Human-Avatar Interaction in Metaverse: Framework for Full-body Interaction

Kit Yung Lam
kylambd@connect.ust.hk
The Hong Kong University of Science and Technology
Hong Kong, HKSAR

Liang Yang
lyangbl@connect.ust.hk
The Hong Kong University of Science and Technology
Hong Kong, HKSAR

Ahmad Alhilal
aalhilal@ust.hk
The Hong Kong University of Science and Technology
Hong Kong, HKSAR

Lik-Hang Lee
likhang.lee@kaist.ac.kr
KAIST
Daejeon, Republic of Korea

Gareth Tyson
gtyson@ust.hk
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China

Pan Hui*
panhui@ust.hk
The Hong Kong University of Science and Technology (Guangzhou)
Guangzhou, China

ABSTRACT

The metaverse is a network of shared virtual environments where people can interact synchronously through their avatars. To enable this, it is necessary to accurately capture and recreate (physical) human motion. This is used to render avatars correctly, reflecting the motion of their corresponding users. In large-scale environments this must be done in real-time. This paper proposes a human-avatar framework with full-body motion capture. Its goal is to deliver high-accuracy capture with low computational and network overheads. It relies on a lightweight Octree data structure to record and transmit motion to other users. We conduct a user study with 22 participants and perform a preliminary evaluation of its scalability. Our user study shows that Octree with Inverse Kinematic achieves the best trade-off, achieving low delay and high accuracy. Our proposed solution delivers the lowest delay, with an average of 67ms in an environment of 8 concurrent users. It attains a 55.7% improvement over the prior techniques.

CCS CONCEPTS

• **Human-centered computing** → *Mixed / augmented reality*; **Web-based interaction**.

KEYWORDS

Metaverse, Human-Avatar Interaction, Full-body Motion Streaming, Octree Data Structure, Octree-Based Algorithm.

ACM Reference Format:

Kit Yung Lam, Liang Yang, Ahmad Alhilal, Lik-Hang Lee, Gareth Tyson, and Pan Hui. 2022. Human-Avatar Interaction in Metaverse: Framework for Full-body Interaction . In *ACM Multimedia Asia (MMAsia '22), December 13–16, 2022, Tokyo, Japan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3551626.3564936

*Pan Hui is also affiliated with Hong Kong University of Science and Technology, Hong Kong SAR, and University of Helsinki, Finland

MMAsia '22, December 13–16, 2022, Tokyo, Japan

ACM ISBN 978-1-4503-9478-9/22/12…$15.00
https://doi.org/10.1145/3551626.3564936

1 INTRODUCTION

The metaverse refers to a 3D cyberspace consisting of shared virtual environments where people can interact synchronously through their avatars [20]. For example, metaverse users can use their avatars to engage in community meetings, concerts, parties, art showings, sports events, sightseeing, and travel [11]. To enable this, users must be able to exchange real-time motion information with imperceptible latency. Through this, avatars should always correspond to the motion of the user they represent.

In some virtual world systems (such as Decentraland and Roblox), touchscreens, mouse, and keyboards are used to control the pose and motion of the user's avatar. This, however, restricts the avatar's actions to basic activities, e.g., standing, walking, and jumping. Recent emerging VR platforms (i.e., Mozilla Hub[1], Spatial[2] and Meta Workrooms[3]) offer ways to automatically capture human motion and recreate it in the virtual world. However, this primarily covers head and hand movements. For example, although Mozilla Hub and Meta Horizon allow users to recreate upper body parts via a WebXR-enabled headset and hand controllers, other body parts (i.e., elbows and legs) are missing. Further, although Spatial displays the entire body, the body interactions are still limited, e.g., the waist and legs are not supported. It is essential to consider the natural input techniques [6] of user-avatar interaction, and how user movements can naturally convert user intention into actions in virtual 3D environments [24]. Thus, we argue a truly immersive experience requires full body tracking and avatar reconstruction that goes beyond this prior work.

With the above in-mind, we propose a framework for real-time avatar motion capture. We assume a distributed environment, in which multiple users are interacting within the metaverse (connected via a wide area network). Due to this, we focus on designing techniques to reduce the network footprint, such that avatar motion updates can be distributed to all participating users with low delay.

[1]https://hubs.mozilla.com
[2]https://spatial.io
[3]https://www.meta.com/work/workrooms/

To achieve this, we rely on the Octree Data Structure to model user motion. This lightweight tree data structure, in which each internal node has exactly eight children, allows us to capture an array of motions with limited overhead. We then employ an adaptive transmission strategy that dynamically selects the fidelity of avatar motion information to share with other clients. We evaluate our solution via a user study consisting of 22 participants. We show that our Octree solution with Inverse Kinematic achieves the best trade-off. Our evaluation demonstrates both technical feasibility and user acceptability. We confirm that, compared to the baselines, we can capture human motion with higher accuracy, while maintaining low human-avatar motion delay. Our proposed solution delivers the lowest delay, with an average of 67ms in an environment of 8 concurrent users. It attains a 55.7% improvement over the prior techniques. Furthermore, the proposed avatars' representations and motions achieve positive user perception in terms of delay, accuracy, and synchronization.

This paper contributes a human-avatar interaction technique that supports seamless interaction between multiple users in the metaverse. The paper is organized as follows. Section 2 briefly summarizes the existing working literature related to our work. Section 3 introduces our Human-Avatar Interaction Framework. In Section 4, we evaluate the proposed framework in different scenarios. Section 5 is our conclusion notes.

2 BACKGROUND

An immersive experience of physical activities involves not only a scene representation and the capturing of gestures, but also accurate visual feedback of the user's body stance and motions. We start by providing an overview of recent work.

2.1 Human-Avatar Interaction

Avatars serve as the user's representation in the metaverse [10, 25]. There is a substantial body of research in this field. Users' perceptions, including feelings of realism [23], presence [21], trust [8], body ownership [14], and group happiness [31], may be influenced by the appearance of their avatars. These perceptions are influenced by a wide range of variables, including the avatar's face [33], its micro-expressions [30], the degree to which the avatar's body is realized [21], and the avatar's representation [7] or position [28]. As a result, avatars have a large impact on the virtual experience [25].

There are a few noteworthy studies that pertain to full-body avatar interaction [9] and point out a realistic full-body avatar increases the sense of embodiment [23] and a much higher level of immersion [16, 19]. Tracking user body gestures enhances metaverse presence [17]. With the evolution of the metaverse, we see a significant number of events requiring real-time direct full-body interactions. These events present a number of unexplored research opportunities, including how to quickly and precisely detect actions, as well as how to compress and transmit such data.

2.2 Full-body Motion Detection

Human motion detection, such as human body modeling and tracking, is a widely researched topic [4, 22, 32]. Motion sensors are the primary means of acquiring the raw data. These sensors include 3D cameras, IMUs, CW radars, IR radars, and ultrasonic arrays. In studies [13, 36, 39] similar to our study, the tracking and analysis of the human body's skeleton is done using wearable inertial-measurement-unit (IMU) sensors. Note, we choose IMUs because they offer high granularity of user movement data, supported by 3-axis acceleration, 3-axis angular velocity, as well as 3-axis magnetometer signals [35]. We believe this makes IMU sensors well suited for our needs.

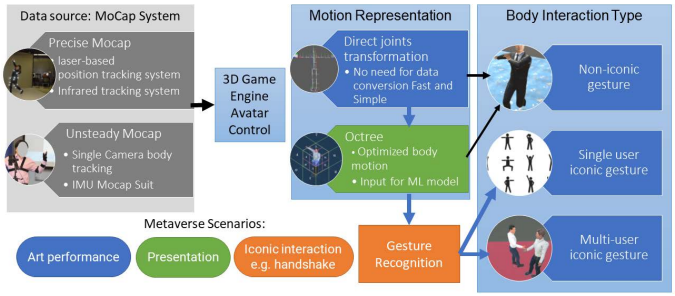


Figure 1: The Interaction framework, capturing the physical motions using MoCap system (left), and passing the joints' positional data to the rendering engine (right).

2.3 Octree-based Algorithm Adaption

An octree is a tree data structure in which each internal node has exactly eight children. When dealing with multi-user and real-time interactions in the metaverse, it is necessary to transmit data in real-time between end users. As such, geometry compression (i.e., the process of coding 3D geometric data in a way that requires less space) is more crucial than ever before. The compression of temporal sequences of geometric data (which is what *animated geometry compression* entails [37]) has received less attention than static geometry reduction. The Octree-based algorithm is one way of performing animated geometry compression. It is suitable for real time applications, offering high compression ratios with reasonable quality. It works by generating a small set of motion vectors for each frame in the sequence by analyzing the motion between consecutive frames. Previous Octree-based studies covers spatial decomposition [18], 3D Animation Compression [38], time series 3D geo-spatial data organization [27], time-varying surfaces compression [12], 3D immersive video codec [29], and non-rigid moving objects [3]. However, these works focus on geometry compression of 3D animation using the Octree data structure. There is an absence of work considering its application to real-time body motion and human-avatar interaction. This is where we place the focus of our work.

3 HUMAN-AVATAR INTERACTION FRAMEWORK

This section describes the design of our human-avatar interaction framework. In particular, we translate the human motion into avatar motion in the metaverse, and reduce the data size of full-body motion streaming using Octree for better scalability.

3.1 Framework Implementation

We first design the 3D model of the users' avatars with full-body skeleton rigged to facilitate their motion in the metaverse. We set up Axis Studio,[4] a motion capture (MoCap) software, on the user's local computer. This software tracks the user's physical motion of via its wireless inertial sensors, the 3rd generation of Perception Neuron.[5]

After retrieving the skeleton tracking data from the MoCap software, our Unity3D application on the local computer translates them into avatar motions. In particular, the joint points are mapped into the motion of the avatar's 3D model. Afterwards, the avatar joint positions are converted into the Octree data structure using the Octomap ţtehornung13auro 3D mapping framework (subsection 3.3). This data is then broadcast to other users' client applications. The remote clients update the avatar's body pose and motion according to Octree Node ID and demodulation of avatar's transform.

In particular, our framework first classifies the body gestures into two types: Non-iconic Gesture and Iconic Gesture [26], in order to help the metaverse application determine which scheme of body-joint data transmission should be employed (subsection 3.2). Here, an iconic Gesture means body gestures that convey any semantic meaning. Non-iconic gestures are the remaining motions. Our framework takes the mask of an avatar pose as the input image for the full-body gesture recognition program [34] to compute the result. The results enable two main features of our framework as follows:

(1) Interaction procedure: The full body gesture recognition results are the trigger points to start, move, and end certain social interactions between two or more users in the metaverse.
(2) Optimization of data granularity of multiple avatars in the network transmission: Some social interaction requires precise body gesture synchronization in the network. The framework can adjust the data streaming scheme based on the gesture recognition results and the distance between two users who are interacting with each other.

We further divide iconic gestures into single-user and multi-user categories. Single-user iconic gestures refer to gestures such as smiling, nodding, etc. Multi-user iconic gestures refer to body gestures that involve two or more users, such as high fives, hugging, shaking hands, etc., which require two or more users to complete a specific social interaction procedure. We apply our framework to the three types of gestures as in Figure 1 (Non-iconic gesture, single user iconic gesture, multi-user iconic gesture); but we take the multi-user iconic gestures as the kernel of this paper, as the focused gesture type serves to support social interactions between human users in diverse virtual spaces. It is important to note that multi-user iconic gestures in real-time are crucial to metaverse users and their seamless interaction. Metaverse users expect responsiveness from other users and this is highly relevant to user experiences. For instance, if any of the users in a social engagement performs iconic gestures, other users should have the ability to acknowledge such gestures so they can respond quickly. Delayed responses, due to improper gestural kernel being selected, would hinder the interaction between avatars and hence deteriorate the user experience.

3.2 Human-Avatar Interaction Data

Our framework can be deployed to any metaverse development platform using the Humanoid Avatars system for the avatars 3D model, such as Unity[6] and Unreal Engine.[7] The common development platforms own Humanoid Avatars module containing no fewer than 15 parts for a connected skeleton. Such modules pave a way that loosely conforms to an actual human skeleton. Our framework leverages the aforementioned modules and constantly captures the user's motion data supported by MoCap software. As a result, the motion representation, also known as the data format of a virtual avatar, drives the joint transformation of the user's avatar. Hip, spine, two legs, thigh and foot, two hands, shoulder and palm, neck, and head were among the 15 key parts of a connected skeleton. The remaining skeleton parts, such as the fingers and upper arms, can be calculated by the human-body Inverse Kinematics (IK) solver. Afterwards, our framework broadcasts the data to the Unity apps of the other users via the WebRTC P2P protocol. The data includes the position and rotation of the avatar's body joints and gesture type. As such, our framework defines two types of motion representations and streaming, as follows.

(1) Direct joint (D-Joints) streaming: The MoCap software calculates joint transformation data, while the local metaverse application translates and applies the data to the avatar 3D model's joint positions and rotations. Then, it broadcasts the translated avatar's joint positions and rotations directly to the remote client applications through WebRTC without any compression. The remote client applications directly apply the received joint transformation data to the corresponding avatar. It is suitable for interactions that require precise motion capture to recreate the avatars. Multi-user social interaction scenarios such as dancing and artistic performance should adopt this scheme.
(2) Octree structure streaming: The local Unity applies the Octree algorithm and broadcast the Octree structure data to the other clients. The positions of the avatar's body joints are converted into Octree IDs, as illustrated in subsection 3.3. This streaming is suitable for MoCap systems that unavoidably have tracking errors such as that using RGB-Camera computer vision. The avatar joints transformation is filtered and stabilized in the local user application, and then broadcast to other clients having avatars in proximity in the metaverse.

In our framework, we apply an adaptive scheme for data streaming to support the aforementioned body gesture types in different metaverse events and scenarios. The adaptive scheme (algorithm 1) determines each avatar individually before broadcasting the data to remote user applications, since the network performance between each remote client is different in the P2P broadcast mode. Remote users should have the metaverse application that can synchronize the avatars' 3D models using our framework. The scheme considers

[4] https://neuronmocap.com/pages/axis-studio
[5] https://neuronmocap.com/pages/perception-neuron-3
[6] https://unity.com
[7] https://www.unrealengine.com/

the network bandwidth and concurrent users in the virtual venue, pre-configured application scenarios (e.g., time-sensitive), gesture type, and distance between the local user and remote users. In particular, the latency between the local and remote user refers to the round trip time (rtt_{A_r}) of data transmission. $\bar{rtt}$ refers to the average round-trip time of the avatar's motion data between the local user and all other remote users, whereas θ_{rtt} refers to latency threshold that is imperceptible to human users (less than 100ms [1]). θ_{rtt} determines the data streaming format (D-joints or Octree). When $\bar{rtt} < \theta_{rtt}$ and the user's gesture belong to iconic multi-user social gestures (e.g., handshake), the user's local application sends D-joints data to all remote users. Otherwise, the algorithm considers the distance and rtt_{A_r} of each remote user individually. θ_{dist} refers to the distance threshold between the local user's avatar and the target user's avatar, which also determines the streaming format.

Algorithm 1. Adaptive scheme

Input: A_{room} ▷ List of remote avatars data in same room
Input: A_{local} ▷ Local avatar data w/ joints transformation(tf)
Input: A_{octree} ▷ Local avatar octree joints data
Input: G_{iconic} ▷ List of multi-user iconic social gesture
θ_{dist} ←Distance threshold ,
$\bar{rtt}$ ←Mean round-trip time between users,
θ_{rtt} ←latency imperceptible to human (<100ms [1])
Function `updateRemoteAvators(`A_{room}`)`**:**
  if $\bar{rtt} \leq \theta_{rtt}$ *AND* $A_{local}.gesture{=}G_{iconic}$ **then**
    foreach $A_r \in A_{room}$ **do**
      send $A_{local}tf$ to A_r;
  else
    foreach $A_r \in A_{room}$ **do**
      if $Distance(A_{local}, A_r) \leq \theta_{dist}$ *AND* $rtt_{A_r} \leq \theta_{rtt}$ **then**
        send $A_{local}tf$ to A_r;
      else
        if $Distance(A_{local}, A_r) \leq \theta_{dist}$ **then**
          send $A_{local}IK$ to A_r;
        else
          if $rtt_{A_r} \leq \theta_{rtt}$ **then**
            send $A_{octree}tf$ to A_r;
          else
            send $A_{octree}IK$ to A_r;
End Function

3.3 Octree Data Structure

Full body motion tracking requires capturing the position and rotation of two legs, two arms, head and torso at least 33 pose landmarks BlazePose [5]. The MoCap system captures no fewer than 51 body joints rotation and positions from head to foot toes and fingers. In 3D game engines, such as Unity 3D and Unreal Engine, the humanoid avatar configuration can control the transformation of 59 key body joints, i.e., direct joints transformation. Using Inverse Kinematics (IK), our framework can achieve human-avatar full-body motion without involving full-body sensing and further reduce the transmission data size. It requires sensing only nine joints (two wrists, two forearms, two upper legs, two feet, and a head), It requires to broadcast only nine joints: two wrists, two forearms, two upper legs, two feet, and the head. It also needs one normalized value for each hand that shows how all five fingers are positioned, from fingers spread out (0) to fist (1), as shown in Figure 2. If one finger value is over 50% larger than the average value of all other fingers, this finger will be disregarded from the average. While this calculation causes the loss of some iconic hand gestures such as the V-sign and thumbs-up, it saves bandwidth for the sake of scalability (see Table 1). The resulting data can be transmitted to the other users with avatars in proximity distance over θ_{dist} to the local user's avatar. Our framework utilizes a Full-body IK (FBIK) solver to calculate the transformation of the remaining avatar's body parts. In particular, it uses the Forward And Backward Reaching Inverse Kinematics (FABRIK) solver [2] for that instead of streaming the transformation and rotation of all the skeleton joints. Instead of employing naive streaming of users' positions and rotations, our framework represents the transformation data of nine joints by mapping them into Octree structure ID. This further reduces the data size before broadcasting.

Figure 2: Finger gestures and corresponding stretch values, from an open hand (value=0, leftmost) to a fist gesture (value=1, rightmost).

We use the Octree to partition the avatar's enclosing three-dimensional space by recursively subdividing it into eight octants. This accelerates locating the body joints. In our framework, we use the Octree 3D representation instead of Direct joints for (*i*) smoothing the motion of the joins as Octree follows the center point of the body. This reduces the jitter and jumping of body joint positions that might occur due to network latency; (*ii*) easier identification of abnormal limb positions due to sensor tracking error; and (*iii*) compressing the avatar's motion data to support virtual events with large number of users.

Our Octree-based algorithm. As illustrated in Figure 3, we set the root Octree bounding box with origin located at the center of the avatar body. The boundary of the first level is set so that no body joint exceeds it. It is determined by the width of the avatar with arms extended, and its height with arms uplifted, and depth with raised leg at 90 degree. The origin is set to the middle of the root Octree and follows the avatar movement. Each cube corresponds to a tree node in the Octree structure and its eight motion vectors are stored in the corresponding tree node. Each node is subdivided into eight smaller cubes when needed. The primary criterion of the subdivision is based on the actual joint position distance to the node center, which is larger than the 1/4 length of the node cube at that level. Using the data retrieved from the Mocap software, the local user application processes the avatar skeleton pose and then

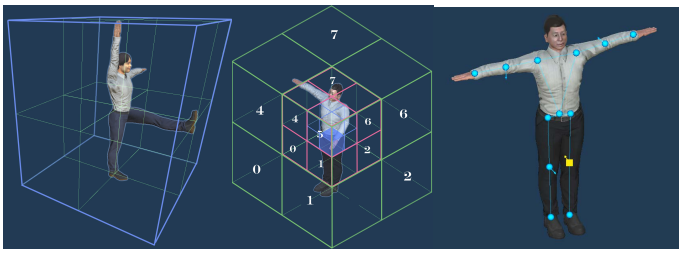


Figure 3: (a) The dimensions of root Octree bounding box is determined by the width with arms extended, height with uplifted arms, and depth with raised leg 90 degree. (b) The left wrist position ID is 531 in our Octree three level. (c) The IK configuration.

searches for the Octree node IDs in which the avatar's joints are located.

4 FRAMEWORK EVALUATION

We next present a preliminary evaluation of our motion capture framework. We evaluate our solution using three metrics. First, we inspect the latency required to disseminate avatar updates to remote users. Second, we measure the associated traffic volume. Third, we rely on a user study to measure the perceived accuracy of our solution.

4.1 Users-Latency Scalability

We first measure the latency of our framework using motion to photon (MTP). This reflects the delay between receiving physical sensor data and updating the avatar's motion on the remote clients. This comprises the data collection, parsing, processing, and distribution of motion updates to other users for avatar rendering.

We vary the number of clients to monitor the impact on MTP delay. The experiment is carried out on eight computers, each with an i-7 CPU and 16GB-32GB RAM; 4 computers running with 1000M LAN, while other 4 computers over WiFi. In an 8-user experiment, each measurement lasts 2 minutes, creating 1.2K records for each avatar per run, and 9.6k records for each run. As a baseline for comparison, we use D-Joints/ F-Joints. We show results from D-Joints IK, Octree F-Joints, and Octree IK [8].

Figure 4 presents the average latency across all runs for each of the four techniques. We observe that our solution, Octree IK, consistently delivers the lowest delay, with an average of 67ms in an environment of 8 concurrent users. It attains an 55.7% improvement over D-Joints F-Joints; an 46.6% improvement of D-Joints IK and an 8.1% improvement of Octree F-Joints. Figure 4 also represents the MTP latency when the number of users increases. We observe that this substantially increases the delay for both D-Joints F-Joints and D-Joints IK. The key reason is that it is necessary to disseminate avatar motion updates to a larger number of users. In contrast, Octree F-Joints and Octree IK remain relatively insensitive to the

[8]D-Joints refers Direct-Joints, F-Joints to Full-Joints, and IK to Inverse Kinematic.

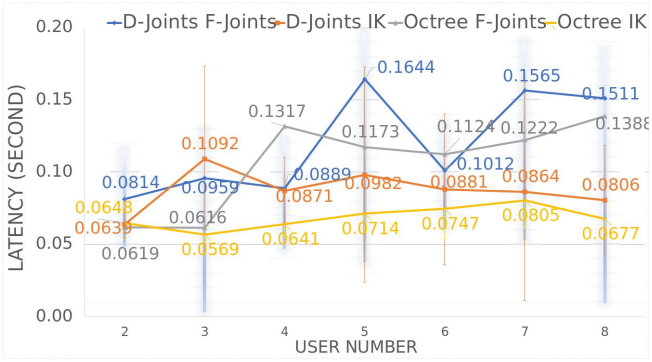


Figure 4: System latency between each users with multiple concurrent users, in different joint data streaming scheme.

Table 1: Data size and latency of three motion representations

	W/o IK		W/ IK	
	D Joints	Octree	D Joints	Octree
Joint Num	50	33	9	9
Data Size (bit)	11,200	5,296	2042	1312
Data Size (30fps)	42.06 KB	19.86 KB	7.657 KB	4.92 KB
Latency (8 users)	0.1511	0.1388	0.0806	0.0677
Latency std	0.2828	0.2045	0.0756	0.0192

increase in the number, as the data compression is higher due to the Octree data structure.

4.2 Broadcast Data Size

We next inspect the data volume transferred by our solution. This is required to exchange avatar motion updates between users.

Table 1 compares the per video frame size of the body joints Octree scheme with direct joints. The data size of the each frame is significantly smaller. The data size of direct joint streaming is 11,200 bit per frame for all joint positions and rotation (7 float data * 50 joints). The data size of octree IK scheme is the smallest, which including 9 octree node ID, 9 joint rotation data in quaternions, 2 hand fingers gripping scale in 8 bit scale. The data size of the Octree scheme is 5296 bit per frame, which includes the position and rotation of 33 body joints parts, and 2 hand fingers gripping scale in 8 bit scale.

4.3 Human Perception

Participants and Apparatus: To investigate the human-perceived impact, we perform a user study with participants, aged 19 to 36. We ask each participant about their technology literacy, which they report as: unfamiliar (5/22), slightly familiar (7/22), moderately familiar (6/22), and very familiar (4/22). We ask 22 participants (15 males and 7 females) to watch a video of a real person wearing the IMU suit and its corresponding avatar for the three use cases. We present three videos, representing key metaverse use cases:

(1) Handshake: This involves two people shaking hands (Figure 5a), exemplifying low-motion multi-user scenarios.

(2) Presentation: This involves a person giving a presentation (Figure 5b), exemplifying a single user scenario.
(3) Dance: This involves a person (Justin Beiber) giving a music concert (Figure 5c), exemplifying a high-motion single user scenario.

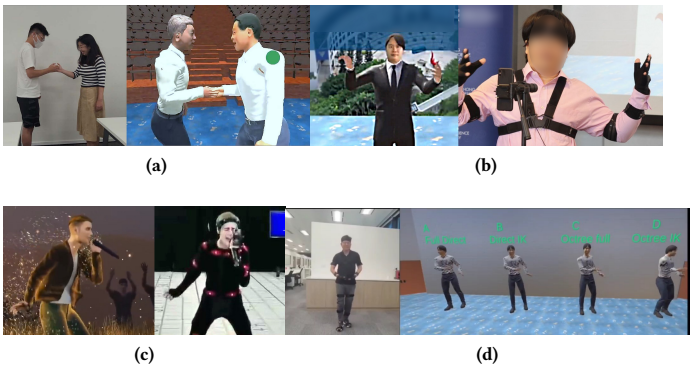


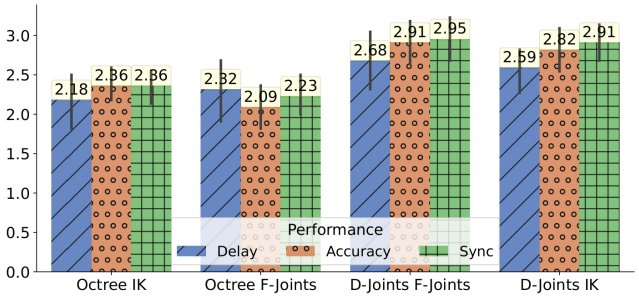
Figure 5: Three scenarios supported by the proposed framework. (a) Handshaking; (b) A presenter on a stage; (c) Dancing ©Justin Bieber; (d) Screen captures showing a video of dancing scenarios presented in the online survey.

After watching each video, using an online survey, the participants rate their experience. We follow a simplified version of the NASA TLX [15] survey. We asked the participants to rate the perceived delay between the physical motions and the avatar's motions on the video (1: noticeable, 2: somewhat noticeable, 3: acceptable and 4: unnoticeable). They rate the accuracy of the avatar recreation on a scale (1: inaccurate, 2: somewhat accurate, 3: accurate, 4: very accurate). They rate the synchronization between the physical body motion and the avatar's motion on a scale (1: no synchronization, 2: somewhat synchronized, 3: synchronized, and 4: fully synchronized). The participants also rate how much each solution is promising for human-avatar interaction in each use case on a scale (1: definitely not, 2: probably not, 3: not sure, 4: probably yes, and 5: definitely yes).

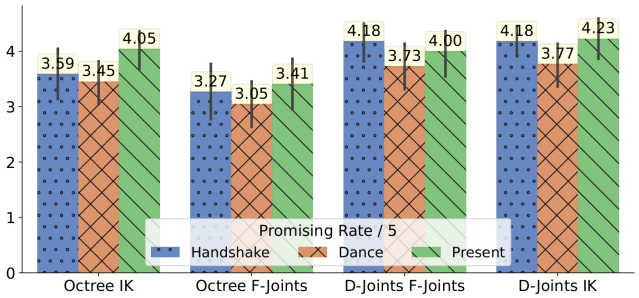
We test four solutions to enable human-avatar interaction: Direct Joints with full-body joints support (D-Joints F-Joints), Direct Joints with Inverse-Kinematic (D-Joints IK), Octree with full-body joints support (Octree F-Joints), and Octree with Inverse-Kinematic (Octree IK), as shown in Figure 5d.

Results: Figure 6 presents the results of the user study, with the 95% confidence intervals as error bars. Figure 6a shows that using D-Joints, F-Joints and D-Joints IK as solutions to recreate human-avatar interactions, leads to highest perceived delay, accuracy and synchronization between the real person's motion and the corresponding avatar's motion.

Using Octree F-Joints achieves slightly lower delay, the lowest accuracy and synchronization. Unsurprisingly, Octree IK achieves the best trade-off (i.e., lowest delay, and high accuracy and synchronization). Figure 6b illustrates that most participants find using D-Joints IK a more promising solution, followed by D-Joints F-Joints for the three use cases, daily etiquette (e.g., handshaking), art performance



(a) Perceived delay, accuracy and synchronization of the solutions.



(b) Perceived promise of each solution for each scenario.

Figure 6: User perception of (a) delay, accuracy and synchronization between the user motion and recreation of avatar's motion; and (b) how promising is each solution for each scenario.

(e.g., dancing). Octree with IK is more promising than D-Joints F-Joints for delivering speech and presentation, while Octree F-Joints appears to be the least promising in the three metaverse scenarios.

5 CONCLUSION

This paper serves as a groundwork to improve human-avatar interaction for large-scale use. It has proposed a simple technique to compress and transmit motion capture data. It relies on a lightweight Octree data structure to record and transmit motion to other users. By compressing such data, we can improve delivery efficiency in low-resource network environments. To evaluate its efficacy, we conducted a user study with 22 participants. Our user study shows that Octree with Inverse Kinematic achieves the best trade-off, achieving low delay and high accuracy. Our proposed solution delivers the lowest delay, with an average of 67ms in an environment of 8 concurrent users. It attains a 55.7% improvement over the prior techniques. For future work, we will examine the need for human-avatar interaction in more diversified scenarios.

ACKNOWLEDGMENTS

This research has been supported in part by the MetaHKUST project from the Hong Kong University of Science and Technology (Guangzhou), and 5GEAR and FIT projects from the Academy of Finland.

REFERENCES

[1] Ahmad Alhilal, Tristan Braud, Bo Han, and Pan Hui. 2022. Nebula: Reliable Low-Latency Video Transmission for Mobile Cloud Gaming. In *Proceedings of the ACM Web Conference 2022* (Virtual Event, Lyon, France) *(WWW '22)*. Association for Computing Machinery, New York, NY, USA, 3407–3417. https://doi.org/10.1145/3485447.3512276

[2] Andreas Aristidou and Joan Lasenby. 2011. FABRIK: A fast, iterative solver for the Inverse Kinematics problem. *Graphical Models* 73, 5 (2011), 243–260.

[3] Norman I. Badler, Michael J. Hollick, and John P. Granieri. 1993. Real-Time Control of a Virtual Human Using Minimal Sensors. *Presence: Teleoper. Virtual Environ.* 2, 1 (jan 1993), 82–86. https://doi.org/10.1162/pres.1993.2.1.82

[4] Ricardo R Barioni, Lucas Figueiredo, Kelvin Cunha, and Veronica Teichrieb. 2018. Human pose tracking from rgb inputs. In *2018 20th Symposium on Virtual and Augmented Reality (SVR)*. IEEE, 176–182.

[5] Valentin Bazarevsky, Ivan Grishchenko, Karthik Raveendran, Tyler Zhu, Fan Zhang, and Matthias Grundmann. 2020. Blazepose: On-device real-time body pose tracking. *arXiv preprint arXiv:2006.10204* (2020).

[6] Carlos Bermejo, Lik Hang Lee, Paul Chojecki, David Przewozny, and Pan Hui. 2021. Exploring Button Designs for Mid-Air Interaction in Virtual Reality: A Hexa-Metric Evaluation of Key Representations and Multi-Modal Cues. *Proc. ACM Hum.-Comput. Interact.* 5, EICS, Article 194 (may 2021), 26 pages. https://doi.org/10.1145/3457141

[7] Heike Brock, Shigeaki Nishina, and Kazuhiro Nakadai. 2018. To animate or anime-te? investigating sign avatar comprehensibility. In *Proceedings of the 18th International Conference on Intelligent Virtual Agents*. 331–332.

[8] Gordon Brown and Michael Prilla. 2020. The effects of consultant avatar size and dynamics on customer trust in online consultations. In *Proceedings of the Conference on Mensch und Computer*. 239–249.

[9] Changyeol Choi, Joohee Jun, Jiwoong Heo, and Kwanguk (Kenny) Kim. 2019. Effects of Virtual-Avatar Motion-Synchrony Levels on Full-Body Interaction. In *Proceedings of the 34th ACM/SIGAPP Symposium on Applied Computing* (Limassol, Cyprus) *(SAC '19)*. Association for Computing Machinery, New York, NY, USA, 701–708. https://doi.org/10.1145/3297280.3297346

[10] Alanah Davis, John Murphy, Dawn Owens, Deepak Khazanchi, and Ilze Zigurs. 2009. Avatars, people, and virtual worlds: Foundations for research in metaverses. *Journal of the Association for Information Systems* 10, 2 (2009), 1.

[11] Yogesh K Dwivedi, Laurie Hughes, Abdullah M Baabdullah, Samuel Ribeiro-Navarrete, Mihalis Giannakis, Mutaz M Al-Debei, Denis Dennehy, Bhimaraya Metri, Dimitrios Buhalis, Christy MK Cheung, et al. 2022. Metaverse beyond the hype: Multidisciplinary perspectives on emerging challenges, opportunities, and agenda for research, practice and policy. *International Journal of Information Management* 66 (2022), 102542.

[12] Ilya Eckstein, Mathieu Desbrun, and C-C Jay Kuo. 2006. Compression of time varying isosurfaces. In *Proceedings of Graphics Interface 2006*. Citeseer, 99–105.

[13] Abu Ilius Faisal, Sumit Majumder, Tapas Mondal, David Cowan, Sasan Naseh, and M Jamal Deen. 2019. Monitoring methods of human body joints: State-of-the-art and research challenges. *Sensors* 19, 11 (2019), 2629.

[14] Guo Freeman and Divine Maloney. 2021. Body, avatar, and me: The presentation and perception of self in social virtual reality. *Proceedings of the ACM on Human-Computer Interaction* 4, CSCW3 (2021), 1–27.

[15] Sandra G Hart. 2006. NASA-task load index (NASA-TLX); 20 years later. In *Proceedings of the human factors and ergonomics society annual meeting*, Vol. 50. Sage publications Sage CA: Los Angeles, CA, 904–908.

[16] Thomas Hilfert and Markus König. 2016. Low-cost virtual reality environment for engineering and construction. *Visualization in Engineering* 4, 1 (2016), 1–18.

[17] Fan Jiang, Xubo Yang, and Lele Feng. 2016. Real-time full-body motion reconstruction and recognition for off-the-shelf VR devices. In *Proceedings of the 15th ACM SIGGRAPH Conference on Virtual-Reality Continuum and Its Applications in Industry-Volume 1*. 309–318.

[18] Julius Kammerl, Nico Blodow, Radu Bogdan Rusu, Suat Gedikli, Michael Beetz, and Eckehard Steinbach. 2012. Real-time compression of point cloud streams. In *2012 IEEE International Conference on Robotics and Automation*. 778–785. https://doi.org/10.1109/ICRA.2012.6224647

[19] Shunichi Kasahara, Keina Konno, Richi Owaki, Tsubasa Nishi, Akiko Takeshita, Takayuki Ito, Shoko Kasuga, and Junichi Ushiba. 2017. Malleable embodiment: changing sense of embodiment by spatial-temporal deformation of virtual human body. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems*. 6438–6448.

[20] Jooyoung Kim. 2021. Advertising in the Metaverse: Research agenda. *Journal of Interactive Advertising* 21, 3 (2021), 141–144.

[21] Martin Kocur, Sarah Graf, and Valentin Schwind. 2020. The impact of missing fingers in virtual reality. In *26th ACM Symposium on Virtual Reality Software and Technology*. 1–5.

[22] Sven Kreiss, Lorenzo Bertoni, and Alexandre Alahi. 2019. Pifpaf: Composite fields for human pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 11977–11986.

[23] Marc Erich Latoschik, Daniel Roth, Dominik Gall, Jascha Achenbach, Thomas Waltemate, and Mario Botsch. 2017. The effect of avatar realism in immersive social virtual realities. In *Proceedings of the 23rd ACM symposium on virtual reality software and technology*. 1–10.

[24] Lik-Hang Lee, Tristan Braud, Kit-Yung Lam, Yui-Pan Yau, and Pan Hui. 2020. From seen to unseen: Designing keyboard-less interfaces for text entry on the constrained screen real estate of Augmented Reality headsets. *Pervasive Mob. Comput.* 64 (2020), 101148.

[25] Lik-Hang Lee, Tristan Braud, Pengyuan Zhou, Lin Wang, Dianlei Xu, Zijun Lin, Abhishek Kumar, Carlos Bermejo, and Pan Hui. 2021. All one needs to know about metaverse: A complete survey on technological singularity, virtual ecosystem, and research agenda. *arXiv preprint arXiv:2110.05352* (2021).

[26] Lik-Hang Lee and Pan Hui. 2018. Interaction Methods for Smart Glasses: A Survey. *IEEE Access* 6 (2018), 28712–28732. https://doi.org/10.1109/ACCESS.2018.2831081

[27] Jing Li, Huayi Wu, Chaowei Yang, David W Wong, and Jibo Xie. 2011. Visualizing dynamic geosciences phenomena using an octree-based view-dependent LOD strategy within virtual globes. *Computers & geosciences* 37, 9 (2011), 1295–1302.

[28] Jean-Luc Lugrin, Ivan Polyschev, Daniel Roth, and Marc Erich Latoschik. 2016. Avatar anthropomorphism and acrophobia. In *Proceedings of the 22nd ACM Conference on Virtual Reality Software and Technology*. 315–316.

[29] Rufael Mekuria, Kees Blom, and Pablo Cesar. 2016. Design, implementation, and evaluation of a point cloud codec for tele-immersive video. *IEEE Transactions on Circuits and Systems for Video Technology* 27, 4 (2016), 828–842.

[30] Dooley Murphy. 2017. Building a hybrid virtual agent for testing user empathy and arousal in response to avatar (micro-) expressions. In *Proceedings of the 23rd ACM Symposium on Virtual Reality Software and Technology*. 1–2.

[31] Rabindra Ratan and Béatrice S Hasler. 2014. Playing well with virtual classmates: relating avatar design to group satisfaction. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*. 564–573.

[32] Ke Sun, Bin Xiao, Dong Liu, and Jingdong Wang. 2019. Deep high-resolution representation learning for human pose estimation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*. 5693–5703.

[33] Xiaozhou Wei, Lijun Yin, Zhiwei Zhu, and Qiang Ji. 2004. Avatar-mediated face tracking and lip reading for human computer interaction. In *Proceedings of the 12th annual ACM international conference on Multimedia*. 500–503.

[34] Sijie Yan, Yuanjun Xiong, and Dahua Lin. 2018. Spatial Temporal Graph Convolutional Networks for Skeleton-Based Action Recognition. In *AAAI*.

[35] Yui-Pan Yau, Lik Hang Lee, Zheng Li, Tristan Braud, Yi-Hsuan Ho, and Pan Hui. 2020. How Subtle Can It Get? A Trimodal Study of Ring-Sized Interfaces for One-Handed Drone Control. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 4, 2, Article 63 (jun 2020), 29 pages. https://doi.org/10.1145/3397319

[36] Qilong Yuan, I-Ming Chen, and Ang Wei Sin. 2013. Method to calibrate the skeleton model using orientation sensors. In *2013 IEEE International Conference on Robotics and Automation*. IEEE, 5297–5302.

[37] Jinghua Zhang, Jinsheng Xu, and Huiming Yu. 2007. Octree-based 3D animation compression with motion vector sharing. In *Fourth International Conference on Information Technology (ITNG'07)*. IEEE, 202–207.

[38] Jinghua Zhang, Jinsheng Xu, and Huiming Yu. 2007. Octree-Based 3D Animation Compression with Motion Vector Sharing. In *Fourth International Conference on Information Technology (ITNG'07)*. 202–207. https://doi.org/10.1109/ITNG.2007.138

[39] Tobias Zimmermann, Bertram Taetz, and Gabriele Bleser. 2018. IMU-to-segment assignment and orientation alignment for the lower body using deep learning. *Sensors* 18, 1 (2018), 302.