手指人偶: 基於手指走路的虛擬替身操作介面

梁中瀚
National Taiwan University
Taipei, Taiwan
r09922a02@ntu.edu.tw

黃大源
Simon Fraser University
Vancouver, BC, Canada
dayuan.huang@outlook.com

陳炳宇
National Taiwan University
Taipei, Taiwan
robin@ntu.edu.tw

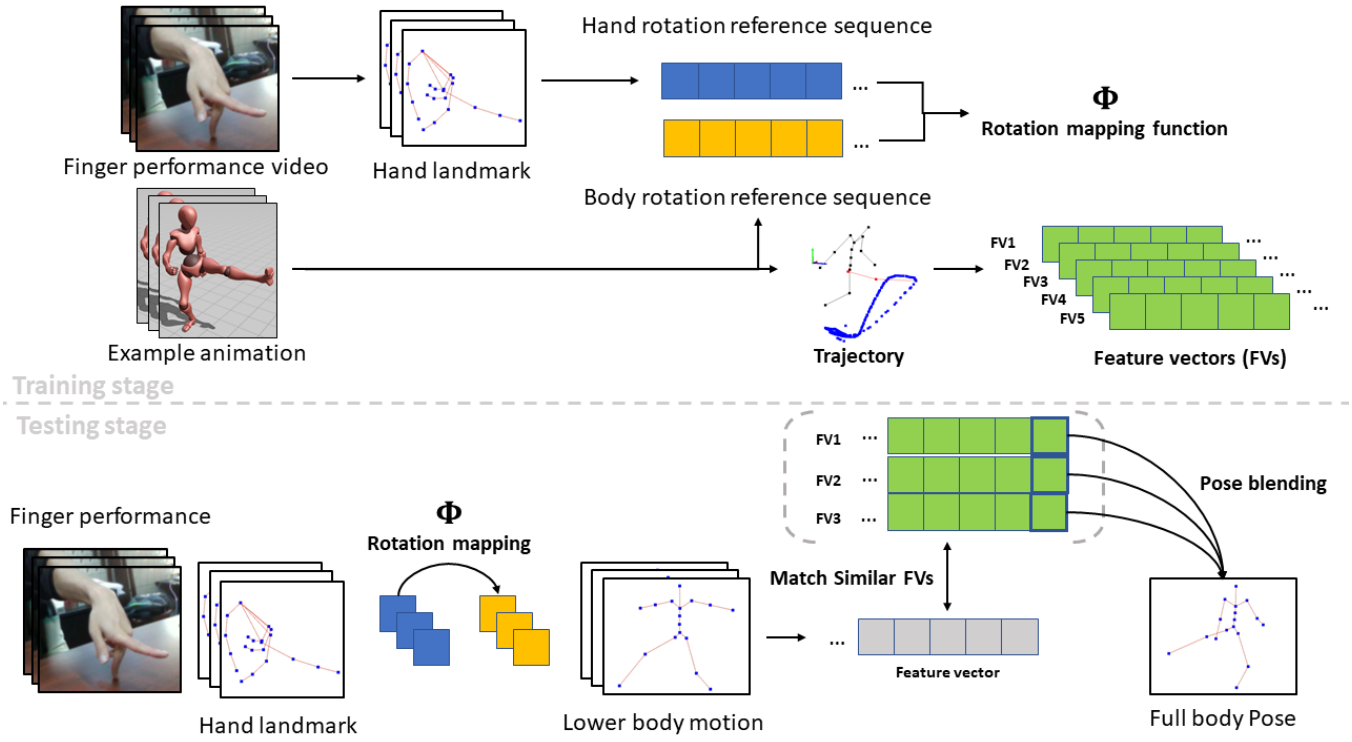

Figure 1: Illustration of the proposed method for finger-walking based puppetry. During the training stage, a rotation mapping function and multiple feature vectors are constructed. In the testing stage, the mapping function is used to map finger rotation to lower body rotation. Then, feature vectors generated from the rotation-mapped lower body motion and the training stage are matched to blend a final full body pose.

ABSTRACT

利用表演達成虛擬操偶的模式已經被廣泛運用在多種不同領域上，包含但不限於遊戲、故事敘述、動畫編輯。人類靈巧的手可以做出非常豐富的動作，正因如此它適合作為操作虛擬角色的介面。我們採用”手指走路”這種直覺且自然的表演方式，用來作為操作虛擬人型角色的介面。首先，我們利用初步的使用者訪談，藉此得到大多數一般的使用者對於”手指走路”的認知範圍。並且，將這些收集到的”手指走路”動作分成不同動作類別。最後，從一個廣泛被使用的動畫資料庫當中，挑選出 5 個多數受訪者認同適合使用”手指走路”表達的虛擬角色動畫為範例動畫。我們提出的虛擬角色操作方法，它會透過轉換”手指走路”過程中的旋轉角度變化到虛擬角色的雙腿達成動作轉移。接著，從範例動畫當中尋找相似的腿部動作用於全身的姿態重建。我們也實作一個互動故事敘述應用程式，藉此展示我們提出的方法有能力產生富有回饋感與可靠的虛擬人型角色動作。

TAICHI '23, Aug 19–20, 2023, Taipei, Taiwan

ACM ISBN 978-1-4503-XXXX-X/18/06…$15.00
https://doi.org/XXXXXXX.XXXXXXX

CCS CONCEPTS

• **Human-centered computing** → **Gestural input**.

KEYWORDS

finger-walking, motion retargeting, performance-based input

ACM Reference Format:
梁中瀚, 黃大源, and 陳炳宇. 2023. 手指人偶: 基於手指走路的虛擬替身操作介面. In *TAICHI '23 The 9th Annual Conference of Taiwan Association of Computer-Human Interaction, Aug 19–20, 2023, Taipei, Taiwan.* ACM, New York, NY, USA, 13 pages. https://doi.org/XXXXXXX.XXXXXXX

1 INTRODUCTION

Computer puppetry, or digital puppetry, is a rising research area, and its technique has been adopted in various applications, such as gaming, storytelling, animation editing, etc. Keyboard and mouse are well-known conventional interfaces for digital puppetry in animation editing. Joystick with buttons and an analog stick is also a common interface to interact with digital avatars in gaming. With the improvement and increased affordability of tracking devices, many low-cost, commodity devices such as Microsoft Kinect, Leap Motion, Nintendo Wii, and Nintendo Switch utilize the human body's motion as an interface for interacting with virtual environments, have become available. Moreover, there has been significant research into novel interaction interfaces for animation editing and storytelling, including those that use tangible or user-specified input devices [7, 8].

Performance-based interfaces, which utilize full-body or partial-body movements to achieve a stronger sense of character control, have gained popularity in the field of virtual avatar manipulation. This is due to the greater relevance between the movement of the user and the movement of the avatar. Besides that, the increasing popularity of virtual reality (VR) and augmented reality (AR) devices and applications has enabled users to experience the movements of a 3D avatar in an immersive virtual world. The embodiment of the interaction interface is critical to the illusion of controlling the avatar in the virtual environment, as typical interfaces such as mouse, keyboard, and joystick cannot provide an intuitive control, potentially disrupting the illusion.

The hand performance interface is one type of performance-based interface, which offers the advantage of requiring minimal performance space, little physical effort, and no additional special device compared to hand-held device interfaces while also preserving a partial sense of embodiment when compared to full-body performance interfaces. Finger-walking, a type of hand performance, is a naturally chosen and widely accepted performance scheme used as a character manipulation interface. According to a user study in [18], participants tended to use the middle and index fingers of their dominant hand to mimic the leg movement of the corresponding full-body motion. The finger-walking performance was found to be an intuitive way of manipulating human avatars, with the embodiment being fulfilled through the relevance of the movement of the fingers to the leg movement.

To achieve finger-walking performance in character manipulation tasks, a method that can transfer the performance to avatar body motion is necessary. However, transferring motion between two different articulated structures is a non-trivial task. While a gesture recognition approach can generate elegant motion similar to an example animation, it can suffer from noticeable delays that can cause leg movement to become asynchronous with finger movement. This limitation renders the approach unsuitable for certain applications such as gaming and storytelling. The first idea that comes to mind to address the issue of asynchronization is direct mapping, which can be achieved by using either contact data from the fingertips or the rotation of the finger joints. While this approach can achieve full synchronization between the finger and leg movements, it can also result in unrealistic leg motion. Furthermore, since finger-walking only mimics leg movement, users may desire to see upper body movement that is relevant to the leg motion, which cannot be generated by direct mapping alone. To address the aforementioned issues, we propose a method that utilizes an example animation to adjust the direct mapping method and make the resulting leg movement feasible. Additionally, we use the example animation to generate corresponding full-body motion that is realistic and expressive, making it suitable for various applications. The full process of the proposed method is illustrated in Figure 1.

In this work, we aim to design and develop a method to achieve real-time finger-walking performance-based puppetry for a human avatar. The key contribution of our method lies in its ability to retarget the motion of two distinct, articulated structures, i.e., the hand motion to the full body motion, thereby resulting in a seamless, synchronous movement of the avatar in response to the user's finger movements. Moreover, the intuitive and expressive method, which requires only a commonly available RGB camera for motion capture, is suitable for various applications such as gaming, storytelling, animation editing, etc. To explore more feasible finger-walking performances and corresponding full-body motion for human avatar puppetry, we first conducted a user study and categorized the collected finger-walking performances. Through the study, we discovered several classes of finger-walking movements that casual users prefer to use for human avatar puppetry and implemented these movements in our proposed method. We have also designed and developed a simple storytelling application to showcase the effectiveness of our proposed performance-based puppetry method. The application allows users to select an action from a 2D user interface and manipulate a 3D virtual avatar through finger performance to create a digital narrative. This demonstrates the ease of use and versatility of the method.

In summary, this work has three main contributions:

- Introduced a motion retargeting method for different articulated structures, with a focus on the finger and human body movements.
- Proposed a finger-walking performance-based puppetry method for manipulating a 3D human avatar, ensuring that the avatar's motion accurately synchronizes with the user's performance, while not losing the elegant and skillful movement in the example animation.
- A user study investigates preferences of the finger-walking performances for mimicking most of the movements in real life.

The remainder of this paper is organized as follows. In the Related Work section, we present three types of puppetry interface studies and compare our work with theirs. The finger-walking performance-based puppetry method we propose is described in the Materials and Methods section, including data preprocessing, motion retargeting, and full-body motion reconstruction. Our storytelling application is presented in the Results section, followed

by a discussion of limitations and potential improvements in the Limitations and Future Work section. Finally, we conclude the paper with a summary of our experiments in the Conclusion section.

2 RELATED WORK

According to the previous study by Ahuja et al. [1], sense of agency (control over avatar) and sense of embodiment (body ownership) are crucial components of an immersive interaction interface. They argued that typical interfaces such as mouse, keyboard, and joystick can only provide a sense of agency. To address this, several studies have proposed novel interfaces for virtual avatar manipulation [1, 8, 18, 30]. These interfaces can be roughly divided into three categories: body performance, hand-based performance, and hand-held devices.

2.1 Body performance interface

The use of body performance as an interface for manipulating the movement of a human avatar is both intuitive and common. This is because the movement of each body part can be directly mapped to the corresponding body part of the human avatar, making it the most embodied interface. Additionally, some studies have utilized body performances to manipulate non-human avatars, and these methods are valuable for their motion retargeting capabilities between different articulated structures.

Several studies has utilized body performance for human avatar manipulation. Chai et al. and Ishigaki et al. [3, 11] used optical tracking technique to capture body motion from users for human avatar manipulation, while Liu et al. [17] utilized inertial sensors to capture body motion of users. They all integrate several prerecorded motion examples and online user performances for the reconstruction of full-body animation. Ahuja et al. [1] proposed a method for reconstructing full-body motion in VR applications using only the motion of two hands. The method accentuates user performance and the output motion is similar to an example animation clip. They underline the embodiment of their method, as it is crucial for the immersive experience of virtual characters in VR. The method we propose in this study attempts to reconstruct full-body motion by searching for short motion examples in animation clips that match the movement of the two legs. It is similar to the one proposed by Ahuja et al. [1], which reconstructs full-body motion using only part of the body's movement.

Part of studies manipuates non-human avatar via body performance. Yamane et al. [29] utilized small set of corresponding key poses of human motion capture data and non-humanoid character to learn a statistic model for non real time retargeting. Shiratori et al. [25] used motion sensors in Wiimote that capture the acceleration of sparse body parts for generating locomotion of a non-humanoid biped avatar in real time. They analyzed the accleration data and computed five features for virtual avatar manipulation, i.e., whether the Wiimote is moving or not, frequency, phase difference between two Wiimotes, amplitude, and direction of inclination. Sakashita et al. [23] used kinect to capture users body motion and special photoreflector arrays to dectect performer's movement of the mouth for manipulating a physical puppet. And used direct rotation mapping which maps the rotation of performer's arm joints to puppet's hand joint. Leite et al. [14] used body motion to manipulate virtual shadow puppet in real time. They direct mapps the captured performer's body motion to the shadow puppet by constraining rotations of XY axis, which the Z axis shoots through the 2D surface. Seol et al. [24] utilized pose' s features to control non-human avatars. They proposed a method that used direct feature mapping and motion coupling, allowing for blending two motions from different categories. This allows different body parts of the avatar to perform motions from different categories. The method is efficient, as it avoids costly retargeting computation by using motion coupling, and can be executed in real time. Unlike the methods mentioned above, Chen et al. [4] proposed a slightly different approach to avatar manipulation. Instead of retargeting between bones or articulated joints, they decided to bind the meshes of the target avatar to the user's skeleton. This provides an intuitive and seamless control experience for specific body parts. However, this approach introduces an extremely flexible control mechanism, resulting in generated motion that may not be as elegant as prerecorded animations. An upper body gesture-based approach for character manipulation is proposed by Hung et al. [10]. Their approach requires the user to complete the entire gesture in order to manipulate the virtual character. However, the asynchrony between the user's hand gesture and character motion can decrease the sense of embodiment.

Unlike the method proposed by above works, which used direct rotation or feature mapping [14, 23, 25], key pose pairing by statistical model [29], motion coupling [24] for retargeting, binding target avatar's meshes to user's skeleton [4], and gesture-based recognition [10]. We adopted a two step retargeting method. First, a rotation mapping is constructed to retaget partial body motion. Then, a full-body pose reconstruction is established through matching similar feature vectors computed from example animations and the retargeted partial body motion, as proposed by Ahuja et al [1]. This approach integrates the performance of the user's fingers with an example animation, resulting in a motion that is synchronized with the finger movements. Additionally, these studies used body motion as an interface for interaction with virtual avatar, but required a significant amount of physical effort from the user to perform desired motions and a large tracking or performance area [9, 15]. On the contrary, using hand performance or hand-held tangible devices to control virtual avatars is a viable option when there are restrictions on the moving area or when the intended movement is difficult to perform physically.

2.2 Hand-held devices interface

To address the issues in the body performance interface, some other researchers come up with using hand-held device as manipulation interface.

Ye et al. [30] utilized the gyroscope in smartphones to generate animation of human avatars. They focus on animation editing through controlling smartphones, which is similar to the way of playing a doll for storytelling. However, the system does not generate animation in real time, which is not suitable in gaming and storytelling. Anderegg et al. [2] also utilized smartphones as a virtual character control interface, providing a real-time interactive experience. Specifically, they proposed MotionStick, which offers

a more intuitive control mechanism allowing users to control the virtual character at a certain distance in front of the phone's camera. However, users employing this mechanism control the character in an indirect manner, as the relevance between the user's gesture and character motion is low, leading to a decreased sense of embodiment. Held et al. [8] proposed a method to manipulate virtual 3D objects, which are reconstructed by the user, through a similar object in reality. However, the method only supports simple manipulation of rigid objects, such as moving a car model forward or backward and turning it around. Objects with articulated joints, such as the hand or leg movement of a human avatar, are not supported. In [9], a physical puppet with attached markers was proposed for animation editing. The puppet interface searches for a suitable motion in a database that is similar to the motion created by the user with the physical puppet. However, it does not generate animation in real-time.

A hand-held device interface requires less space for manipulation tasks, but often involves the use of specialized hardware that may need careful calibration, making it less accessible for casual users. Moreover, the use of hardware to map human avatar's body parts can decrease the sense of embodiment in manipulation.

2.3 Hand-based performance interface

Hand performance can be an appropriate interface for applications with restrictions on physical space or motions that are difficult and require significant physical effort, such as jumping high or performing round kicks. Wang et al. [27] have noted three common hand performances styles: finger-walking, glove puppet, and marionettes. These performance styles have a long history in various cultures. In finger-walking, two fingers are used to imitate the movement of legs. In glove puppet performance, two fingers represent the movement of two hands and another finger represents the head. The marionette is also a widely recognized performance type, in which rods and strings connect to the puppet and the performer manipulates it through these strings with their fingers. The results of the user study in [18] indicate that finger-walking is the most natural way to manipulate the motion of a human avatar. The majority of users chose to use their index and middle fingers to imitate the leg motion.

Some studies used finger-walking performance as interface for human avatar manipulation. In [18], a system for full-body animation editing is proposed that uses a touch-sensitive tabletop to capture contact data of finger-walking performance. The captured contact data is then used to search for suitable short motion clips, resulting in full-body animation. Since the system utilizes contact data to generate animation, only certain finger performances that have a strong correlation with contact on a plane can be employed. Lam et al. [13] proposed another method that uses finger-walking data to edit animations, captured by a data glove that records the rotation of finger joints. The rotation is then retargeted to the leg and arm joints of a human avatar, based on the assumption of symmetrical and cyclic movements. However, this assumption restricts the body animations that can be retargeted. In addition, the method retrieves Euler angles along a single axis from an example body motion and employs interpolation for mapping, which can result in unnatural and undesirable motion. Our method not only utilizes quaternion to represent joint rotations but also incorporates motion adjustment through predefined animations, resulting in a more natural and reliable character motion.

In addition to the finger-walking style performance, glove-puppet and marionette (string puppet) performances have also been widely studied in various researchers. One such method, described in [20] involves generating glove puppet animations using hand motion captured by a data glove. The data glove is used to control the puppet in a virtual world through glove puppet performance, which is similar to finger-walking but controls the upper body of the puppet. Procedural animation is then employed to generate motion for the puppet's feet or body, compensating for the motion generated by the data glove. However, the compensated lower body motion through procedural animation may be asynchronous with the upper body motion transferred from the data glove. The method we propose is capable of generating seamless full-body motion by extracting appropriate poses from an example animation. Way et al. [28] proposed similar glove puppet technique and provided multiplayer platform for users to interact with each other in a virtual environment. In Mani-Pull-Action [15], a comprehensive introduction to hand anatomy and biomechanics is presented, along with a hand mapping model between the human hand and a virtual avatar inspired by traditional marionette and glove puppet techniques. Oshita et al. [22] uses Leap Motion to detect hand movements for avatar manipulation. The proposed interface is inspired by the control mechanism of a real string puppet, which is suspended by approximately 10 strings attached to different parts of the puppet' s body and the controller. However, for novice users, additional training is needed to become familiar with the interface since it does not provide physical feedback compared to a real puppet. Oshita and Fender et al. [6, 21] proposed methods that allow users to simply drag the character' s body parts like a string puppet to perform character actions. The former uses a multi-touch input device that limits the interface's input to 2D and integrates some example postures to generate continuous and physically plausible motion. The latter adopts a data glove for hand movement capturing and allows users to drag parts of the avatar in a 3D virtual world. Other studies have utilized simple gestures or direct mapping between the user's hand and the avatar's hand for avatar manipulation. Cheng et al. and Liang et al. [5, 16] use Leap Motion to capture hand movements. The former uses hand gestures to manipulate a non-human avatar in a storytelling application, while the latter maps the user's hand motion directly to the avatar's hand and uses inverse kinematics to generate corresponding arm motion. Unlike the aforementioned studies that utilized predefined bindings between hand joints and character body joints, an algorithm capable of automatically determining the bindings was proposed in HandAvatar [12]. The authors considered three factors - control precision, intuition, and comfort - through an optimization process. The generated bindings enable users to control specific character body parts by transferring the rotation between joints. However, the character motion generated through direct rotation mapping lacks adjustment by predefined animation, which may result in unnatural character movement. User-defined gestures for character manipulation are explored in Puppeteer [10]. The authors propose a control interface that utilizes finger joint features to recognize the user's hand gesture, which aids in selecting the appropriate

character animation to display. However, their method requires the user to complete the entire hand gesture before the animation can be displayed, which reduces the manipulation experience and introduces asynchrony between the user's gesture and character motion.

In summary, we adopt a simple RGB camera along with a hand pose estimator, Mediapipe, for tracking finger-walking performance. This is because data gloves are often specialized and not affordable for casual users. Additionally, based on previous research [18], finger-walking performance is the first idea that comes to users' minds when attempting to manipulate a human avatar, rather than other hand motions. The two previous studies that employed finger-walking performance [13, 18] have several limitations. Either the performance is constrained to having contact with a surface or the generated avatar motion is unrealistic due to the use of direct Euler angle mapping. Therefore, we focus on finger-walking performance as a human avatar manipulation interface, and proposed a method that can address the above issues.

3 PRELIMINARY STUDY

3.1 Methodology

To investigate additional types of finger-walking performances and determine which body movements casual users prefer to manipulate using finger-walking performances, we conducted a preliminary study. The goal and aim of our preliminary study and other exploratory studies [10, 18] are similar, although we completed a more clearer identification on types of finger-walking movement that casual users may perform to control human avatar. Also, the human body motions corresponding to each type of finger-walking movement are identified. Wang et al. [27] conducted a detailed user study on body skeleton mapping for hands. They explored the mapping by assessing the user's ability to map haptic sensations on the hand back to the full body. In our work, we adopted the hand-closed pose mapping they discovered as the finger-walking control interface. However, our preliminary study did not examine haptic sensations. Instead, we identified finger-walking gestures that utilized the same mapping, which were preferred by users for controlling a human character. To our best acknowledge, we are the first study accomplished such complete investigation and report on finger-walking movement.

Previous studies investigating hand gesture design, such as [10, 18], have often used gaming or animation clips to assess user preferences for character manipulation. However, there are still many foot-based movements that remain unexplored. Furthermore, the Olympic Games feature a wide range of sports with diverse and dynamic movements, including various foot-based actions. Therefore, we screened 28 sports from the Tokyo 2020 Olympics, excluding water sports and redundant sports such as volleyball and beach volleyball. We selected a single movement from a video of a chosen sport for the study.

Five participants (4 male, 1 female) were recruited for the study. They were first shown a video of a single movement from an Olympic game, and then asked to perform the movement using finger-walking and another hand movement with the same hand without restriction. These performances were intended to capture the best expression of the movement from the video, while being within the camera's range to capture their full motion. During the study, participants were allowed to replay the video and perform the movement multiple times. After performing the movement using both finger-walking and another hand movement, participants were asked to choose the best way to perform the movement between the hand and finger performances they just completed and a performance using their entire body. When making their selection, participants were reminded to consider the criteria of goodness, ease of performance, and fatigue. Additionally, a brief survey was conducted to probe the reason for choosing a specific performance as the best expression of the movement.

3.2 Result and Observations

The 5 participants each performed 28 finger-walking motions, each of which corresponded to a movement in a video of an Olympic sport. We categorized these motions into 8 primitive finger movements, excluding the global movements of the entire hand. The categorized result is listed in Table 1, and the categorization of global movements of the entire hand is listed in Table 2. Furthermore, we counted the number of sports in each category that most participants preferred to use finger-walking to mimic the movement in it (more than 3 participants), and the result is illustrated in Figure 2. After classifying all the hand performances into several categories and sum up the number of preferences of finger-walking, we observed that:

- Half of the movements from the sports were preferred to be performed via finger-walking by the participants.
- Body actions with little leg motion and the corresponding finger movements involve minimal finger movements are categorized under the stand category. Most of the participants agreed that those body actions are not well-suited for representation using finger-walking. Instead, participants prefer mimicking upper body movements or hand gestures of the athletes depicted in the video using their entire body or hand, such as drawing a bow or aiming a gun.
- Body actions associated with 'single step' finger-walking movements typically involve interacting with an object using hand and lower body movements, which represent the entire body's global translation. Given that the primary focus of these actions is hand movement, participants prefer mimicking the hand movements using their entire hand, such as pretending to hold a racket and using it to hit a badminton ball.
- Most participants agreed that finger-walking movements, such as "kick" and "front and side split," are suitable for representing certain body actions. The majority of these actions involve leg movements, such as kicking in karate and splitting in ribbon rhythmic gymnastics. In contrast, judo throws do not involve expressive leg and body movements. Rather, only stumbling the opponent by bending a leg is

Table 1: Sports in Olympic Games Tokyo 2020 and the corresponding categories of finger-walking movement. Participants prefer to use finger-walking to perform the movement in the sport, which is in bold text.

Run	**Cycling**, Sport climbing, **Athletics**, Boxing, Table tennis, Tennis, **Wrestling**
Kick	**Football**, **Taekwondo**, **Karate**
Cross fingers	**Skateboarding**
Stand	Shooting, Archery, Golf
Front and side split	**Artistic gymnastic**, **Judo**, **Rhythmic gymnastic**
Single step	Badminton, Fencing, Softball, Handball
Crouch	Equestrian, **Hockey**, Beach volleyball, **Trampoline gymnastic**, **Weightlifting**, **Rugby sevens**

representative. Nonetheless, participants agreed that the rotation and translation of the entire body movement, accompanied by leg bending, can express a judo throw, and these movements can best be represented by finger-walking.

In addition to the aforementioned observations, we identified alternative hand performance techniques that can replace the finger-walking method in certain situations. One participant noted that while most of the action in activities such as badminton, tennis, and table tennis involves hand movements to interact with objects, there are also non-essential leg movements involved in moving the whole body. In these cases, the participant used finger-walking to mimic the translation of the entire body and then switched to representing the hand movements of the action using the same fingers. In addition to this technique, two other participants developed their own approaches. Initially, they attempted to represent both hands and legs using four different fingers, believing that this would best express the entire body action. However, after a few attempts, they found that this technique was difficult to perform and caused their fingers and hands to cramp. Four of the participants agreed that another type of performance, which involves mimicking the hand and finger movements of the action using their own hand to interact with an imaginary object, is suitable for this type of body action. They also believed that during this process, the translation of their hand can represent the displacement of the body. For example, throwing a softball or swinging a tennis racket.

According to Figure 2, actions belonging to "single step" and "stand" are not suitable to perform by finger-walking. We found body actions that are related to other categories in a well known animation database, Mixamo, except the action related to "cross finger". Finally, we chose 5 body motions that related to each of the categories to take part in our proposed system, and they are listed in Table 3.

4 MATERIALS AND METHOD

We proposed a finger-walking performance-based puppetry system which allows user to control virtual avatar by straightforward hand gesture in real time. From the perspective of user, the usage

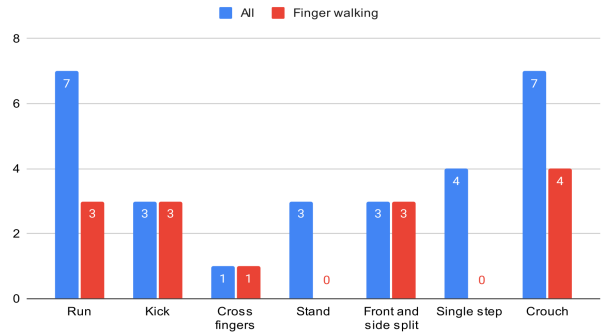

Figure 2: Number of sports in each finger-walking category and number of sports that most participants preferred to use finger-walking to perform

of the system can be divided into two steps: training stage and testing stage. During the training stage, user need to mimic the animation by finger-walking performance. Then the replay of the performance was shown to the user and the time that the movement starts and ends need to be labeled. Also, the time that motion starts and ends in the example animation needs to be labeled. In testing stage, the animation that user intended to perform must be specified. After that, user can perform the animation by hand, and the proposed system is able to show an avatar motion synchronized to the user's hand performance and similar to the desired animation. Figure 3 illustrates the system diagram.

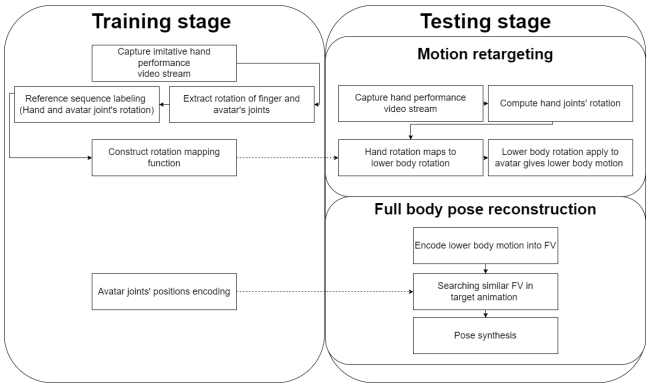

Figure 3: System diagram

In terms of data processing pipeline, some data is preprocessed in the training stage and the testing stage can be divided into two parts: motion retargeting and full body pose reconstruction. The following sections describe the system in detail. First, data collection and data preprocessing steps were introduced in 4.1, we collected hand performance data on our own with a simple RGB camera. And, animation was collected from a well known database: Mixamo. Subsequently, the first data processing step, hand motion retargeting to lower body motion, is explained in 4.2. The other

Table 2: Number of sports in each finger-walking category related to other two taxonomy

	Hand movement				Contact with surface		
	translation	rotation	static	both	In air (no contact)	static	moving
Run	4	0	2	1	2	0	5
Kick	0	0	3	0	0	3	0
Cross fingers	0	0	0	1	0	0	1
Stand	0	2	1	0	0	2	1
Front and side split	1	1	1	0	0	0	3
Single step	4	0	0	0	0	0	4
Crouch	4	0	2	1	1	2	4
Sum	13	3	9	3	3	7	18

Table 3: Example animations used in this study and the corresponding finger-walking category

Example animation	Finger-walking category
Run	Run
Joyful jump	Crouch
Double leg jump	Crouch
Front kick	Kick
Side kick	Front and side split

data preprocessing step, full body pose reconstruction by similar motion searching, is explained in 4.3.

4.1 Data and preprocessing in training stage

Two kinds of data were used as reference information in the proposed hand performance-based animating system, hand performance video and character animation. A simple easy to get RGB camera Realsense from Intel, which was used to capture hand performance video. And the video is recorded in resolution (848, 480) and the frame rate was 60 frames per seconds (FPS). Note that, Realsense camera performs a depth estimation result but it was not used in this study, since the hand pose estimation method we used does not require depth information. Mediapipe [19], a cross-platform machine learning library for streaming data inference was used to estimate 3D hand pose information. Figure 4 illustrates an estimated 3D hand pose by Mediapipe overlay on a hand image and a hand articulation used by Mediapipe which consists of 21 joints. Since finger-walking is trying to mimic two legs motion by index finger and middle finger, only joint of wrist and joints of index and middle finger are used in this study, and they are paired with corresponding avatar's joints of two legs. The relationship was depicted in Figure 5. Those joints' rotation was computed from the estimated pose. The other reference information for the proposed method is character animation that the user intends to perform. There are five animations utilized in this study, collected from Mixamo, including three types of movements, namely running, jumping, and kicking. The used animations were listed in Table 3.

We achieved motion retargeting from hand to lower body using rotation mapping, similar to the method described in [13]. This involved computing the joint rotations in both the hand and avatar. First, a Kalman filter is applied to the landmarks that Mediapipe

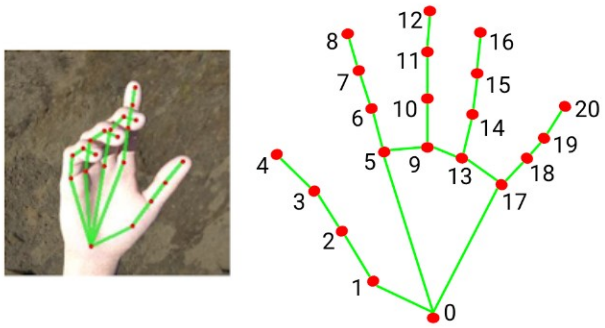

Figure 4: Pose detection result and detected hand articulation by Mediapipe [19]

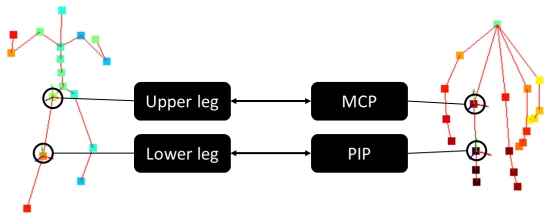

Figure 5: Relationship between finger joints and lower body joints. Mapping of rotation from finger joints to rotation of lower body joints is constructed at the training stage. Finger articulation with index MCP and PIP labeled on the right. Body skeleton hierarchy with upper leg and lower leg labeled on the left.

estimated for smoothing and denoising. In the finger-walking scenario, user only intend to mimic body motion by their index and middle finger. Thus, rotation of Metacarpophalangeal (MCP) and Proximal Interphalangeal (PIP) in index and middle finger was computed. The MCP joint of the hand has two degrees of freedom. The

first degree of freedom involves rotation along the x-axis, which is perpendicular to the palm's normal vector and parallel to the vector connecting the MCP of the index finger and the MCP of the middle finger. This movement is commonly known as the flexion and extension angle [15]. The second degree of freedom involves rotation along the z-axis, which is parallel to the palm's normal vector and perpendicular to the x-axis, and is referred to as the abduction and adduction angle. The PIP is a single degree of freedom joint with rotation occurring along an axis that is parallel to the cross product of the vector from MCP to PIP and the vector from PIP to the Distal Interphalangeal (DIP).

The joints of the hand are paired with the joints of the lower body in finger-walking performance. The rotation of the MCP joint is likely intended to represent the rotation of the upper leg joint, while the rotation of the PIP joint may represent the rotation of the lower leg (knee) joint. Additionally, the degrees of freedom of the MCP and upper leg joint are similar, as are the degrees of freedom of the PIP and lower leg joint. Since hand performance usually manifest a imprecise movement of lower body, it is reasonable to leave some hand movement that are not intend to be representing on the avatar.Thus, even though the DoF of MCP is two, only rotation along one axis was used in motion retargeting for a single category of animation. The proposed system is able to transform the sketchy hand performance into a proper avatar motion similar to the vivid animation that has been specified. For example, when user is trying to mimic a front kick motion by roughly raising index finger. A perfect and synchronous avatar front kick movement is generated by the proposed method, which is similar to the prespecified front kick animation. As mentioned above, the start and end frame of the reference hand performance was labeled in training stage. Maximum and minimum value of the rotation angle from the start frame to end frame are extracted and linear interpolation was used to generate sample values between them. As the movement from user could include delay, unintended finger motion, which might cause an abnormal and asynchronous avatar motion. It is rational to discard the original rotations between the maximum and minimum values, since the rotations are noisy and may cause unacceptable mapping results. The generated sample rotations were defined as Hand Rotation Reference Sequence (HRRS) and be used for motion retargeting.

Orientation of lower body joints can be extracted directly from the animation file, thus extra computation is not needed. Quaternion was used to represent orientation of lower body joints, since manipulating Euler angle along a single axis of a joint is able to get an unexpected movement. Additionally, interpolating in quaternion space ensures a more feasible movement than in Euler angles. On the contrary, rotation of the hand joints are only used for identifying the user's intention, and it is not for animating a character directly. It was represented in Euler angle as a reference. As mentioned earlier, the start and end frame of each animation is labeled in training stage and it is defined as Animation Rotation Reference Sequence (ARRS). Moreover, ARRS was interpolated to the same number of sample rotations as HRRS by Spherical linear interpolation (Slerp). Two reference sequences define a mapping function.

Lower body motion needs to be encoded for full body pose reconstruction, since hand performance does not mimic the motion of upper body. Rotation of lower body joints is computed by the mapping function. Then lower body motion can be computed by applying those rotations. Besides the lower body motion from applying mapped rotation of hand joints, motion from animation also needs to be encoded. By searching for a proper motion from animation, which is similar to the lower body motion computed from hand rotation, a precise and feasible avatar movement can be displayed. Note that, motion from animation needs to set the hip rotation and position to 0, since we do not retarget any finger's or hand's motion to hip motion. Furthermore, applying the mapped finger rotation results in the restriction of movement to the lower body of the avatar, while the upper body remains in a static T-pose. This is illustrated by the avatar in the middle of Figure 6. Motion of body joints is represented by a time series of 3D positions, and it can be seen as a trajectory. We encode a short-term time series of 3D positions into a FV by window function. The window function simply concatenates a time series of 3D positions into a 1 dimensional vector, and the size of the window is set to 10 in this study. We generated additional FVs of motion in the animation by adjusting the speed, as described in the article by Ahuja et al. [1]. Differences in positions within an FV were treated as velocities, which were then multiplied by a constant called the speed ratio. In this study, we used six speed ratios: 0, 0.1, 0.3, 0.5, 1, and 1.5. To generate a new FV, we used the last position in the previous FV as a starting point and added the velocities multiplied by the speed ratio. The purpose of generating these additional FVs was to account for variations in speed between hand and lower body movements in the animation. When there were large differences in the speeds of FVs, the terminal positions of similar FVs may not correspond accurately.

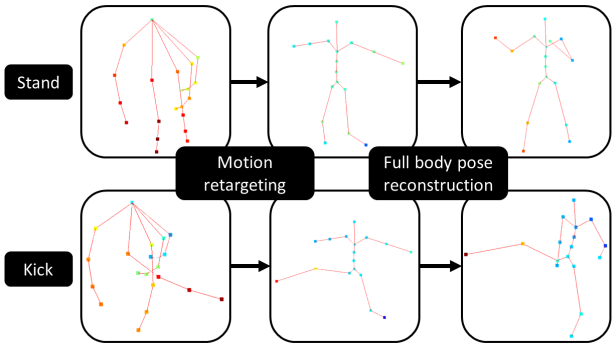

Figure 6: Hand and body poses in each computation step of the testing stage. The figure depicts hand articulation, lower body position after rotation mapping, and full body pose after pose blending from left to right. The leftmost hand pose mimics the stand-up state in the front kick animation, while the rightmost hand pose mimics a kick with the right leg in the front kick animation by raising the index finger.

4.2 Hand motion retargeting to lower body motion

A mapping which maps finger joints' rotation to lower body joints' rotation is constructed at training stage, since they belong to two

different kinematic chain systems and the range of rotation value is different. The mapping method proposed in [13] is referenced, but part of it is modified due to our needs. [13] proposed to use an euler angle along a single axis of avatar' s joint and an Euler angle along a single axis of finger' s joints to construct a mapping function. As the extraction of an Euler angle along a single axis can result in undesired rotation and Euler angle is not suitable for interpolation, we adopt quaternion for representing rotation of avatar' s joint. In contrast, we used Euler angle to represent the rotation of finger joints, as the performance of the hand is an imprecise imitation of the target animation. Moreover, the rotation of finger joints is used exclusively for controlling the avatar and is not interpolated for display to the user. Figure 7 illustrates the difference between using Euler angle along single axis and the quaternion of avatar' s example movement. The target animation in Figure 7 is a side kick movement and the user tried to use abduction of the index finger to imitate the abduction of the leg. The index finger and the corresponding leg is colored in red and the remaining joints and vertices are colored in black. The pose in the second column shows that using Euler angle to do rotation mapping can cause undesired results, which is not similar to the desired example animation (illustrated in the 4th column). The subsequent step will attempt to match both trajectories, which may result in an inaccurate full body pose reconstruction.

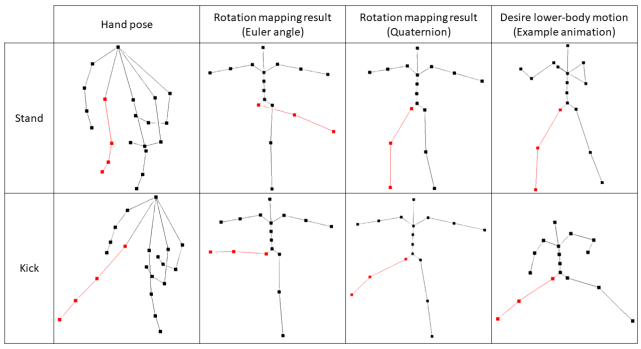

Figure 7: Compare between mapping in euler angle and quaternion. The first row depicts the stand pose in a side kick animation and the second row depicts the pose that kicks to the highest point. The index finger and corresponding leg, which performs the major movements, are colored in red. And the example animation has set the hip position as the origin and set the hip rotation to 0.

In the section Data and preprocessing, we mentioned that starting time and ending time of a specific motion was labeled in the training stage, and two reference sequences were constructed, HRRS and ARRS. Each reference sequence represents rotations through time of a lower body joint or a finger joint. HRRS is defined as a sequence $X = [x_1, x_2, ..., x_n]$, which includes the elements representing rotation of a finger joint at a specific frame. And n represents the number of frames in HRRS, which is set to 100 in this study. ARRS is defined as a sequence $Y = [y_1, y_2, ..., y_n]$, which includes the elements representing rotation of a lower body joint at a specific frame. The frame count of HRRS and ARRS are the same, as we interpolated the same number of frames in the training stage. Then a set of rotation pairs $M = (x_1, y_1), ..., (x_n, y_n)$ can be constructed, which is used for mapping $\Phi : x \rightarrow y$. During testing stage, a finger joint rotation, denoted as $\hat{x}$, is estimated in real time which represents the bending state of the finger joint. Mapping is done by identify a x_t in X, which exhibits the highest degree of similarity to x, and output the corresponding y_t in Y.

After rotation mapping, the result rotation was applied to lower body joints of an avatar. And by utilizing forward kinematic, lower body joints were moved by those rotations. Two legs were considered as two different two-bone kinematic chains, and each chain consists of three joints, upper leg, lower leg and foot. Since hip joint do not rotate by hand rotation, positions of both left and right upper leg joints remain fixed. However, the upper leg and lower leg are rotated by the mapped rotations, thus lower leg and foot joints are moved and generate a temporal trajectory. While hand movements typically approximate an avatar's motion, the animations used in our study feature rich movements in both legs. Therefore, we searched for similar motions in the example animation and in the motion generated by applying mapped rotations, using a single foot trajectory to minimize computational load. As foot is an end effector of a leg, the position of it is influenced by rotation of both lower leg and upper leg joints, thereby implies the position of foot encompasses the information pertaining to both rotations. Additionally, user is sensitive to the synchronization between fingertips and foot of an avatar, when using hand performance to control the avatar. In summary, a foot position of a specific leg is utilized in a search for the desired motion within an animation, which the user intends to perform.

The mapped rotation of a joint is denoted as $\hat{y}$. And the position of the end effector is denoted as $\hat{z}$. Rotations of finger joints are streaming data in the testing stage, and the proposed system maps them to lower body joints rotations then converts them to positions of lower body joints. Thus, mapped rotations of a joint can be denoted as $\hat{Y} = [y_{t-\hat{m}+1}, ..., y_{\hat{t-1}}, \hat{y_t}]$, where $\hat{Y}$ is a sequence consisting of data from m frames. In addition, position sequence of a end effector can be denoted as $\hat{Z} = [z_{t-\hat{m}+1}, ..., z_{\hat{t-1}}, \hat{z_t}]$.

4.3 Full body pose reconstruction

The position sequence of an end effector was encoded to a FV and used for identifying a proper full body pose. As the lower body performs movements that synchronize to finger-motion after retargeting, but the upper body remains still. Position sequence is concatenated into a one dimensional vector $\hat{w} = [z_{t-\hat{m},x}, ..., z_{t-\hat{1},x}, z_{\hat{t},x}, z_{t-\hat{m},y}, ..., z_{t-\hat{1},y}, z_{\hat{t},y}, z_{t-\hat{m},z}, ..., z_{t-\hat{1},z}, z_{\hat{t},z}]$, $z_{\hat{t},x}$ represents the position of the foot along the x axis at frame t. The m in the vector represents the number of frames utilized for encoding a feature vector, and in this study, it is set to a value of 10. Different from the FV encoding method used in [1], we did not use velocity and acceleration as features. The speed of the lower body motion after retargeting is the same as the finger motion, and they might differ greatly from the lower body motion in the example animation. Therefore, the inclusion of velocity and acceleration within the feature vector is not helpful and may decrease the speed of searching for similar FV. As mentioned earlier in the section data and preprocessing, in order to address

the issue of a significant discrepancy in the speed of movement between user performance and animation, we augment the feature vector by incorporating 6 different speed ratios. Positions of end effector were encoded into FVs in training stage as the same way, and it is indicated by a sequence of FVs $S = [f_1, f_2, ..., f_u]$. The scalar t denotes the t-th frame in animation, while the variable u represents the number of feature vectors that have been encoded. To find the top k f_t that are the most similar to $\hat{w}$ with l2 distance as a similarity measurement method, we implemented a K-Nearest Neighbor based on KDTree data structure via Scipy [26]. The scalar k is set to 5 in this study. Each FV f_t is corresponding to a full body pose at t-th frame in animation. k full body poses in animation relevant to the top k f_t are weighted averaged, and the weights are inversely proportional to the distance between f_t and $\hat{w}$. A example of searching for similar f_t encoded from a front kick animation is illustrated in Figure 8. The blue dots represent the last position of each feature vector in sequence S, which can also be interpreted as the full trajectory of the foot motion in the example animation. The first column of the figure shows the last 3D position of the $\hat{w}$ indicated by a green star, while the red dots represent the last 3D positions of the 5 f_t that are most similar to the $\hat{w}$. The second column of the figure displays the same trajectory shown as blue dots, with one of them highlighted in red to indicate the last position of a FV f_t. The corresponding full-body pose of the avatar is illustrated in this column. Finally, the Exponentially Weighted Moving Average (EWMA) technique, given by $P_t = \lambda P_t + (1 - \lambda)P_{t-1}$, is employed for smoothing the reconstructed full body pose. Here, P_t denotes the full body pose at time t, and λ represents a weight controlling the strength of the smoothing process. λ is set to 0.3 is this study.

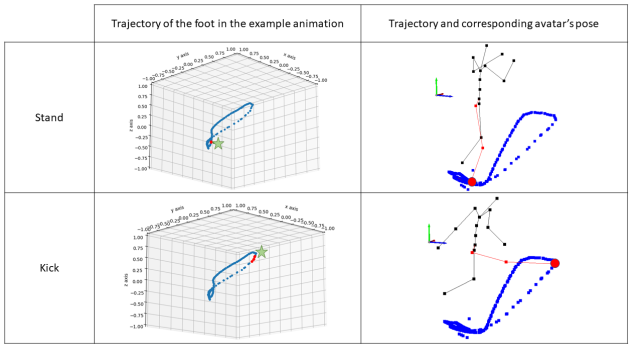

Figure 8: Trajectory matching and full-body pose corresponding to a position related to a FV. This figure used front kick animation to illustrate the trajectory matching result. In the first column, the full trajectory of the foot in example animation was colored in blue, foot position related to mapped rotation from finger motion was depicted by green star, and some foot positions of the full trajectory, which are most similar to the foot position related to mapped rotation, are colored in red. In the second column, the same full trajectory of the foot is illustrated, a single position, which corresponds to the full body pose, is colored in red.

5 RESULT

The goal of our work is to achieve real time character control with agency and embodiment for the user, thus the computations need to be completed in low delay. Many games and movies' refresh rate were 30 frames per seconds (FPS). To let users feel that the character control is synchronized with the character movement shown on the screen, the proposed control system must respond in 30 Hz or above. Respond in 30 Hz means that the latency of computation must be below 0.33 seconds.

We only compute the latency in the testing stage, since there is no need to synchronize hand movement and character shown on the screen in the training stage. Although, we still roughly measure the cost to complete one action's rotation reference sequence and FVs in corresponding animation is in 3 minutes. As illustrated in Figure 3, there are two steps in the testing stage, motion retargeting and full body pose reconstruction. These two steps can be separated into another 5 smaller computation steps, hand pose estimation, rotation mapping, forward kinematic, search for similar FVs and pose blending. The average time to complete the 5 computation steps are listed in Table 4, and there are 5 hand performance videos used as input to the proposed system. Time consumption is measured by utilizing a package *timeit*, which is a standard library in python for measuring the computation time of a specific code snippet . The 5 videos consist of hand performances mimicking 5 different actions, *front kick*, *side kick*, *run*, *joyful jump* and *double leg jump*, and they were recorded around 1 minute. The sum of average computation times of each step is 0.03403, which is a feasible value that the user won't notice a delay between visual avatar movement and movement of hand control. The most computation time is the hand pose estimation by Mediapipe, which is around 0.03 seconds, and the sum of other computation steps' time cost is below 0.01 seconds. Since the bottleneck is computation of hand pose estimation, using a more time cost effective method can reduce the delay.

The processes of the 5 computation steps are stated in the following by using front kick animation as example animation. First, the user must specify an action that is intended to be performed, which is front kick, and the performance is captured by a camera. The hand pose is estimated via Mediapipe by the video stream of the performance. After that, finger joints' rotation $\hat{x}$ is computed from the estimated hand landmarks. The landmarks depicted in the left side of Figure 6 correspond to the hand performance of a front kick. The starting pose for the kick is shown in the first row of Figure 6, and the highest pose achieved during the kick is illustrated in second row of Figure 6. The finger joints' rotation $\hat{x}$ is then mapped to the rotation of avatar's lower body joints $\hat{y}$ by the set of rotation pairs M constructed in the training stage. Subsequently, $\hat{y}$ is applied to the avatar and end effector's position $\hat{z}$ is computed. Besides that, the position of upper leg and lower leg are also computed and shown in the middle of Figure 6 with upper body stays in T-pose. A FV $\hat{w}$ was constructed by encoding multiple positions $\hat{z}$ within a given window size. This FV is then used to search for the most similar FVs within a sequence, $S = [f_1, f_2, ..., f_u]$, which was generated from a target animation during the training stage. The sequence S was generated by sampling the joint positions at a rate of 0.05 seconds per frame

Table 4: Computation time in each process

	Hand pose estimation via Mediapipe	Rotation computation and mapping	Forward kinematic	Searching for similar FVs	Pose blending	Sum
Mean	0.03208	0.00085	0.00010	0.00048	0.00051	0.03403
Standard deviation	0.00378	0.00078	0.00010	0.00102	0.00044	x
90 percentile	0.03241	0.00104	0.00025	0.00054	0.00059	0.03486

and replaying the animation for 15 seconds, resulting in a total of 295 positions sampled and $u = 286$ FVs generated. To further improve the performance, these FVs were augmented by adjusting the speed between the positions, resulting in a final set of 2002 FVs ready for searching. 5 FVs in S, which are most similar to $\hat{w}$, were identified and the relevant full body poses were weighted sum to perform pose blending via distances between the FVs and $\hat{w}$. Finally, blended pose is smoothed by EWMA, which weighted sum the blended pose in the current frame and in the previous frames. 17 body joints were shown on the right side of Figure 6 with different phases of a blended *front kick* pose.

5.1 Storytelling application

A simple storytelling application has been developed to demonstrate the feasibility of the proposed finger-walking performance-based interface. The user interacts with a digital 3D scene and 3D avatar displayed on a monitor, with a RGB camera placed in front of the monitor and a keyboard available for system configuration. Users can use the keyboard to choose the action that is intended to perform, and manipulate 2D translation of the virtual avatar. In addition to the virtual scene and avatar manipulated by the user, a group of 2D user interface (UI) are also displayed on the screen Figure 9. The group of UI consists of six panels, the first panel is showing the current action intended to be performed, the other panels are labeled with action names, which are able to be performed, and a corresponding number. All the actions are listed in Table 3. Users can select the desired action to be performed by the avatar by pressing the corresponding key on the keyboard, for example, press 1 to perform the first action or press 2 to perform the second action.

The storytelling application provides a sample scene where the user plays the role of a human protagonist attempting to get an apple from a tree and return home. During the digital storytelling experience, the user is required to manipulate the protagonist's movements and interact with objects in the virtual environment. In the initial scene, the protagonist is positioned near to a tree and must approach it, followed by kicking the trunk in order to cause an apple to fall from the branches Figure 10a and 10b. In the second scene, the protagonist must cross a road with cars passing by. The user must quickly assess the situation and choose the jump action, deftly dodging the oncoming cars one by one 10c. After that, the protagonist finally walks home and gives a joyful jump 10d and 10e. This application demonstrates the proposed interface' s feasibility. It allows users to manipulate an avatar and have it interact with a virtual environment to complete a digital narrative using hand performance. The interface's capability to manipulate avatar-environment interactions opens up the potential for creating gaming applications.

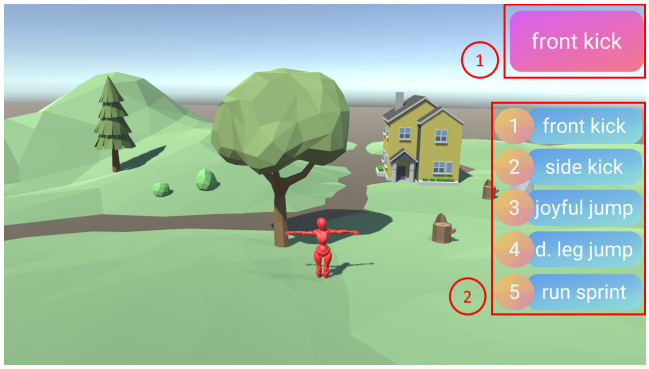

Figure 9: Graphical User Interface in the storytelling application. The top right panel, labeled with 1, shows the current action to be performed. Below the first panel, there are five panels showing the possible action that can be performed, which the user can choose by pressing the corresponding number on the keyboard. This interface was used to allow users to interactively control the actions of a virtual character in the storytelling application.

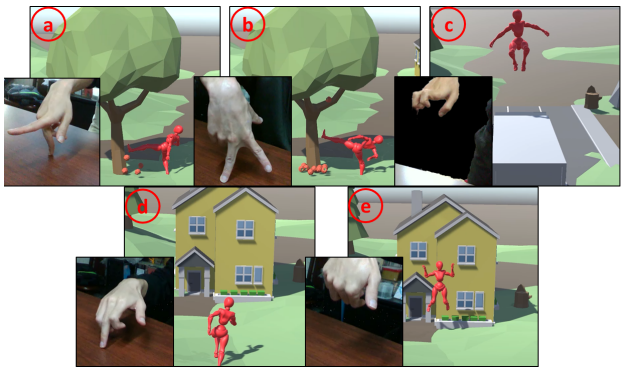

Figure 10: Examples of storytelling scenes and corresponding finger-walking performances. Subfigures (a) and (b) depict the user manipulating an avatar to kick a tree. Subfigure (c) demonstrates the use of double leg jumps to avoid passing cars. Subfigures (d) and (e) show the user attempting to run back home and celebrating with a joyful jump, respectively.

5.2 Limitation and Future Work

Despite the effectiveness of our proposed method in generating avatar motion that is synchronized with the user's hand performance, there are still some limitations and restrictions that need to be addressed. These limitations and restrictions are interesting to be studied in the future. The first limitation is that we use Mediapipe for hand pose estimation, which has a delay of around 0.03 seconds, and is higher than the sum of other computation delays of around 0.01. Most current GPUs can support 60 FPS in standard games, resulting in a delay of around 0.017. Clearly, the hand pose estimation task is the bottleneck and can result in the avatar displayed on the monitor not being fully synchronized with the user's hand movement. In the future, the implementation of a more computationally efficient method for hand pose estimation should be considered in order to reduce the delay time. Additionally, utilizing hardware specifically designed for hand pose estimation is another option that has the potential to significantly accelerate computation and bring the delay of the overall system closer to the delay of modern GPUs on current games.

In the full-body pose reconstruction step, we only considered the right foot FV w in relation to the retargeted hand performance when searching for similar feature vectors in example animations. This is because the example animations used have rich motions on the right foot, causing the user to focus their movements on the index finger, which mimics the right leg motion, rather than the middle finger, and they don't want the middle finger motion to be mapped to the left leg. For example, when mimicking the right leg kicking action with their hand, the user raises their index finger to imitate the right leg kick, but the subtle rotation that occurs in the relaxed middle finger can cause perturbation in the retargeted FV and result in unnatural poses during the subsequent FV search.In future studies, FVs related to both feet can be considered to make the proposed method more general and robust, as there are many animations with rich movements on the left foot or both feet. Another area of interest for future study is to determine whether the consideration of each foot's FV, based on the similarity between the resulting motion and the example animation, can improve the method.

In the motion retargeting step, only one euler angle along a single axis for each finger joint is considered for rotation mapping. The finger joints are the MCP and PIP, and their corresponding body joints are the upper and lower legs. The rotation of the PIP consists of only one euler angle along a single axis, but the MCP rotation consists of two euler angles. In this study, only one euler angle along a single axis of the MCP is used. This is because the example animation adopted in this study is simple and the upper leg joint is rotating in a single direction. Also, human hand performance cannot complete a complex movement consciously, for example, moves in bi-direction and precisely reproduces the same movement several times. As a result, requiring users to perform complex bi-directional movements could result in inconsistent joint rotations, making it an infeasible interface for casual users. In future studies, all euler angles along all axes of each finger joint can be taken into consideration. This will eliminate the need to carefully select the appropriate axis of the euler angle for rotation mapping, resulting in a more streamlined process.

The proposed method aims to generate human avatar motion that is closely synchronized with the user's hand performance, using a predefined example animation as a reference. In future studies, it may be possible to add a hand action recognition method to the system, which would allow it to automatically choose the appropriate example animation based on the user's hand motion, reducing the need for manual selection. However, the hand recognition function requires additional computation time and often finishes recognition after the hand action is completed, potentially causing a delay in the manipulation interface and reducing the embodiment. Future research into finding the optimal method for choosing the intended example animation and the minimum acceptable delay time to preserve embodiment is of interest.

Based on the results of the preliminary study, we found that the body actions of "stand" and "single step" are not suitable for finger-walking performance. "Stand" actions involve minimal leg movement, while the majority of "single step" actions focus on hand and upper body movements with leg motion used mainly for translation. Consequently, these two types of body actions were not implemented in the storytelling system. Exploring feasible hand performances to express these body actions and studying the corresponding retargeting method between hand performance and avatar motion would be interesting areas for future research. Another type of body action that we did not implement in the system is "cross finger." The reason for this is that no suitable body motion was found in the motion database for finger-walking associated with "cross finger." In the future, it would be interesting to identify appropriate body motion of "cross finger" and investigate expressive and comfortable hand movements to perform it.

6 CONCLUSION

We proposed a real-time digital puppetry method that uses finger-walking performance to manipulate a human avatar. The method features a novel interface that is responsive and expressive, enabling users to interact with a 3D virtual environment using hand performance captured by a consumer-level RGB camera. Also, a preliminary study is conducted to explore finger-walking movements that novice users prefer to perform for representing corresponding body motion. We propose a motion retargeting technique for different articulated structures. The technique generates full-body poses by retargeting the user's hand motion to the avatar's lower body motion. Then, it matches the motion trajectory of the retargeted motion to the motions in an example animation to reconstruct the full-body pose. Finally, it blends matched motions from the example animation to provide the user with an intuitive and expressive agency interface over the avatar. The proposed method is useful for 3D gaming and storytelling applications that require an immersive experience in a virtual world. Additionally, a storytelling application has been implemented and demonstrates the usefulness and effectiveness of the proposed method.

ACKNOWLEDGMENTS

To Robert, for the bagels and explaining CMYK and color spaces.

REFERENCES

[1] Karan Ahuja, Eyal Ofek, Mar Gonzalez-Franco, Christian Holz, and Andrew D. Wilson. 2021. CoolMoves: User Motion Accentuation in Virtual Reality. *Proc.*

ACM Interact. Mob. Wearable Ubiquitous Technol. 5, 2, Article 52 (jun 2021), 23 pages. https://doi.org/10.1145/3463499

[2] Raphael Anderegg, Loïc Ciccone, and Robert W. Sumner. 2018. PuppetPhone: Puppeteering Virtual Characters Using a Smartphone. In *Proceedings of the 11th ACM SIGGRAPH Conference on Motion, Interaction and Games* (Limassol, Cyprus) *(MIG '18)*. Association for Computing Machinery, New York, NY, USA, Article 5, 6 pages. https://doi.org/10.1145/3274247.3274511

[3] Jinxiang Chai and Jessica K. Hodgins. 2005. Performance Animation from Low-Dimensional Control Signals. *ACM Trans. Graph.* 24, 3 (jul 2005), 686–696. https://doi.org/10.1145/1073204.1073248

[4] Jiawen Chen, Shahram Izadi, and Andrew Fitzgibbon. 2012. KinÊTre: Animating the World with the Human Body. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge, Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 435–444. https://doi.org/10.1145/2380116.2380171

[5] Yu-Ting Cheng, Timothy K. Shih, and Chih-Yang Lin. 2017. Create a puppet play and interative digital models with leap Motion. In *2017 10th International Conference on Ubi-media Computing and Workshops (Ubi-Media)*. 1–6. https://doi.org/10.1109/UMEDIA.2017.8074081

[6] Andreas Fender, Jörg Müller, and David Lindlbauer. 2015. Creature Teacher: A Performance-Based Animation System for Creating Cyclic Movements. In *Proceedings of the 3rd ACM Symposium on Spatial User Interaction* (Los Angeles, California, USA) *(SUI '15)*. Association for Computing Machinery, New York, NY, USA, 113–122. https://doi.org/10.1145/2788940.2788944

[7] Oliver Glauser, Wan-Chun Ma, Daniele Panozzo, Alec Jacobson, Otmar Hilliges, and Olga Sorkine-Hornung. 2016. Rig Animation with a Tangible and Modular Input Device. *ACM Trans. Graph.* 35, 4, Article 144 (jul 2016), 11 pages. https://doi.org/10.1145/2897824.2925909

[8] Robert Held, Ankit Gupta, Brian Curless, and Maneesh Agrawala. 2012. 3D Puppetry: A Kinect-Based Interface for 3D Animation. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology* (Cambridge, Massachusetts, USA) *(UIST '12)*. Association for Computing Machinery, New York, NY, USA, 423–434. https://doi.org/10.1145/2380116.2380170

[9] Narukawa Hiroki, Natapon Pantuwong, and Masanori Sugimoto. 2012. A puppet interface for the development of an intuitive computer animation system. In *Proceedings of the 21st International Conference on Pattern Recognition (ICPR2012)*. 3136–3139.

[10] Ching-Wen Hung, Ruei-Che Chang, Hong-Sheng Chen, Chung Han Liang, Liwei Chan, and Bing-Yu Chen. 2022. Puppeteer: Manipulating Human Avatar Actions with Intuitive Hand Gestures and Upper-Body Postures. In *Adjunct Proceedings of the 35th Annual ACM Symposium on User Interface Software and Technology* (Bend, OR, USA) *(UIST '22 Adjunct)*. Association for Computing Machinery, New York, NY, USA, Article 3, 3 pages. https://doi.org/10.1145/3526114.3558689

[11] Satoru Ishigaki, Timothy White, Victor B. Zordan, and C. Karen Liu. 2009. Performance-Based Control Interface for Character Animation. *ACM Trans. Graph.* 28, 3, Article 61 (jul 2009), 8 pages. https://doi.org/10.1145/1531326.1531367

[12] Yu Jiang, Zhipeng Li, Mufei He, David Lindlbauer, and Yukang Yan. 2023. *HandAvatar: Embodying Non-Humanoid Virtual Avatars through Hands.* Association for Computing Machinery, New York, NY, USA. https://doi.org/10.1145/3544548.3581027

[13] Wai-Chun Lam, Feng Zou, and Taku Komura. 2004. Motion Editing with Data Glove. In *Proceedings of the 2004 ACM SIGCHI International Conference on Advances in Computer Entertainment Technology* (Singapore) *(ACE '04)*. Association for Computing Machinery, New York, NY, USA, 337–342. https://doi.org/10.1145/1067343.1067393

[14] Luís Leite and Veronica Orvalho. 2012. Shape Your Body: Control a Virtual Silhouette Using Body Motion. In *CHI '12 Extended Abstracts on Human Factors in Computing Systems* (Austin, Texas, USA) *(CHI EA '12)*. Association for Computing Machinery, New York, NY, USA, 1913–1918. https://doi.org/10.1145/2212776.2223728

[15] Luis LEite and Veronica Orvalho. 2017. Mani-Pull-Action: Hand-Based Digital Puppetry. *Proc. ACM Hum.-Comput. Interact.* 1, EICS, Article 2 (jun 2017), 16 pages. https://doi.org/10.1145/3095804

[16] Hui Liang, Jian Chang, Ismail K. Kazmi, Jian J. Zhang, and Peifeng Jiao. 2017. Hand Gesture-Based Interactive Puppetry System to Assist Storytelling for Children. *Vis. Comput.* 33, 4 (apr 2017), 517–531. https://doi.org/10.1007/s00371-016-1272-6

[17] Huajun Liu, Xiaolin Wei, Jinxiang Chai, Inwoo Ha, and Taehyun Rhee. 2011. Realtime Human Motion Control with a Small Number of Inertial Sensors. In *Symposium on Interactive 3D Graphics and Games* (San Francisco, California) *(I3D '11)*. Association for Computing Machinery, New York, NY, USA, 133–140. https://doi.org/10.1145/1944745.1944768

[18] Noah Lockwood and Karan Singh. 2012. Finger Walking: Motion Editing with Contact-Based Hand Performance. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Lausanne, Switzerland) *(SCA '12)*. Eurographics Association, Goslar, DEU, 43–52.

[19] Camillo Lugaresi, Jiuqiang Tang, Hadon Nash, Chris McClanahan, Esha Uboweja, Michael Hays, Fan Zhang, Chuo-Ling Chang, Ming Guang Yong, Juhyun Lee, Wan-Teh Chang, Wei Hua, Manfred Georg, and Matthias Grundmann. 2019. MediaPipe: A Framework for Building Perception Pipelines. arXiv:1906.08172 [cs.DC]

[20] Zhiqiang Luo, I-Ming Chen, Song Huat Yeo, Chih-Chung Lin, and Tsai-Yen Li. 2010. Building Hand Motion-Based Character Animation: The Case of Puppetry. In *2010 International Conference on Cyberworlds*. 46–52. https://doi.org/10.1109/CW.2010.24

[21] Masaki Oshita. 2014. Multi-touch Interface and Motion Control Model for Interactive Character Animation. In *Transactions on Computational Science XXIII: Special Issue on Cyberworlds*, Marina L. Gavrilova, C. J. Kenneth Tan, Xiaoyang Mao, and Lichan Hong (Eds.).

[22] Masaki Oshita, Yuta Senju, and Syun Morishige. 2013. Character Motion Control Interface with Hand Manipulation Inspired by Puppet Mechanism. In *Proceedings of the 12th ACM SIGGRAPH International Conference on Virtual-Reality Continuum and Its Applications in Industry* (Hong Kong, Hong Kong) *(VRCAI '13)*. Association for Computing Machinery, New York, NY, USA, 131–138. https://doi.org/10.1145/2534329.2534360

[23] Mose Sakashita, Tatsuya Minagawa, Amy Koike, Ippei Suzuki, Keisuke Kawahara, and Yoichi Ochiai. 2017. You as a Puppet: Evaluation of Telepresence User Interface for Puppetry. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology* (Québec City, QC, Canada) *(UIST '17)*. Association for Computing Machinery, New York, NY, USA, 217–228. https://doi.org/10.1145/3126594.3126608

[24] Yeongho Seol, Carol O'Sullivan, and Jehee Lee. 2013. Creature Features: Online Motion Puppetry for Non-Human Characters. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Anaheim, California) *(SCA '13)*. Association for Computing Machinery, New York, NY, USA, 213–221. https://doi.org/10.1145/2485895.2485903

[25] Takaaki Shiratori and Jessica K. Hodgins. 2008. Accelerometer-Based User Interfaces for the Control of a Physically Simulated Character. In *ACM SIGGRAPH Asia 2008 Papers* (Singapore) *(SIGGRAPH Asia '08)*. Association for Computing Machinery, New York, NY, USA, Article 123, 9 pages. https://doi.org/10.1145/1457515.1409076

[26] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods* 17 (2020), 261–272. https://doi.org/10.1038/s41592-019-0686-2

[27] Bo-Xiang Wang, Yu-Wei Wang, Yen-Kai Chen, Chun-Miao Tseng, Min-Chien Hsu, Cheng An Hsieh, Hsin-Ying Lee, and Mike Y. Chen. 2020. Miniature Haptics: Experiencing Haptic Feedback through Hand-Based and Embodied Avatars. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems* (Honolulu, HI, USA) *(CHI '20)*. Association for Computing Machinery, New York, NY, USA, 1–8. https://doi.org/10.1145/3313831.3376292

[28] Der-Lor Way, Weng-Kei Lau, and Tzu Ying Huang. 2019. Glove Puppetry Cloud Theater through a Virtual Reality Network. In *ACM SIGGRAPH 2019 Posters* (Los Angeles, California) *(SIGGRAPH '19)*. Association for Computing Machinery, New York, NY, USA, Article 12, 2 pages. https://doi.org/10.1145/3306214.3338564

[29] Katsu Yamane, Yuka Ariki, and Jessica Hodgins. 2010. Animating Non-Humanoid Characters with Human Motion Data. In *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Madrid, Spain) *(SCA '10)*. Eurographics Association, Goslar, DEU, 169–178.

[30] Hui Ye, Kin Chung Kwan, Wanchao Su, and Hongbo Fu. 2020. ARAnimator: In-Situ Character Animation in Mobile AR with User-Defined Motion Gestures. *ACM Trans. Graph.* 39, 4, Article 83 (aug 2020), 12 pages. https://doi.org/10.1145/3386569.3392404