus, explained the assigned value for each finger, and demonstrated how to represent different numbers using various finger combinations in Abacus Gestures.

Once the participants confirmed their understanding of Abacus Gestures, we initiated the first phase of the study - the think-aloud protocol. In this phase, the interviewer provided a random number ranging from 0 to 99, and the participants were tasked with verbalizing the specific finger combination required to represent that number using Abacus Gestures. Once participants voiced their strategy, they were given a new number. This cycle continued until the interviewer was satisfied that the participants were able to swiftly and accurately detail their strategies. This phase typically lasted around five minutes.

Following the initial phase, we proceeded to the second phase, in which participants were seated before the computer and tasked with creating the Abacus Gestures corresponding to random numbers displayed on the screen. Acting as a wizard for the gesture detection system, the interviewer would quietly press a button to prompt the next number once the participant had correctly formed the requisite Abacus Gesture. In this phase, participants completed five task blocks of corresponding Abacus Gestures creation, with opportunities for rest after each block.

Subsequently, we introduced our system, which implements the Abacus Gesture detection algorithm, to the participants. They initially practiced forming several random Abacus Gestures using the system. Once they expressed comfort with the process, we commenced the study's third phase. Here, participants completed five trials of the Abacus Gestures formation task, with resting periods allowed after each trial. For each trial, the system sequentially prompted a set of 10 random numbers, and participants were tasked with forming the corresponding Abacus Gestures. Once a sequence of 10 was completed, the system would prompt the next sequence. Upon the completion of all five trials, we asked participants to complete the NASA Task Load Index (NASA TLX) questionnaire [33] to gauge their perceived task load while forming Abacus Gestures using our system.

Upon the completion of all three phases of the study, we invited participants to share their experiences with Abacus Gestures. We solicited feedback regarding the ease or difficulty they encountered in creating specific finger combinations. Additionally, we encouraged them to express their opinions about the accuracy of the Abacus Gestures detection system and potential areas for improvement.

Participants completed all three phases in a single session, which typically lasted approximately an hour. For their participation in the study, each participant received a 15 USD Amazon gift card.

Approximately four months after the in-person session, we dispatched an online survey to all participants to assess the long-term learnability and memorability of Abacus Gestures.

5.5 Data Collection and Analysis

We video-recorded all tasks for subsequent analysis and logged the timestamps for every gesture input during both the second and third phases of the study. Additionally, we gathered responses from the online survey completed by the participants.

Our analysis encompassed all video recordings, participant feedback, and survey responses, aiming to identify patterns related to the three hypotheses. Furthermore, we processed and analyzed the log files of gesture inputs to determine the time taken for gesture input, the accuracy of detection, and the ease of developing motor memory for Abacus Gestures.

6 FINDINGS

In this section, we report the findings from the three phases of the user study and their implications on our initial hypotheses.

6.1 Learnability of Abacus Gestures

First, we present our findings regarding how quickly participants could learn abacus gestures, which gestures were easier to develop motor memory, and how well they could retain their knowledge of the gestures in the long term.

6.1.1 Learning Time of Abacus Gestures. During the think-aloud phase of our user study, we observed that participants rapidly acquired the ability to form different finger combinations for Abacus Gestures. This learning process typically spanned a concise timeframe of five minutes and involved the creation of fifteen Abacus Gestures. Initially, for the first three to four numbers, participants sequentially extended their fingers, cognizant of each finger's value, and offered a response once their total finger count equaled the given number. However, this strategy evolved with the fifth number, wherein participants simultaneously extended their fingers, fine-tuning one or two fingers towards the end—for instance, initially forming the number six with their thumb and index finger, then incorporating the middle finger to represent seven. After creating eight to ten Abacus Gestures, we noticed that participants could effortlessly articulate and perform the gestures. To ensure sustained ease in their articulation process, we persisted with this exercise for a maximum of fifteen Abacus Gestures. The simplicity and swift mastery of Abacus Gestures were acknowledged by all participants.

All participants agreed that the Abacus Gestures are very easy to learn for them. P7 expressed his enthusiasm, stating he could "do it [Abacus Gestures] all day." P16 highlighted the simplicity of understanding the Abacus Gestures. P10, previously acquainted with the Abacus system, expressed substantial excitement about the introduction of finger-based Abacus Gestures. P17 offered the following comment:

> *"This is very easy! If we teach children in schools how to count with fingers, they can also use these gestures. Like, if I learned them [Abacus Gestures] in school, I would have been faster using them."*

Initially, participants encountered difficulty making Abacus Gestures for the numbers five and fifty using their Thumbs. Many participants associated the number five with all five fingers of a hand, given the visual correspondence. However, they quickly adapted to using the Thumb as the correct gesture for five and fifty. P1 shared the following observation:

> *"I am used to counting five with all five fingers. So, my hand automatically goes to this [five fingers as five]. So, I slightly get confused. I think if I keep practicing for some time, it won't be a problem."*

Overall, learning the finger combinations to make Abacus Gestures was very quick and easy for participants.

6.1.2 Motor Memory Development. We observed the speed at which participants developed motor memory for the Abacus Gestures, allowing them to perform gestures without counting the value of their fingers. We have

Table 2. A table categorizing the Abacus Gestures based on their ease of developing motor memory.

Category	Motor Memory Development	Abacus Gestures
Visual Match	Easy	0, 1, 2, 3, 4
Mnemonic	Easy	5, 9, 50, 55, 90, 99
Unimanual	Easy	6, 10, 20, 30, 40, 60
	Medium	7, 8, 70, 80
Bimanual Symmetric	Easy	11, 22, 33, 44
	Medium	66, 77
	Hard	88
Bimanul Asymmetric	Easy	12, 13, 14, 15, 16, 19, 21, 23, 24, 25, 26, 29, 31, 32, 34, 35, 36, 39, 41, 42, 43, 45, 46, 49 51, 52, 53, 54, 56, 59, 61, 62, 63, 64, 65, 69, 91, 92, 93, 94, 95, 96
	Medium	17, 27, 37, 47, 57, 67, 71, 72, 73, 74, 75, 76, 79, 97
	Hard	18, 28, 38, 48, 58, 68, 78, 81, 82, 83, 84, 85, 86, 87, 89, 98

categorized the gestures into five categories and three tiers under each category, each representing varying degrees of ease in motor memory development, as displayed in Table 2. Each category receives a detailed discussion in the subsequent sections.

Visually Matched Gestures. We noted that Abacus Gestures, which visually corresponded to finger combinations, were quickly and effortlessly learned and executed by participants. Five gestures fell into this category - 0, 1, 2, 3, 4. Moreover, participants developed motor memory for these gestures with notable speed after performing the gestures a couple of times.

Gesture Mnemonic. We identified another category of Abacus Gestures that did not visually correspond to finger combinations but were very easy for participants due to their uniqueness, which helped them develop mnemonics. This category includes six gestures - 5, 9, 50, 55, 90, and 99. All participants found making gestures 9 and 90, which required all open fingers on either of the hands, very easy, as they did not need to think about how many fingers to open. P13 commented that making 9 and 90 was the easiest for him since he could just open all his fingers without worrying about raising the wrong number of fingers. Five participants (P1, P5, P6, P13, and P18) compared 9 and 90 to the 'stop' gesture, as 9 represents the maximum number or the stopping point for the dominant hand and 90 for the non-dominant hand. Similarly, 99 was very easy to learn for participants as it was the maximum gesture, and they did not need to count their fingers' worth to create it.

We also found that using the Thumb to make 5, 50, and 55 was unique, memorable, and easy to learn for participants. Initially, participants opened all five fingers to represent 5 and 50 due to the visual match between the number and the fingers. Three participants (P1, P5, and P8) initially thought that learning to represent 5 with their thumb would be difficult, as they were accustomed to associating 5, 50, and 55 with having all five fingers open. However, they were surprised at how quickly they developed motor memory to make 5, 50, and 55 with their thumbs. Eight participants (P4, P6, P7, P10, P11, P16, P17, and P19) commented that they could associate 5, 50, and 55 with the 'thumbs up' gesture, which helped them perform these gestures quickly. These two unique finger combinations, open Thumb and all open fingers were also reported to be very comfortable to perform by participants.

Unimanual Gestures. Unimanual Abacus Gestures, which required only a single hand, were also easy to learn and make for participants. On the dominant hand, Abacus Gesture 6 fell in this category. On the non-dominant hand, the gestures 10, 20, 30, 40, and 60 were also easy to learn and develop motor memory for participants.

Although gestures 7, 8, 70, and 80 were also unimanual, we found that these gestures required more time for participants to master compared to the other unimanual gesture. After further investigation, we observed that Abacus Gestures that required Thumb + additional fingers (except for 9) took more time for participants to develop motor memory. Initially, participants made these gestures by opening the Thumb and then adding one finger at a time. They were able to quickly develop motor memory for gestures 6 and 60. However, making 7, 70, and 8, 80 took more time than 6, 60, which is why we consider 6 and 60 in the easy tier and the rest in the medium tier. In addition, in the initial few attempts to make 7, 70, and 8, 80, we observed participants actively looking at the fingers and counting the worth to make sure they were making the correct finger combinations. However, after the initial attempts, participants could develop motor memory and perform these gestures without counting the fingers.

Overall, we found that most unimanual gestures were easier to learn and develop motor memory than bimanual gestures for participants. P5 provided insight into these why these numbers were easier:

> *"When I am making numbers with a single hand, I only need to focus on just one hand, and the gestures come naturally. However, when both of my hands are involved, this attention gets divided, and I need to make sure I am combining the correct fingers from both hands."*

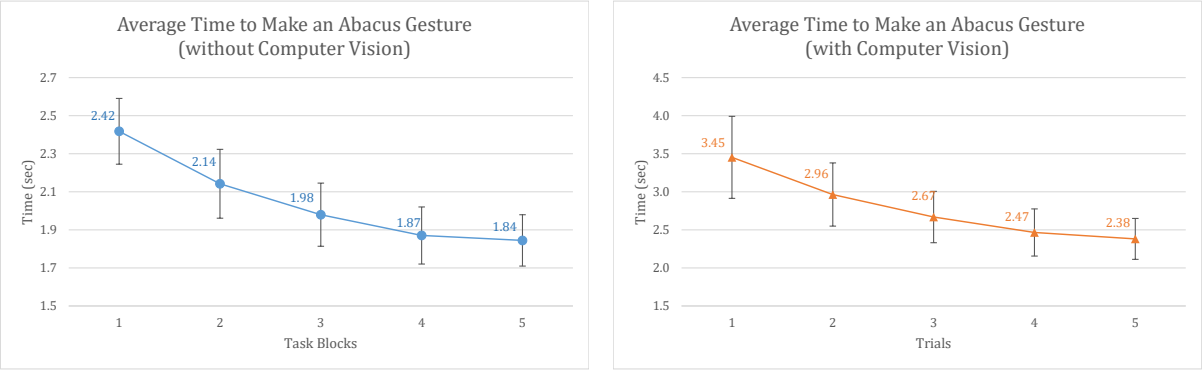
Bimanual Symmetric Gestures. For bimanual Abacus Gestures, symmetric gestures or the gestures that required the same finger combination from both hands were mostly easy to learn for participants. Particularly, learning the four symmetric Abacus Gestures - 11, 22, 33, and 44 was very easy and fast in this category. None of these Abacus Gestures involve the Thumb, and the number at the unit place is a direct visual match to the fingers of the dominant hand. In addition, the non-dominant hand has the exact same finger combinations as the dominant hand, making these gestures faster for motor memory development.

Two other symmetric Abacus Gestures, 66 and 77, were medium tier in this category as participants needed to combine both their hands; each hand involved the Thumb + additional fingers. Finally, we found the symmetric gesture 88 to be hard to learn as participants required opening Thumb + three additional fingers in both hands, which required them to actively count their fingers on both hands.

Bimanual Asymmetric Gestures. The rest of the Abacus Gestures fall under this category. We observed that gestures that required 7 and 70 had medium learnability, whereas gestures involving 8 and 80 required more time to learn and develop motor memory. This was unsurprising as we found unimanual 7, 70, and 8, 80 also took more time for participants to learn compared to the other unimanual gestures. Therefore, involving another hand with these Abacus Gestures took even more time for participants to make and learn. We particularly found that asymmetric gestures involving 8 and 80 were the hardest for participants to learn. Interestingly, 6 participants (P2, P3, P5, P13, P5, and P18) mentioned that to distinguish between 7 and 8, they had to consciously think about which fingers they were raising. Consequently, 78 and 87 were the two numbers that took the most time and effort for these participants to learn, as they had to focus on both hands and actively count the fingers' worth almost every time it appeared.

The rest of the bimanual asymmetric gestures were found easy to learn for participants as they could develop motor memories for them after some time. All of these gestures involve 1 - 6, 9 on the dominant hand and 10 - 60, 90 on the non-dominant hand. Again, these gestures were also in the easy tier in their corresponding unimanual categories.

6.1.3 Long-Term Memorability of Abacus Gestures. 10 out of 20 participants took part in the survey. The approximate interval between the in-person study and the online survey was 120 days on average. From the survey data,



(a) (b)

Fig. 8. (a) Average time to make an Abacus Gesture over the five task blocks in Phase 2 (without the Computer Vision module). (b) Average time to make an Abacus Gesture over the five trials in Phase 3 (with the Computer Vision module). Note that the required time per gesture plateaued near the end in both phases, indicating participants' performance was almost saturated during the study.

we found that all 10 participants answered all the questions correctly. In addition, we discovered that participants also retained their preferred finger combinations (e.g., using the Middle + Ring + Little combination on the dominant hand to make 3) from their earlier study to make the gestures. These findings indicate that Abacus Gestures are memorable over a long period of time.

The above findings support H1 that learning the finger combinations for Abacus Gestures was very quick for participants.

6.2 Ease of Execution of Abacus Gestures

Next, we report the performance of the participants in making Abacus Gestures and the comfort of making these gestures using the finger combinations.

6.2.1 Average Gesture Making Time Without Computer Vision Module. Figure 8a shows the average time to make an Abacus Gesture over the five task blocks calculated from phase 2 of the user study. In this phase, there was no underlying computer vision (CV) module for gesture detection. When the CV module was absent, participants required 2.42 seconds (SD: 0.17) per Abacus Gesture on the first task block. In the fifth task block, this per gesture-making time was reduced to 1.84 seconds (SD: 0.13), which was 31.52% less than the first block. A Repeated Measure ANOVA found a significant main effect of the task blocks on the required time per gesture ($F(4, 16) = 173.632, p < .001$) without the CV module, which indicated that participants' performance of Abacus Gestures improved significantly over the five task blocks.

Note that the average gesture-making time plateaued near the fifth block, indicating that participants could learn to make the Abacus Gestures within a short amount of practice and reach very close to their maximum performance.

6.2.2 Average Gesture Making Time With Computer Vision Module. Figure 8b indicates the average time to make an Abacus Gesture over the five trials calculated from phase 3, where our underlying computer vision (CV) module detected the gestures. With the CV module, participants required 3.45 seconds (SD: 0.54) per Abacus Gesture on the first trial. In the fifth trial, this per gesture-making time was reduced to 2.38 seconds (SD: 0.27),

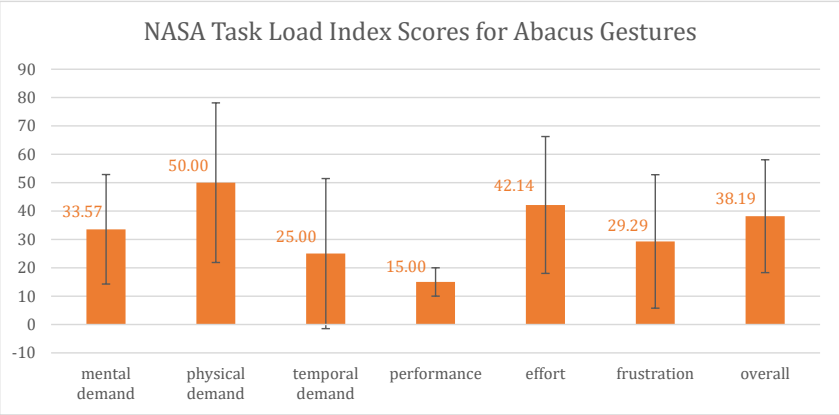


Fig. 9. The average NASA TLX scores for making Abacus Gestures.

which was 31.01% less than the first trial. A Repeated Measure ANOVA found a significant main effect of the trials on the required time per gesture ($F(4, 16) = 44.206, p < .001$) with the CV module, which indicated that participants' performance of Abacus Gestures improved significantly over the five trials.

Similar to phase 2, the average gesture-making time plateaued near the fifth block, indicating that participants could learn to make the Abacus Gestures within a short amount of practice. We observed that in both the second and the third phase, the average Abacus Gestures had similar trends over time. However, with the CV module, the overall gesture-making time was higher compared to the other phase. We found that, without the CV module, participants made the gestures more casually (e.g., not closing the fingers accurately to make a fist). On the other hand, with the CV module, they needed to make these gestures accurately in front of the camera. Otherwise, the Abacus Gestures were not detected accurately by the system, which took more time. In addition to making the gestures accurately, participants also looked at the feedback from the system with the CV module and verified that they were making the correct Abacus Gestures. Such feedback was not present in the second phase.

6.2.3 Perceived Task Load. Figure 9 represents the NASA TLX scores to make Abacus Gestures for all participants. The result indicates that participants required more physical demand than mental demand (50 vs. 33.57) for making Abacus Gestures, which is unsurprising because making mid-air gestures is physically demanding. In addition, less mental demand indicates that participants were able to learn the technique and use the gestures without much cognitive effort. The overall task load for the making Abacus Gestures was 38.18 on average after making 250 gestures (SD: 19.89), indicating the gestures were usable with a moderate task load for mid-air interaction.

6.2.4 Effect of Finger Dexterity on Gesture Execution. From the study data and participants' feedback, we observed that finger dexterity played a major role in making different Abacus Gestures. Consequently, some Abacus Gestures were easier to make with fingers than others. Table 3 provides categorizes the Abacus Gestures based on how easy or difficult the Abacus Gestures were to execute. According to the participants, the five most comfortable finger combinations were Index (1, 10), Index + Middle (2, 20), Thumb (5, 50), Thumb + Index (6, 60), and all five fingers (9, 90). All participants, except for P2, also felt comfortable making 4/40 with the combination of Index + Middle + Ring + Little fingers. Only P2 reported that his thumb naturally opened slightly when opening the other four fingers - Index + Middle + Ring + Little. Therefore, when he tried to make Abacus Gestures 4, 40, it

Table 3. A table categorizing the Abacus Gestures depending on their ease of execution.

Execution	Abacus Gestures	Most Frequent Combinations (N)	Alternative Combinations (N)
Easy	0	All Fingers Closed (20)	N/A
	1, 10	Index (19)	Little (1)
	2, 20	Index + Middle (19)	Ring + Little (1)
	4, 40	Index + Middle + Ring + Little (20)	N/A
	5, 50	Thumb (20)	N/A
	6, 60	Thumb + Index (19)	Thumb + Little (1)
	9, 90	All Fingers Open (20)	N/A
Medium	3, 30	Index + Middle + Ring (16)	Middle + Ring + Little (4)
	7, 70	Thumb + Index + Middle (18)	Thumb + Index + Little (2)
Hard	8, 80	Thumb + Middle + Ring + Little (10)	Thumb + Index + Middle + Ring (9), Thumb + Index + Middle + Little (1)

was occasionally detected as 9, 90 by our system at first. However, P2 commented that it was not an issue for him after some time, as he got accustomed to making the combination without opening the Thumb.

5 out of 20 participants (P3, P5, P12, P13, and P5) reported that making 7, 70 with Thumb + Index + Middle was easy initially, but they felt slightly uncomfortable after making this combination for a long time. Among them, 2 participants modified the combination and made the gestures 7, 70 using the combination Thumb + Index + Little, which was similar to a 'rock' or 'horn' gesture and easy to perform. The flexibility to make a gesture with multiple finger combinations allowed participants to adapt and find a comfortable finger combination; all participants appreciated this. A similar adaptation we observed for making the gestures 3, 30 by 4 participants (P2, P9, P12, and P18). They relied on the Middle + Ring + Little finger combination instead of the Index + Middle + Ring combination used by the other 16 participants.

Participants had the most difficulty making Abacus Gestures 8, 80 with the combination of Thumb + Index + Middle + Ring. While making 8 using this combination, either the Ring finger was too low (detected as 7, 70) or the Little finger was open with the Ring finger (detected as 9, 90). 11 out of 20 participants found this sequential finger combination difficult to make. As a workaround, participants chose different finger combinations for gestures 8, 80. The most preferred finger combination was Thumb + Middle + Ring + Little, used by 11 participants, instead of the sequential combination (used by 9 participants). Only a single participants, P14, used the Thumb + Index + Middle + Little combination to make the gestures 8, 80.

To better understand our results, we resorted to two metrics related to the anatomy of human hands - individuation index and finger enslaving. Although the human hand possesses high dexterity, not all fingers can move independently. The degree to which a finger can move individually is the individuation index of the finger. The Thumb and the Index finger have high individuation indices, whereas the Ring finger has the lowest individuation index [41, 72]. The second metric is finger enslaving, which refers to the unintentional movement of a finger due to the movement of another finger. Again, we found that both the Thumb and the Index finger have minor enslaving effects, whereas the Middle and the Little finger highly enslave the Ring finger, limiting its independent movement. Due to the high individuation and low enslaving effect of the Thumb and the Index finger, Abacus Gestures involving only these two fingers were very easy to make for participants (e.g., gestures 1, 10, 5, 50, and 6, 60). On the contrary, making the gestures 8, 80 with the Thumb + Index + Middle + Ring combination was

difficult as the closed Little finger limits the opening of the Ring Finger [41, 72]. For the same reason, the Thumb + Middle + Ring + Little combination was easier for the gestures 8, 80, which was preferred by most participants.

The above findings support H2.

6.3 Usability of Abacus Gestures Detection System

Finally, we report the accuracy and usability of our single commodity camera-based gesture detection algorithm.

6.3.1 Gesture Detection Accuracy. We found that the hand tracking and detection accuracy of Abacus Gestures with our implemented system was over 98% using just a 2D commodity camera, and the users' hands were completely free without any marker or accessories. The reason for this higher accuracy is that the Abacus Gestures require only 2D coordinates from the keypoints. In addition, the users faced their palms toward the camera during the study, which made hand detection and tracking very robust and improved gesture detection accuracy. Compared to our setup, MediaPipe reported their accuracy on real-world hand images with different lighting conditions and occlusions due to the orientation of the hands in different directions. In our lab study, we found over 95% accuracy of detection when the palm was facing the body (discussed in Section 4.5).

Investigating further into the scenarios where our system could not detect the correct gestures, we identified three issues. The first issue was environmental, where the background interfered with the participants' hands. 3 out of 20 participants (P6, P12, and P19) mentioned that they had to adjust their hand occasionally for the system to recognize the gestures. For example, P12 initially faced problems with the system as her dress had bright and colorful patterns causing the system to inaccurately detect some gestures. Later, she adjusted her hand positions so that the hands did not overlap with the dress, which significantly improved the detection accuracy.

Secondly, the system detected some gestures incorrectly when users' fingers were not properly opened or closed. For example, in the neutral position, some participants' were not making a fist, and their Thumbs were slightly opened, causing the system to incorrectly detect 5 and 50. In addition, during the beginning of the sequence in Phase 3 of the study, some participants moved their hands (e.g., randomly touching their face), and the system detected a random gesture. Moreover, while making 8, 80, the gestures were sometimes detected as 7, 70, or 9, 90 due to partially open fingers, as discussed in Section 6.2.4. As we progressed with the study, we found that participants adapted their behavior, and the incorrect detection declined.

Finally, we found that when participants became faster in making these Abacus Gestures, sometimes the system failed to recognize the gesture. The reason for this was the stabilization threshold ΔT, introduced in Section 4.4. Particularly, participants were faster in opening and closing their Thumb for 5, 50 and their Index finger for 1, 10. Due to fast opening and closing, the gestures were not stable and were not recognized by the system. As a result, participants required a second slower attempt to input these gestures in some cases.

6.3.2 Feedback Regarding User Experience. All participants found Abacus Gestures to be interesting and fun to use for mid-air interaction. They appreciated the design and the accuracy of the system for the detection of finger counting and utilizing the counts as input gestures. Participants encouraged the system feedback that showed the current gesture and got updated when the finger combination changed. P15 found the feedback very helpful for making Abacus Gestures without errors. P13 commented that:

> *"How the system supports counting is very helpful. It does not take a number until I close all my fingers. So I can literally count with my fingers when I need to make a bigger number. Like when I am making 68, my mind usually goes to the closest number, like 70. So I can count 70 with my left hand. Then I can take down 1 finger from the left hand to go to 60 and count 8 with my right hand. This is very helpful for me as it supports how I am thinking in my mind."*

P5 also found that as the system does not enter a number until closing all fingers, it took the pressure off of her to make the exact count with both hands and allowed time and flexibility to adjust the finger combinations for making large numbers. She commented:

> *"Although I need to focus on both hands when making numbers like 87, I can take my time and divide the attention. For example, I can focus on my left hand first to make an 80, and the system also shows 80. Then it waits until I focus on my right hand to make a 7. Again, I can see the updated count 87 from the system, which assures my finger combination is correct. Finally, I can close my fingers to enter the number. So, the system makes it easier for me to enter the gestures correctly."*

Participants also expressed that the system was very accurate in detecting gestures from finger combinations. For example, P15 commented that he had difficulty making 8 with his right hand. He could barely open the Ring finger without raising the Little finger. However, he was surprised to see that Abacus Gesture could detect the gestures correctly almost every time. Overall, the counting process, detection accuracy, and gesture delimiting technique were well-received by all participants for mid-air gesture-based interaction.

The findings above support our third hypothesis.

7 DISCUSSION AND FUTURE WORK

In this section, we discuss the implications of our findings, provide guidelines regarding the use of Abacus Gestures, and explore potential use cases.

7.1 Design Implications

Given that our proposed gesture set encompasses a total of 100 gestures, one could conceivably map these gestures to corresponding interface operations or elements, such as menu items, in an arbitrary manner. However, this arbitrary allocation of gestures could inadvertently amplify the cognitive burden on the users. To counteract this potential issue, we advocate for assigning these gestures according to a systematic arrangement that users can readily mentally map or categorize. By adhering to this methodology, Hick's Law in interface design [35] suggests that the cognitive load on users will only increase logarithmically, even as the number of choices proliferates, provided the underlying arrangement possesses some orders.

We additionally recommend the following strategies to map our gestures to corresponding interface operations or elements.

7.1.1 Assigning Gestures by their Learnability and Ease of Execution. When assigning Abacus Gestures for a particular application, gestures that are easy to learn and comfortable to execute should be assigned first. In Table 2 and 3, we categorized all Abacus Gestures based on their learnability and comfort of execution. For example, the five Abacus Gestures in the Visual Match category in Table 2 were very easy for users to learn from the beginning. As a result, these Abacus Gestures should be assigned to the most frequent interactions. Using the same principle, Abacus Gestures that take more time for participants to learn and perform can be assigned to infrequent interactions.

7.1.2 Mapping Gestures to Mataphors. When designing for a specific application, gestures can be assigned based on metaphors so that users can easily interact with the system. For example, for a media player system, stopping a media can be assigned to gesture 9 (all fingers open on the dominant hand), which represents a 'stop' sign. Similarly, gesture 5 or 'thumbs up' can be used to add media as a favorite item. A similar metaphor-based gesture assignment has been found useful in prior work as well [34].

7.1.3 Assigning Gesture by Mnemonic. Gestures assigned by some form of mnemonic can also be easy to learn for users. For example, users always remember eventful numbers such as the date of birth of them or their children.

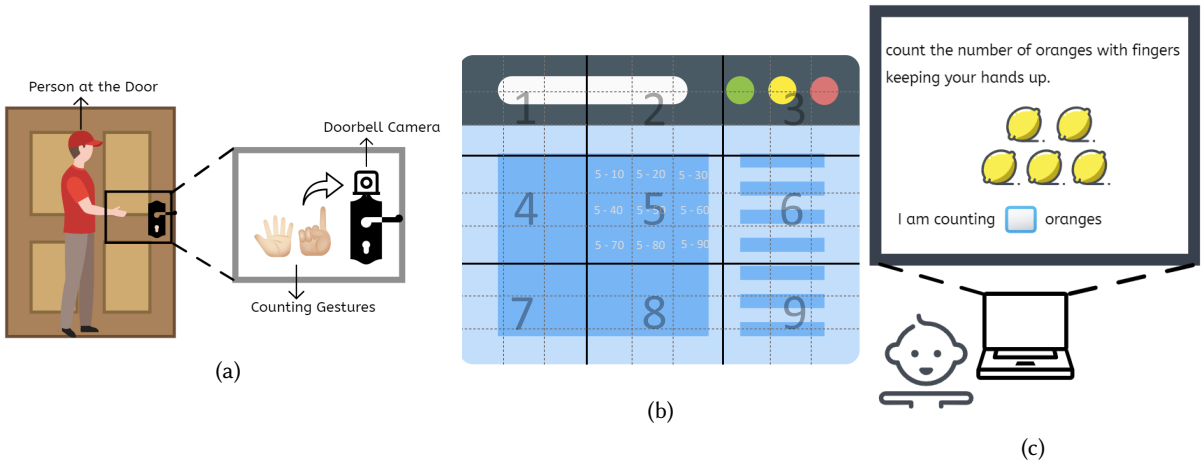


Fig. 10. Three potential use-cases for Abacus Gestures. (a) A delivery person at the door is typing a short message using the doorbell camera. (b) An extended mouse grid. A computer window is divided into nine grids; each contains additional nine sub-grids inside. The mouse cursor can be moved to a smaller sub-grid with Abacus Gestures. (c) An interactive counting game for children to teach arithmetic concepts in a fun and engaging way. Children can count using Finger Abacus and input the gestures to play the game.

Therefore, frequent interactions can be assigned to such numbers to facilitate users' memorability and learnability. In this case, an interface that lets users customize Abacus Gestures can be a good choice.

7.1.4 Designing Math-based Shortcut. After assigning gestures for a particular application, remaining unassigned Abacus Gestures can be used to design math-based shortcuts combining multiple operations under a single Abacus Gesture. One strategy is to assign the sum of the subtasks of a multi-step task as the shortcut. For example, if 20 files are assigned to Abacus Gestures 1 to 20, and the deletion operation is assigned to gesture 50, then the sum of 50 and the file number can be directly used to delete that file (Abacus Gestures 50 + 9 = 59 to delete the file 9). A similar strategy is also used in Grade 2 Braille [82], where the unassigned braille patterns represent frequently occurring prefixes/suffixes (e.g., -ing). However, we acknowledge that such addition-based shortcuts can collide with other Abacus Gestures and may require some form of disambiguation.

7.2 Potential Use Cases of Finger Abacus Gestures

Here, we envision four potential use cases for interaction with Abacus Gestures in mid-air.

7.2.1 Supporting Multimodal Interaction with Abacus Gestures. Abacus Gestures can complement other modalities, such as speech, to improve the quality of interaction. One such example can be text correction using the combination of speech and abacus gestures. Correcting errors in touch- and speech-based interfaces can be cumbersome and often require users to manually move the cursor to a specific location and then correct the errors. To make the process quicker and easier, prior work [92] combined gaze and speech, where users first look at the incorrect word to select it and then utter the new word to correct it. A similar goal can be accomplished using a combination of speech and Abacus Gestures, considering each word in a sentence tagged using numbers from 1 to n. For example, if the third word of the sentence is incorrect, users can raise the Abacus Gesture 3 to select the word and utter the new word to correct it. This strategy can be particularly helpful for non-visual users

who cannot use their eye gaze. They can use Abacus Gestures to query words, identify the incorrect word, and then utter the correct word.

7.2.2 Mid-Air Typing with a Commodity Camera. Abacus Gestures can be mapped to a keyboard layout to enter short texts and digits in IoT devices with a camera for touchless interaction. For example, users can leave short messages with doorbell cameras or enter one-time passcodes (OTPs) in kiosks with Abacus Gestures. Figure 10a shows a delivery person at the door leaving a short message and the contact number for the unavailable recipient using Abacus Gestures with a doorbell camera. The camera feed can be processed by a microcontroller (e.g., Raspberry Pi) to convert the gestures to letters and save the message for the recipient. To facilitate typing, the layout can be printed and placed near the camera. Later, the recipient can use the contact number to reach out to the delivery person to receive a package.

7.2.3 Touch-less and Keyboard-less Computer Interaction. Abacus Gestures can be used to interact with computers when users cannot access a physical mouse or a keyboard. One example of touchless interaction is MouseGrid, which divides the computer interface into 9 cells (1 - 9) using a 3×3 grid. Users can move their mouse pointer to a specific cell by speaking the cell number. We can extend the MouseGrid by dividing each of the 9 cells into 9 sub-cells using another 3×3 inner grid, effectively dividing the screen into 81 cells. To move the mouse to a specific grid cell, users can select the first-level grid with their dominant-hand count and then select a sub-cell using their non-dominant-hand count. Unlike prior work ([11, 18]) that leveraged traditional 10-finger counting for limited menu navigation, Abacus Gestures can also support navigation containing a total of 100 menu items.

7.2.4 Interactive Games for Children to Learn Arithmetic Operations. Prior work in the neuro-cognitive development of children reported that children who possess good finger-based numerical representation skills could acquire better arithmetic skills [55]. Therefore, learning finger math early is beneficial for later numerical development [8, 24]. To support learning finger-based arithmetic operations, we can utilize Finger Abacus gestures to design interactive games for children. to familiarize them with number sense and basic arithmetic operations. Figure 10c illustrates such a use-case, where a child is playing a hypothetical game on a computer. The game prompts the child to count the number of oranges shown on the screen. As the child makes the count, the interface will show the count on the screen. When the child reaches the appropriate count, the interface will provide feedback to the child about the correct count and move to the next task. With progress, other arithmetic tasks (e.g., addition and subtraction) can be included for the child.

7.3 Limitation and Future Work

In our study, we focused on evaluating the usability, ease of performance, and detection accuracy of our proposed large set of Abacus Gestures. Our goal was to use a familiar metaphor, finger-counting in this case, to propose a large number of gestures. In addition, we discussed the implication of the findings, proposed gesture assignment policy, and discussed potential use cases. However, a limitation of this work is that we did not evaluate the performance of our proposed gesture set for any specific task and evaluate the performance of the gesture set to accomplish the task. In our future work, we aim to evaluate the assignment of Abacus Gestures for specific tasks and determine the memorability, usability, and performance of the gesture set. In addition, we aim to evaluate whether our gesture assignment policy in a non-math context is useful or not and compare our gesture set with available representative gesture sets.

We also aimed to uniformly cover all gestures from 0 to 99 in our prompts during the study. However, not all participants performed each gesture with equal frequency, and the gestures were not associated with any task that necessitated the assignment of all 100 gestures. As a result, the usability of our gestures might differ depending on the nature of real-world tasks. Moving forward, we plan to employ our gestures in scenarios where

the complete set of 100 gestures is required, ensuring equal user performance for a thorough assessment of learnability, execution, and usability.

8 CONCLUSION

In this work, we propose Abacus Gestures, a large set of usable finger-counting gestures for mid-air interaction. Abacus Gestures are designed using Finger Abacus counting as a metaphor to support a total of 100 gestures. We detected these gestures with a 2D commodity camera and an off-the-shelf computer vision algorithm. We conducted a user study with 20 participants to understand the learnability, memorability, and ease of use of our proposed gestures. We found that Abacus gestures were very easy to learn for participants just after five minutes of practice. In addition, these gestures were memorable over a long period of time and were comfortable for users to perform. The gesture could also be detected using just a single commodity camera with over 98% and 95% accuracy when the palm is facing the camera and the body, respectively. Based on the findings and observations, We categorized the gestures in terms of their motor memory development time and ease of performance. We also provided design guidelines regarding the use of this large math-based gesture set in various mid-air applications.

ACKNOWLEDGMENTS

We thank anonymous reviewers for their insightful feedback. This work was supported in part by the Center for Biodevices at Penn State seed grant # 460000000530.

REFERENCES

[1] 2017. An incredible way to do maths: finger calculations the Indian way. https://www.news24.com/parent/learn/back-to-school/maths-finger-calculations-the-indian-way-20170227
[2] 2019. Learning Mental Maths with The Abacus Finger Theory. https://supermaths.co.uk/making-maths-fun-with-the-abacus-finger-theory/
[3] 2022. Finger Abacus: Learn the Technique to Calculate Faster. https://enthu.com/blog/abacus/finger-abacus/
[4] 2022. How to use Palm Gesture to take selfie on Samsung mobile?. https://www.samsung.com/sg/support/mobile-devices/how-to-use-palm-gesture-to-take-selfie-on-samsung-mobile-device/
[5] 2022. Kinect for Windows. https://learn.microsoft.com/en-us/windows/apps/design/devices/kinect-for-windows
[6] 2022. Leap Motion Controller. https://www.ultraleap.com/product/leap-motion-controller/
[7] Roland Aigner, Daniel Wigdor, Hrvoje Benko, Michael Haller, David Lindbauer, Alexandra Ion, Shengdong Zhao, and JTKV Koh. 2012. Understanding mid-air hand gestures: A study of human preferences in usage of gesture types for hci. *Microsoft Research TechReport MSR-TR-2012-111* 2 (2012), 30.
[8] Michael Andres, Samuel Di Luca, and Mauro Pesenti. 2008. Finger counting: The missing tool? *Behavioral and Brain Sciences* 31, 6 (2008), 642–643.
[9] Shaikh Shawon Arefin Shimon, Courtney Lutton, Zichun Xu, Sarah Morrison-Smith, Christina Boucher, and Jaime Ruiz. 2016. Exploring non-touchscreen gestures for smartwatches. In *Proceedings of the 2016 chi conference on human factors in computing systems*. 3822–3833.
[10] Gilles Bailly, Eric Lecolinet, and Yves Guiard. 2010. Finger-count & radial-stroke shortcuts: 2 techniques for augmenting linear menus on multi-touch surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 591–594.
[11] Gilles Bailly, Jörg Müller, and Eric Lecolinet. 2012. Design and evaluation of finger-count interaction: Combining multitouch gestures and menus. *International Journal of Human-Computer Studies* 70, 10 (2012), 673–689.
[12] Gilles Bailly, Jörg Müller, Michael Rohs, Daniel Wigdor, and Sven Kratz. 2012. Shoesense: a new perspective on gestural interaction and wearable applications. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 1239–1248.
[13] Gilles Bailly, Robert Walter, Jörg Müller, Tongyan Ning, and Eric Lecolinet. 2011. Comparing free hand menu techniques for distant displays using linear, marking and finger-count menus. In *IFIP Conference on Human-Computer Interaction*. Springer, 248–262.
[14] Ted Boren and Judith Ramey. 2000. Thinking aloud: Reconciling theory and practice. *IEEE transactions on professional communication* 43, 3 (2000), 261–278.
[15] Idil Bostan, Oğuz Turan Buruk, Mert Canat, Mustafa Ozan Tezcan, Celalettin Yurdakul, Tilbe Göksun, and Oğuzhan Özcan. 2017. Hands as a controller: User preferences for hand specific on-skin gestures. In *Proceedings of the 2017 Conference on Designing Interactive Systems*. 1123–1134.
[16] Carl B Boyer and Uta C Merzbach. 2011. *A history of mathematics*. John Wiley & Sons.

[17] Jessica R Cauchard, Jane L E, Kevin Y Zhai, and James A Landay. 2015. Drone & me: an exploration into natural human-drone interaction. In *Proceedings of the 2015 ACM international joint conference on pervasive and ubiquitous computing*. 361–365.

[18] Tusher Chakraborty, Md Taksir Hasan Majumder, Md Nasim, and ABM Alim Al Islam. 2018. Duity: A Low-cost and Pervasive Finger-count Based Hand Gesture Recognition System for Low-literate and Novice Users. In *Proceedings of the 15th EAI International Conference on Mobile and Ubiquitous Systems: Computing, Networking and Services*. 352–361.

[19] Liwei Chan, Rong-Hao Liang, Ming-Chang Tsai, Kai-Yin Cheng, Chao-Huai Su, Mike Y Chen, Wen-Huang Cheng, and Bing-Yu Chen. 2013. FingerPad: private and subtle interaction using fingertips. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. 255–260.

[20] CL Chen, TH Wu, MC Cheng, YH Huang, CY Sheu, JC Hsieh, and JS Lee. 2006. Prospective demonstration of brain plasticity after intensive abacus-based mental calculation training: An fMRI study. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 569, 2 (2006), 567–571.

[21] Ke-Yu Chen, Kent Lyons, Sean White, and Shwetak Patel. 2013. uTrack: 3D input using two magnetic sensors. In *Proceedings of the 26th annual ACM symposium on User interface software and technology*. 237–244.

[22] Zhen Chen, Xiaochi Ma, Zeya Peng, Ying Zhou, Mengge Yao, Zheng Ma, Ci Wang, Zaifeng Gao, and Mowei Shen. 2018. User-defined gestures for gestural interaction: extending from hands to other body parts. *International Journal of Human–Computer Interaction* 34, 3 (2018), 238–250.

[23] Philip S Cho and Wing Chee So. 2018. A feel for numbers: The changing role of gesture in manipulating the mental representation of an abacus among children at different skill levels. *Frontiers in psychology* 9 (2018), 1267.

[24] Virginie Crollen, Xavier Seron, and Marie-Pascale Noël. 2011. Is finger-counting necessary for the development of arithmetic abilities? *Frontiers in Psychology* 2 (2011), 242.

[25] Nils Dahlbäck, Arne Jönsson, and Lars Ahrenberg. 1993. Wizard of Oz studies: why and how. In *Proceedings of the 1st international conference on Intelligent user interfaces*. 193–200.

[26] Tobias Dantzig and Joseph Mazur. 2007. *Number: The language of science*. Penguin.

[27] Stanislas Dehaene. 2011. *The number sense: How the mind creates mathematics*. OUP USA.

[28] Niloofar Dezfuli, Mohammadreza Khalilbeigi, Jochen Huber, Florian Müller, and Max Mühlhäuser. 2012. PalmRC: imaginary palm-based remote control for eyes-free television interaction. In *Proceedings of the 10th European conference on Interactive tv and video*. 27–34.

[29] Haiwei Dong, Ali Danesh, Nadia Figueroa, and Abdulmotaleb El Saddik. 2015. An elicitation study on gesture preferences and memorability toward a practical hand-gesture vocabulary for smart televisions. *IEEE access* 3 (2015), 543–555.

[30] Tanja Döring, Dagmar Kern, Paul Marshall, Max Pfeiffer, Johannes Schöning, Volker Gruhn, and Albrecht Schmidt. 2011. Gestural interaction on the steering wheel: reducing the visual demand. In *Proceedings of the sigchi conference on human factors in computing systems*. 483–492.

[31] Rochel Gelman and Charles R Gallistel. 1986. *The child's understanding of number*. Harvard University Press.

[32] Chris Harrison, Hrvoje Benko, and Andrew D Wilson. 2011. OmniTouch: wearable multitouch interaction everywhere. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*. 441–450.

[33] Sandra G Hart and Lowell E Staveland. 1988. Development of NASA-TLX (Task Load Index): Results of empirical and theoretical research. In *Advances in psychology*. Vol. 52. Elsevier, 139–183.

[34] Niels Henze, Andreas Löcken, Susanne Boll, Tobias Hesselmann, and Martin Pielot. 2010. Free-hand gestures for music playback: deriving gestures with a user-centred process. In *Proceedings of the 9th international conference on Mobile and Ubiquitous Multimedia*. 1–10.

[35] William E Hick. 1952. On the rate of gain of information. *Quarterly Journal of experimental psychology* 4, 1 (1952), 11–26.

[36] Yuzheng Hu, Fengji Geng, Lixia Tao, Nantu Hu, Fenglei Du, Kuang Fu, and Feiyan Chen. 2011. Enhanced white matter tracts integrity in children with abacus training. *Human brain mapping* 32, 1 (2011), 10–21.

[37] Hessam Jahani, Hasan J Alyamani, Manolya Kavakli, Arindam Dey, and Mark Billinghurst. 2017. User evaluation of hand gestures for designing an intelligent in-vehicle interface. In *Designing the Digital Transformation: 12th International Conference, DESRIST 2017, Karlsruhe, Germany, May 30–June 1, 2017, Proceedings 12*. Springer, 104–121.

[38] David Kim, Otmar Hilliges, Shahram Izadi, Alex D Butler, Jiawen Chen, Iason Oikonomidis, and Patrick Olivier. 2012. Digits: freehand 3D interactions anywhere using a wrist-worn gloveless sensor. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*. 167–176.

[39] Panayiotis Koutsabasis and Panagiotis Vogiatzidakis. 2019. Empirical research in mid-air interaction: A systematic review. *International Journal of Human–Computer Interaction* 35, 18 (2019), 1747–1768.

[40] Arun Kulshreshth and Joseph J LaViola Jr. 2014. Exploring the usefulness of finger-based 3D gesture menu selection. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1093–1102.

[41] Catherine E Lang and Marc H Schieber. 2004. Human finger independence: limitations due to passive mechanical coupling versus active neuromuscular control. *Journal of neurophysiology* 92, 5 (2004), 2802–2810.

[42] Daeho Lee and Youngtae Park. 2009. Vision-based remote control system by motion detection and open finger counting. *IEEE Transactions on Consumer Electronics* 55, 4 (2009), 2308–2313.
[43] JS Lee, CL Chen, TH Wu, JC Hsieh, YT Wui, MC Cheng, and YH Huang. 2003. Brain activation during abacus-based mental calculation with fMRI: a comparison between abacus experts and normal subjects. In *First International IEEE EMBS Conference on Neural Engineering, 2003. Conference Proceedings.* IEEE, 553–556.
[44] Julien-Charles Lévesque, Denis Laurendeau, and Marielle Mokhtari. 2011. Bimanual gestural interface for virtual environments. In *2011 IEEE Virtual Reality Conference.* IEEE, 223–224.
[45] Edwin M Lieberthal. 1979. *The complete book of fingermath.* McGraw-Hill.
[46] Mingyu Liu, Mathieu Nancel, and Daniel Vogel. 2015. Gunslinger: Subtle arms-down mid-air interaction. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology.* 63–71.
[47] Gan Lu, Lik-Kwan Shark, Geoff Hall, and Ulrike Zeshan. 2012. Immersive manipulation of virtual objects through glove-based hand gesture interaction. *Virtual Reality* 16 (2012), 243–252.
[48] Meng Ma, Pascal Fallavollita, Séverine Habert, Simon Weidert, and Nassir Navab. 2016. Device-and system-independent personal touchless user interface for operating rooms: one personal UI to control all displays in an operating room. *International journal of computer assisted radiology and surgery* 11 (2016), 853–861.
[49] Cleborne D Maddux, Dennis L Cates, and Virginia M Sowell. 1983. Abacus or Fingermath: How do we Decide? *Journal of Visual Impairment & Blindness* 77, 5 (1983), 210–213.
[50] Nathan Magrofuoco and Jean Vanderdonckt. 2019. Gelicit: a cloud platform for distributed gesture elicitation studies. *Proceedings of the ACM on Human-Computer Interaction* 3, EICS (2019), 1–41.
[51] Anders Markussen, Mikkel R Jakobsen, and Kasper Hornbæk. 2013. Selection-based mid-air text entry on large displays. In *Human-Computer Interaction–INTERACT 2013: 14th IFIP TC 13 International Conference, Cape Town, South Africa, September 2-6, 2013, Proceedings, Part I 14.* Springer, 401–418.
[52] Anders Markussen, Mikkel Rønne Jakobsen, and Kasper Hornbæk. 2014. Vulture: a mid-air word-gesture keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* 1073–1082.
[53] James Michael McDougal. 1979. *The effect of Chisanbop mathematics calculation on the achievement of fourth grade students.* Ph. D. Dissertation. Citeseer.
[54] Pranav Mistry and Pattie Maes. 2009. SixthSense: a wearable gestural interface. In *ACM SIGGRAPH ASIA 2009 Art Gallery & Emerging Technologies: Adaptation.* 85–85.
[55] Korbinian Moeller, Laura Martignon, Silvia Wessolowski, Joachim Engel, and Hans-Christoph Nuerk. 2011. Effects of finger counting on numerical development–the opposing views of neurocognition and mathematics education. *Frontiers in psychology* 2 (2011), 328.
[56] Meredith Ringel Morris, Andreea Danielescu, Steven Drucker, Danyel Fisher, Bongshin Lee, MC Schraefel, and Jacob O Wobbrock. 2014. Reducing legacy bias in gesture elicitation studies. *interactions* 21, 3 (2014), 40–45.
[57] Meredith Ringel Morris, Jacob O Wobbrock, and Andrew D Wilson. 2010. Understanding users' preferences for surface gestures. In *Proceedings of graphics interface 2010.* 261–268.
[58] Michael Nielsen, Thomas B Moeslund, Moritz Störring, and Erik Granum. 2008. Gesture interfaces. *HCI Beyond the GUI: Design for Haptic, Speech, Olfactory, and Other Nontraditional Interfaces, Elsevier* (2008).
[59] Mohammad Obaid, Felix Kistler, Gabrielė Kasparavičiūtė, Asim Evren Yantaç, and Morten Fjeld. 2016. How would you gesture navigate a drone? a user-centered approach to control a drone. In *Proceedings of the 20th International Academic Mindtrek Conference.* 113–121.
[60] Wayne Piekarski and Ross Smith. 2006. Robust gloves for 3D interaction in mobile outdoor AR environments. In *2006 IEEE/ACM International Symposium on Mixed and Augmented Reality.* IEEE, 251–252.
[61] Julie Rico and Stephen Brewster. 2010. Usable gestures for mobile interfaces: evaluating social acceptability. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* 887–896.
[62] Isabel Benavente Rodriguez and Nicolai Marquardt. 2017. Gesture elicitation study on how to opt-in & opt-out from interactions with public displays. In *Proceedings of the 2017 ACM International Conference on Interactive Surfaces and Spaces.* 32–41.
[63] Nathaniel Rossol, Irene Cheng, Rui Shen, and Anup Basu. 2014. Touchfree medical interfaces. In *2014 36th Annual International Conference of the IEEE Engineering in Medicine and Biology Society.* IEEE, 6597–6600.
[64] Gustavo Alberto Rovelo Ruiz, Davy Vanacken, Kris Luyten, Francisco Abad, and Emilio Camahort. 2014. Multi-viewer gesture-based interaction for omni-directional video. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* 4077–4086.
[65] Jaime Ruiz and Yang Li. 2011. DoubleFlip: a motion gesture delimiter for mobile interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems.* ACM, 1979341, 2717–2720. https://doi.org/10.1145/1978942.1979341
[66] Jaime Ruiz, Yang Li, and Edward Lank. 2011. User-defined motion gestures for mobile interaction. In *Proceedings of the SIGCHI conference on human factors in computing systems.* 197–206.
[67] T Scott Saponas, Chris Harrison, and Hrvoje Benko. 2011. PocketTouch: through-fabric capacitive touch input. In *Proceedings of the 24th annual ACM symposium on User interface software and technology.* 303–308.

[68] Shaishav Siddhpuria, Keiko Katsuragawa, James R Wallace, and Edward Lank. 2017. Exploring at-your-side gestural interaction for ubiquitous environments. In *Proceedings of the 2017 Conference on Designing Interactive Systems*. 1111–1122.
[69] Chaklam Silpasuwanchai and Xiangshi Ren. 2015. Designing concurrent full-body gestures for intense gameplay. *International Journal of Human-Computer Studies* 80 (2015), 1–13.
[70] Peng Song, Wooi Boon Goh, William Hutama, Chi-Wing Fu, and Xiaopei Liu. 2012. A handle bar metaphor for virtual object manipulation with mid-air interaction. In *Proceedings of the SIGCHI conference on human factors in computing systems*. 1297–1306.
[71] Firat Soylu, Frank K Lester Jr, and Sharlene D Newman. 2018. You can count on your fingers: The role of fingers in early mathematical development. *Journal of Numerical Cognition* 4, 1 (2018), 107–135.
[72] Josien C Van Den Noort, Nathalie Van Beek, Thomas Van Der Kraan, DirkJan HEJ Veeger, Dick F Stegeman, Peter H Veltink, and Huub Maas. 2016. Variable and asymmetric range of enslaving: Fingers can act independently over small range of flexion. *Plos one* 11, 12 (2016), e0168636.
[73] Radu-Daniel Vatavu. 2013. There's a world outside your TV: exploring interactions beyond the physical TV screen. In *Proceedings of the 11th european conference on Interactive TV and video*. 143–152.
[74] Radu-Daniel Vatavu. 2017. Characterizing gesture knowledge transfer across multiple contexts of use. *Journal on Multimodal User Interfaces* 11, 4 (2017), 301–314.
[75] Radu-Daniel Vatavu and Ionut-Alexandru Zaiti. 2014. Leap gestures for TV: insights from an elicitation study. In *Proceedings of the ACM International Conference on Interactive Experiences for TV and Online Video*. 131–138.
[76] JP Verma. 2015. *Repeated measures design for empirical researchers*. John Wiley & Sons.
[77] Vanessa Vuibert, Wolfgang Stuerzlinger, and Jeremy R Cooperstock. 2015. Evaluation of docking task performance using mid-air interaction techniques. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction*. 44–52.
[78] Chat Wacharamanotham, Kashyap Todi, Marty Pye, and Jan Borchers. 2014. Understanding finger input above desktop devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1083–1092.
[79] Wikipedia contributors. 2021. Mental abacus — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Mental_abacus&oldid=1043409368 [Online; accessed 15-September-2022].
[80] Wikipedia contributors. 2022. Abacus — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Abacus&oldid=1108064048 [Online; accessed 15-September-2022].
[81] Wikipedia contributors. 2022. Chisanbop — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Chisanbop&oldid=1096817061 [Online; accessed 15-September-2022].
[82] Wikipedia contributors. 2022. English Braille — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=English_Braille&oldid=1122582380 [Online; accessed 16-December-2022].
[83] Wikipedia contributors. 2022. Mental calculation — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Mental_calculation&oldid=1125009549 [Online; accessed 5-December-2022].
[84] Wikipedia contributors. 2022. Mental Calculation World Cup — Wikipedia, The Free Encyclopedia. https://en.wikipedia.org/w/index.php?title=Mental_Calculation_World_Cup&oldid=1101504387 [Online; accessed 5-December-2022].
[85] 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*. 1083–1092.
[86] Huiyue Wu, Jianmin Wang, and Xiaolong Zhang. 2016. User-centered gesture development in TV viewing environment. *Multimedia Tools and Applications* 75 (2016), 733–760.
[87] Tung-Hsin Wu, Chia-Lin Chen, Yung-Hui Huang, Ren-Shyan Liu, Jen-Chuen Hsieh, and Jason JS Lee. 2009. Effects of long-term practice and task complexity on brain activities when performing abacus-based mental calculations: a PET study. *European journal of nuclear medicine and molecular imaging* 36, 3 (2009), 436–445.
[88] Zheer Xu, Weihao Chen, Dongyang Zhao, Jiehui Luo, Te-Yen Wu, Jun Gong, Sicheng Yin, Jialun Zhai, and Xing-Dong Yang. 2020. Bitiptext: Bimanual eyes-free text entry on a fingertip keyboard. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*. 1–13.
[89] Zheer Xu, Pui Chung Wong, Jun Gong, Te-Yen Wu, Aditya Shekhar Nittala, Xiaojun Bi, Jürgen Steimle, Hongbo Fu, Kening Zhu, and Xing-Dong Yang. 2019. Tiptext: Eyes-free text entry on a fingertip keyboard. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*. 883–899.
[90] Gareth Young, Hamish Milne, Daniel Griffiths, Elliot Padfield, Robert Blenkinsopp, and Orestis Georgiou. 2020. Designing mid-air haptic gesture controlled user interfaces for cars. *Proceedings of the ACM on Human-Computer Interaction* 4, EICS (2020), 1–23.
[91] 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. *arXiv preprint arXiv:2006.10214* (2020).
[92] Maozheng Zhao, Henry Huang, Zhi Li, Rui Liu, Wenzhe Cui, Kajal Toshniwal, Ananya Goel, Andrew Wang, Xia Zhao, Sina Rashidian, Furqan Baig, Khiem Phi, Shumin Zhai, IV Ramakrishnan, Fusheng Wang, and Xiaojun Bi. 2022. EyeSayCorrect: Eye Gaze and Voice Based Hands-Free Text Correction for Mobile Devices. In *27th International Conference on Intelligent User Interfaces* (Helsinki, Finland) *(IUI '22)*. Association for Computing Machinery, New York, NY, USA, 470–482. https://doi.org/10.1145/3490099.3511103

A APPENDIX

Algorithm 1: Interacting with Abacus Gestures

input :
CF : camera feed
ΔT : stabilizer threshold

begin
  $stable\_gesture \leftarrow empty$
  $last\_\Delta t\_gestures \leftarrow empty$
  while CF *is active* **do**
    $t \leftarrow$ current timestamp
    $frame_t \leftarrow GetFrameAtTimestamp(CF,\ t)$
    $\boldsymbol{K^D} \leftarrow GetHandKeypoints(frame_t,$ Dominant$)$
    $\boldsymbol{K^{D'}} \leftarrow TransformHandKeypoints(\boldsymbol{K^D})$
    $\boldsymbol{K^{ND}} \leftarrow GetHandKeypoints(frame_t,$ Non-dominant$)$
    $\boldsymbol{K^{ND'}} \leftarrow TransformHandKeypoints(\boldsymbol{K^{ND}})$
    $abacus\_gesture_t \leftarrow CalculateAbacusGesture(\boldsymbol{K^{D'}},\ \boldsymbol{K^{ND'}})$
    if $abacus\_gesture_t = 0$ **then**
      $delimitGesture(stable\_gesture)$
      $stable\_gesture \leftarrow empty$
    else
      if $GestureIsStable(abacus\_gesture_t,\ last\_\Delta t\_gestures,\ \Delta T)$ **then**
        $stable\_gesture \leftarrow abacus\_gesture_t$
      end
    end
  end
end

Algorithm 2: Calculating Abacus Gesture

input :
$\boldsymbol{K}^{D'}$: a matrix of 21 transformed keypoint vectors on the dominant hand
$\boldsymbol{K}^{ND'}$: a matrix of 21 transformed keypoint vectors on the non-dominant hand
output:
$finger\_abacus\_count$: total finger abacus count from the keypoints

```
begin
    finger_abacus_count ← 0
    if points from the dominant hand then
        if thumb is open then /* see Table 1 for the conditions */
            finger_abacus_count ← finger_abacus_count + 5
        end
        if index is open then
            finger_abacus_count ← finger_abacus_count + 1
        end
        if middle is open then
            finger_abacus_count ← finger_abacus_count + 1
        end
        if ring is open then
            finger_abacus_count ← finger_abacus_count + 1
        end
        if little is open then
            finger_abacus_count ← finger_abacus_count + 1
        end
    end
    if points from the non-dominant hand then
        if thumb is open then
            finger_abacus_count ← finger_abacus_count + 50
        end
        if index is open then
            finger_abacus_count ← finger_abacus_count + 10
        end
        if middle is open then
            finger_abacus_count ← finger_abacus_count + 10
        end
        if ring is open then
            finger_abacus_count ← finger_abacus_count + 10
        end
        if little is open then
            finger_abacus_count ← finger_abacus_count + 10
        end
    end
end
```