Received 11 May 2023, accepted 14 July 2023, date of publication 27 July 2023, date of current version 2 August 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3299323

RESEARCH ARTICLE

Geometry-Incorporated Posing of a Full-Body Avatar From Sparse Trackers

TARAVAT ANVARI AND KYOUNGJU PARK, (Member, IEEE)
School of Computer Science and Engineering, Chung-Ang University, Seoul 06974, South Korea

Corresponding author: Kyoungju Park (kjpark@cau.ac.kr)

This work was supported in part by the Mid-Career Research Program through the NRF Grant funded by the Korea MEST under Grant NRF-2021R1A2C1014210.

ABSTRACT Accurately rendering a user's full body in a virtual environment is crucial for embodied mixed reality (MR) experiences. Conventional MR systems provide sparse trackers such as a headset and two hand-held controllers. Recent studies have intensively investigated learning methods to regress untracked joints from sparse trackers and have produced plausible poses in real time for MR applications. However, most studies have assumed that they either know the position of the root joint or constrain it, yielding stiff pelvis motions. This paper presents the first geometry-incorporated learning method to generate the position and rotation of all joints, including the root joint, from the head and hands information for a wide range of motions. We split the problem into identifying a reference frame and a pose inference with respect to the new reference frame. Our method defines an avatar frame by setting a non-joint as the origin and transforms joint data in a world coordinate system into the avatar coordinate system. Our learning builds on a propagating long short-term memory (LSTM) network exploiting prior knowledge of the kinematic chains and the previous time domain. The learned joints are transformed back to obtain the positions with respect to the world frame. In our experiments, our method achieves competitive accuracy and robustness with the state-of-the-art speed of approximately 130 fps on motion capture datasets and the wild tracking data obtained from commercial MR devices. Our experiments confirm that the proposed method is practically applicable to MR systems.

INDEX TERMS 3D human pose estimation, avatar, mixed reality, virtual reality.

I. INTRODUCTION

The presence of a virtual body is quite compelling for immersive experiences, particularly when the body moves like one's own body [1]. Today's mixed reality(MR) system provides three trackers: a head-mounted device (HMD) and two hand-held controllers. It is possible to visualize a user's head and hands in a virtual world and to interact with virtual objects. Inverse kinematics(IK) methods generate plausible upper-body motions from three trackers and enable gesture interactions with the visualized upper-body of avatars. Users can see their own upper bodies during virtual reality (VR) interaction and others' upper bodies in augmented reality (AR) collaborative tasks. These virtual upper bodies bring a sense of embodiment and lead to a sense of presence [2].

The animation of virtual full bodies enhances embodiment, immersion, and presence further than that of virtual upper bodies. However, from the limited number of MR system trackers, creating a virtual full-body is difficult. Users' motions in MR range from upper-body gesture interaction to full-body walking, running, sitting, jumping, and so on. The problem of generating a full-body avatar from these users' motions requires solving the pose estimation problem and the positioning and orientation problems of an avatar. The virtual full-body generation of a user with three trackers is a vastly underdetermined problem.

For the last couple of years, a significant amount of work has addressed generating full-body poses in real-time from sparse trackers. Previous studies used six sensors with a deep neural network [3], [4], [5], [6], [7], [8], [9], four sensors by attaching an additional sensor to the pelvis joint [10], and three sensors of consumer-grade devices [11], [12], [13], [14], [15], [16]. Previous work using three sensors has constrained the motion using a pre-generated template motion repository [11] or implicitly assumed the location of the pelvis by encoding all joints relative to the pelvis

The associate editor coordinating the review of this manuscript and approving it for publication was Xiaogang Jin.

[12], [13]. Jiang et al.'s recent work [14] tackles the global and local pose together and estimates a full-body pose from three sensors with promising results. However, because their method obtains the position of the pelvis using forward kinematics, the location of the pelvis joint is constrained by the head. Physics-based methods with reinforcement learning produce high-quality, realistic global and local poses [15], [16]. However, these reinforcement learning methods are not practical for VR in their current form due to the relatively slow execution time and occasional failure to track abrupt and non-trained complex motions. Existing methods for the full-body pose from three sparse trackers have limitations, such as using additional trackers, assuming that the root joint position is known, and failing to track a wide range of motions needs to be faster and more flexible for the root joint.

We propose a new geometry-incorporated learning approach that splits our problem into finding a reference frame of an avatar and estimating a pose with respect to the avatar frame. We refer to our approach as GeoPose. GeoPose defines a reference frame of a human body from three trackers and represents the data with respect to the defined avatar frame. Then, a pose regression network finds mapping in an avatar coordinate system. Our modified propagating long short-term memory (p-LSTM) network embeds the joint connectivity into the deep learning structure. Our p-LSTM first determines the position and orientation of the pelvis joint in an avatar coordinate system, which is the root joint of an articulated body. Subsequently, p-LSTM regresses the connected joints sequentially. Our network's output is then transformed back into a world coordinate system. Our GeoPose enables inferring all joints, including the pelvis joint, with respect to an avatar frame and locating and orienting an avatar in a world coordinate system.

By representing the joint with respect to an avatar frame, we obtain the following benefits. 1) We decouple global motion and local pose geometrically, 2) Position and orientation of *three trackers* in MR are represented with respect to the avatar frame instead of the anonymous world frame of the applications, 3) Position and orientation of *a root joint* in motion capture data are represented with respect to the avatar frame instead of the world frame at the set-up time, 4) Positions of *the articulated joints* are represented with respect to the avatar frame instead of the world frame of the motion capture system, 5) The pelvis joint has no more near-zero values by defining the origin of the avatar frame on the ground instead of the pelvis location, and this avoids a many-to-one mapping problem from leaf node joints to a root joint.

To the best of our knowledge, GeoPose is the first method for inferring six degrees of freedom (DoFs) of the pelvis joint from three trackers. The geometry-incorporated method ensures robust global localization and orientation of the avatar. Joint data representations with respect to an avatar frame increase the accuracy of pose generation across a wide range of poses from the three trackers. Experiments validate the effectiveness of our method on a motion capture dataset and actual VR data. Actual VR experiments include bending the upper body, sitting, squatting, jumping, etc.. GeoPose achieves competitive accuracy, high speed, and robustness compared to state-of-the-art methods.

II. RELATED WORKS

Human motion capture plays an important role in the gaming, film, and MR industries. Commercial motion capture solutions such as Vicon [17] have captured high-quality human performance. These high-end MoCap systems use tens of optical markers or inertial measurement units (IMUs) to capture accurate motion. Such systems are complicated to set up, expensive, and inadequate for everyday VR and AR applications. Animating complex full-body avatars based on sparse input from conventional MR platforms has become increasingly demanding. Numerous studies have been conducted on this topic. Since our method only requires sparse trackers as inputs, we do not discuss image-based approaches. We review neural networks for full-body motion generation using sparse sensor trackers.

A. FULL BODY POSE FROM SPARSE TRACKERS

Many state-of-the-art approaches have achieved full-body poses using six body-worn sensors on the user's head, limbs, and waist [3], [4], [5], [6], [7], [8]. Marcard et al. [5] present an offline method for human motion capture. Huang et al. [6] propose the first deep learning method, which uses a bidirectional recurrent neural network (RNN) to estimate the human pose in real-time. TransPose [7] estimates global translations and body poses from six IMUs at 90fps with state-of-the-art accuracy and later combines it with physics-based motion optimization for joint torque and ground reaction [8]. EM-Pose [9] proposes an approach based on electromagnetic (EM) field sensing.

Research on estimating full-body human motion using even fewer tracking signals often requires a sensor on the waist for flexible lower-body motions. Wouda et al. [18] employ an LSTM-based model to reconstruct the full body from five sensors mounted on a user's limbs and the waist. LoBSTr [10] shows a gated recurrent unit (GRU) model to predict lower-body motions from four upper-body VR sensors mounted on the head, hands, and waist and to compute the upper body using an IK solver. They state that three sensors are difficult to achieve a wide range of motions.

CoolMoves [11] was the first to use inputs from only the headset and hand-held controllers. Since they use a template motion repository to synthesize human motion, match similar motions, and interpolate between them, they limit the range of synthesized motions. Dittadi et al. [12] use a variational autoencoder (VAE) framework to compress the head and hands inputs to a latent space, generating a full-body pose by sampling from that latent space. However, the authors assume that the avatar root is at the origin and the representation of the other joints is relative to the root. Therefore, they implicitly use the pelvis as a fourth input location, which means that they solve the problem from four locations and three rotations of the sensors.

FLAG [13] proposes an approach based on conditional normalizing flows for sparse inputs to compute the exact pose likelihood and outperforms state-of-the-art methods on the AMASS dataset, leading to a deficient error. However, the authors also assume that the avatar root is at the origin and has the same problem as the VAE framework. Therefore, a VAE framework and FLAG are only practical for actual data when users stay at a specific location. AvatarPoser [14] employs a Transformer to learn pose features, decouples the global motion from the learned pose features, and optimizes the learned features using an IK. They achieve accurate results at a competitive speed. However, since they use forward kinematics to find the position of the pelvis, the pelvis location is constrained to the head location. As a result, the pelvis position is stiff so that the height of the pelvis joint becomes approximately half the head height, and the in-plane pelvis position moves accordingly to the in-plane head position.

Neural3Points [15] combines a data-driven method with physics-based simulation and uses deep reinforcement learning to reconstruct realistic full-body movements of the user using three VR trackers. QuestSim [16] incorporates an off-the-shelf physics simulator into a reinforcement-learning pipeline to constrain the solution space to physically correct poses. QuestSim produces accurate and plausible simulations by mitigating artifacts, such as jitter, foot skating, and unstable contacts. However, these reinforcement learning approaches have limitations with respect to unexpected user motion in MR interactions. Moreover, the delay is about 100∼160 ms, making it impossible to meet real-time VR requirements.

B. LSTM

Recent advances in neural networks have demonstrated their potential for effectively handling extensive, high-dimensional datasets. Once trained, these models have a low memory footprint and are executed quickly [19]. Recurrent Neural Networks (RNN) and their derivatives, such as LSTM [20] and Gated Recurrent Units (GRU) [21], have been successful in sequence prediction tasks [22] and tasks with densely connected data [23]. For example, Lin et al. [24] employed RNN, LSTM, and GRU models to learn joint points over time series, as RNNs are well suited for handling highly related problems. These studies show that RNN models can learn the dependencies between spatially correlated data, such as the rotation of human joints. Lee et al. [25] presented p-LSTMs by connecting several LSTM networks in series to reflect body part-based structural connectivity as prior knowledge.

III. PROBLEM STATEMENT

MR systems provide the positions and orientations of the HMDs and hand-held controllers with respect to the world frame of MR applications. Let $\mathbf{p}_w(s,t)$, a 3 by 1 vector, and $\mathbf{R}_w(s,t)$, a 3 by 3 matrix, denote the position and rotation matrix of an s tracker at a t timeframe using the world coordinate system. Since each dataset has a different order of rotations for Euler angles, we rather employ a rotation matrix. From these sparse tracker data, GeoPose finds the reference frame of an avatar, an avatar frame, and regresses the position of the articulated joints with respect to the avatar frame.

An avatar frame is defined with the desired location and the three basis vectors of an avatar instance. Desired location and basis vectors of the avatar instance contain the relationship between the avatar instance and world frame at the t timeframe. From the given $\mathbf{p}_w(s,t)$ and $\mathbf{R}_w(s,t)$, this paper obtains the $\mathbf{p}_a(s,t)$ and $\mathbf{R}_a(s,t)$, the position and rotation matrix of an s sensor at the t timeframe with respect to the avatar frame respectively. This relationship is represented as follows:

$$\mathbf{p}_a(s,t) = \mathbf{M}(t)(\mathbf{p}_w(s,t) - \mathbf{a}_w(t)). \tag{1}$$

$$\mathbf{R}_a(s,t) = \mathbf{M}(t)\mathbf{R}_w(s,t). \tag{2}$$

where $\mathbf{M}(t)$ is a 3 by 3 rotation matrix and $\mathbf{a}_w(t)$ is a 3 by 1 translation vector.

Then, this paper achieves the positions and rotations of the articulated joints for a full body by learning a mapping function $f : \mathbb{R}^{12s_n} \rightarrow \mathbb{R}^{12j_n}$ through the following equation,

$$\{\mathbf{p}_a(s,t), \mathbf{R}_a(s,t)\}_{1:t_n}^{1:j_n} = f(\{\mathbf{p}_a(s,t), \mathbf{R}_a(s,t)\})_{1:t_n}^{1:s_n}. \tag{3}$$

where s_n is the number of joints tracked by the MR system, j_n denotes the number of joints of the full-body avatar, and t_n matches the number of observed MR frames considered from the past. Specifically, s_n is 3 for a headset and both hand-held controllers, and j_n is 21.

We find the transformations in Eq. 1 and Eq. 2 by defining and updating the avatar reference frames, and Sect. IV describes this in detail. With respect to our avatar frame, the data representations are posturally meaningful, so the mapping in Eq. 3 is feasible. Sect. V introduces the network architecture to infer the position of articulated joints from the headset and both hand-held controllers. To regress the posture of an avatar from sparse trackers in MR, a vast number of 3D ground-truth pose data from a motion capture system are required. The output of the network is transformed back to achieve the positions in a world coordinate system.

IV. MODEL GEOMETRY

A. AVATAR'S REFERENCE FRAME

Using the reference frame of an avatar, GeoPose enables distinguishing global translation and rotation of an avatar instance and representing the articulated joints with respect to the avatar frame. Rather than typical reference frames for human bodies with a pelvis joint as an origin, our avatar frame has an origin on the ground to avoid the near-zero values associated with the position of the pelvis joint, and the basis vectors of our avatar frame align to the body's side, upward, and forward directions. Fig. 1 shows the avatar frames in MR and motion capture systems, updated according to a human body's motion. Therefore the joints' representations deliver posturally the same meaning at any time in MR and motion capture systems. By expressing the position and orientation of the joints using avatar coordinates, the representations of joints in motion capture systems and MR systems keep correspondence.

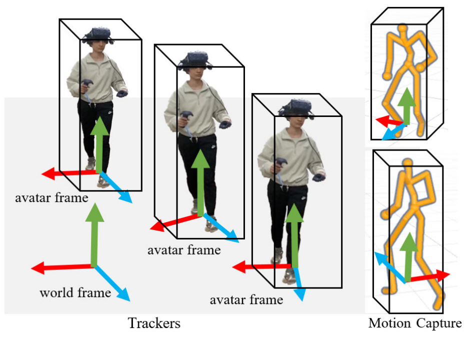

FIGURE 1. Avatar frames in MR (left) and motion capture (right). The origin is on the floor, and the x,y, and z-axes are unit normal vectors along the side, upward, and forward directions drawn in red, green, and blue arrows.

Given a set of tracker positions and orientations, $\{\mathbf{p}_w(s,t), \mathbf{R}_w(s,t)\}^{s=1:3}$ in the Cartesian coordinate system, we create a new avatar frame that has the origin on the nearest floor and the axes along the side, upward, and forward directions of the avatar. We obtain the origin, $\mathbf{a}_w(t)$, by projecting the center of a bounding box of three trackers at t timeframe, $\mathbf{c}_w(t)$, onto the ground plane,

$$\mathbf{a}_w(t) = \mathbf{c}_w(t) - \frac{(\mathbf{c}_w(t)\cdot \mathbf{y}_w)}{\|\mathbf{y}_w\|\|\mathbf{y}_w\|}\mathbf{y}_w. \tag{4}$$

where the normal vector of a ground plane is the $\mathbf{y}_w$, the basis vector along a y_w-axis in a world coordinate system. Next, we want to specify the coordinate system of an avatar with $\mathbf{x}_a$, $\mathbf{y}_a$, and $\mathbf{z}_a$ basis vectors that represent the side, upward, and forward directions of the body. We set a ground plane normal $\mathbf{y}_w$ as one coordinate direction,

$$\mathbf{y}_a = \mathbf{y}_w. \tag{5}$$

and find two other orthogonal directions that we call $\mathbf{x}_a$ and $\mathbf{z}_a$. We obtain the candidate direction for the avatar's viewing, $\tilde{\mathbf{z}}_a$, from the rotation matrix of the headset sensor, $\mathbf{R}_w(1,t)$, as follows:

$$\tilde{\mathbf{z}}_a = \mathbf{R}_w(1,t)\mathbf{z}_w.$$

The rotation matrix $\mathbf{R}_w(s,t)$ orients the s sensors at t timeframe and transforms the x_w, y_w, z_w-axes of the world coordinate system into the side, upward, and viewing directions of the sensors. The side direction, $\mathbf{x}_a$, must be orthogonal to both $\mathbf{x}_a$ and $\tilde{\mathbf{z}}_a$ and is found by taking the cross-product,

$$\mathbf{x}_a = \mathbf{y}_a \times \tilde{\mathbf{z}}_a/\|\tilde{\mathbf{z}}_a\|. \tag{6}$$

The third orthogonal direction is

$$\mathbf{z}_a = \frac{\mathbf{x}_a \times \mathbf{y}_a}{\|\mathbf{x}_a\|\|\mathbf{y}_a\|}. \tag{7}$$

B. COORDINATE CONVERSION FOR TRACKERS

MR systems provide $\mathbf{p}_w(s,t)$ and $\mathbf{R}_w(s,t)$ in a world coordinate system. We transform a given data in a world coordinate system into an avatar coordinate system. Transformations begin with finding the avatar frame as described in Sect. IV-A. A point in the avatar coordinates is rotated and translated to a point in the world coordinates,

$$\mathbf{p}_w(s,t) = \mathbf{A}(t)\mathbf{p}_a(s,t) + \mathbf{a}_w(t). \tag{8}$$

$$\mathbf{R}_w(s,t) = \mathbf{A}(t)\mathbf{R}_a(s,t). \tag{9}$$

where $\mathbf{A}(t)$ is a rotation matrix whose columns are the basis vectors of the avatar frame. That is,

$$\mathbf{M}(t) = \mathbf{A}(t)^{-1} = \begin{bmatrix} \mathbf{x}_a^T \\ \mathbf{y}_a^T \\ \mathbf{z}_a^T \end{bmatrix}. \tag{10}$$

We want to go in the opposite direction to obtain the representation of vectors in the world coordinate system in the avatar coordinate system. We want $\mathbf{A}(t)^{-1}$, and the desired matrix $\mathbf{M}(t)$ is

$$\mathbf{M}(t) = \mathbf{A}(t)^{-1} = \begin{bmatrix} \mathbf{x}_a^T \\ \mathbf{y}_a^T \\ \mathbf{z}_a^T \end{bmatrix}. \tag{11}$$

Since a rotation matrix is an orthonomal matrix in that the columns are orthogonal to each other and unit vectors, the inverse of a rotation matrix is a transpose matrix. We transform $\mathbf{p}_w(s,t)$ and $\mathbf{R}_w(s,t)$ and achieve the representations in the avatar frame as described in Eq. 1 and Eq. 2.

C. COORDINATE CONVERSION FOR MOTION CAPTURE

Since a typical motion capture system sets a pelvis joint as an origin and the initial frame stays the same even after capturing subjects' moving around in space, the representations of the positions and rotations at t timeframe are with respect to the frame defined at an initial timeframe. This fixed frame yields the undesired effects that the pelvis joint has rotation angles about the unknown and meaningless axes. To avoid this, we transform the motion capture data into the defined avatar coordinate system so that the axes are posturally meaningful.

To define an avatar frame, we need to use the positions and orientations of the head and hand controller. The representations of the head and hand joints in motion capture systems are with respect to the parent joints, and those in MR systems are with respect to the reference frame. In order to make the same representations, we add imaginary trackers to motion capture data for the headset and two hand-held controllers. These imaginary trackers represent their position and rotations with respect to the reference frame instead of the parent joint. Using these imaginary trackers, we define an avatar frame.

Our transformations for motion capture data are; 1) to create the imaginary trackers 2) to define the avatar frame, and 3) to transform data representations in an avatar coordinate system. At first, we create the imaginary trackers whose representations correspond to the MR tracker inputs with respect to a world frame, $\mathbf{p}_w(s,t), \mathbf{R}_w(s,t)$ for $s = 1, 2, 3$. In a world frame, we use the center point of the left and right eye joints for a headset $s = 1$, the left-hand joint for a left controller $s = 2$, and the right-hand joint for a right controller $s = 3$. By adding the offset ε along the principal direction, these computed data match the headset and hand-held controllers in the physical world. Next, we define the avatar frame as in Sect. IV-A. Then, we transform into an avatar coordinate system to achieve $\mathbf{p}_a(s, t), \mathbf{R}_a(s, t)$ for $s = 4 \cdots 21$ where the number of joints to represent the full body is 21. We repeat these coordinate transformations for each timeframe t of motion capture data.

Transformed position and rotation values have representations in an avatar coordinate system. After our transformations, the rotation angles of the imaginary trackers at $s = 1, 2, 3$ and the pelvis joint at $s = 4$ are rotation about the axes from the avatar frame, while the rotation angles of the captured joints at $s = 5 \ldots 21$ are rotation about the axes from the parent joints. The avatar frame is updated at every timeframe so that the transformed motion capture data is robust to the user's global motion.

V. NETWORK ARCHITECTURE FOR POSING

A. INPUT AND OUTPUT REPRESENTATION

For a mapping f defined in (3), the input is the position $\mathbf{p}_a(s)$ and rotation matrix $\mathbf{R}_a(s)$ of $s = 1, 2, 3$ in an avatar coordinate system that are acquired as (10) and (11) from the head and two hand-held controllers. Due to the discontinuity of the axis-angle representation [26], we use a rotation matrix, discard the third row, and end up with a continuous 6D representation, $\theta_a(s)$, 6 by 1 vector. The final input representation is a concatenated vector of position and rotation from all given sparse inputs, $\mathbf{X} \in \mathbb{R}^{27}$, as follows:

$$\mathbf{X} = \left[\mathbf{p}_a(1)^T, \theta_a(1)^T, \cdots, \mathbf{p}_a(3)^T, \theta_a(3)^T\right]. \tag{12}$$

Note that $\mathbf{p}_a$ and θ_a in $\mathbf{X}$ are the global displacements and rotations with respect to the avatar frame.

The output is the global translation and rotation of the pelvis joint, $s = 4$, with respect to the avatar frame, and the local rotation of the articulated joints, $s = 5, \cdots, 21$ with respect to their parent joints. Our output representation is a concatenated vector from the pelvis and the other 17 joints, $\mathbf{Y} \in \mathbb{R}^{111}$, as follows:

$$\mathbf{Y} = \left[\mathbf{p}_a(4)^T, \theta_a(4)^T, \theta_a(5)^T, \cdots, \theta_a(21)^T\right]. \tag{13}$$

The network's output defines the pose of a human body with respect to the avatar frame, and is rotated and translated to find the pose with respect to the world frame as described in Eq. 8 and Eq. 9

B. NETWORK ARCHITECTURE

Our network exploits and modifies the p-LSTMs [25] that incorporates prior knowledge of joint connectivities into the learning network and considers spatial correlation as well as temporal correlation. The basic architecture of a p-LSTM network consists of a series of LSTM cells. Each LSTM cell takes as input a pose vector, the LSTM cell then updates its internal state based on the input pose vector and the previous state of the network, and the output of the LSTM cell is a hidden state vector that is passed to the next LSTM cell in the sequence. Our key modification is to set a sequential order of LSTM cells to achieve the root joint first, the upper-body, and then the lower body.

As depicted in Fig. 2 (c), a series of nine pose layers alongside 10 pose cue layers are connected to find the pose following the kinematics tree of 21 joints. Fig. 2 (d) shows the p-LSTM cell, which consists of one LSTM network and one pose layer. Specifically, the first p-LSTM builds the root joint's position and rotations from three sparse trackers. The inferred root joint is merged into the input pose in the pose layer of the first p-LSTM. The merged information is propagated along the next sequentially connected p-LSTM network. The entire pose is constructed in the order shown in Fig.2 (e). Two types of pose cues can be created depending on how outputs are merged: 1) Concatenation method, which concatenates the output of one p-LSTM to its input (from first p-LSTM to 5^{th}, and from 6^{th} p-LSTM to 10^{th} pose layer). 2) Elimination method, which deletes the rotations of the unnecessary joints (from 5^{th} p-LSTM to 6^{th}). The propagated pose cue is regressed to the full-body pose via FC. The output of the pose network, Y, is changed to the world coordinates to achieve the final output pose.

C. LOSS FUNCTION

We minimize a loss function of the position and rotation losses from the p-LSTMs. The loss function of the first p-LSTM model, L_{pelvis} is

$$L_{pelvis} = \lambda_1(\hat{\mathbf{p}}_a(4) - \mathbf{p}_a(4))^2 + \lambda_2(\hat{\theta}_a(4) - \theta_a(4))^2. \tag{14}$$

where $\hat{\mathbf{p}}$ and $\hat{\theta}$ denote the ground truth of the position and rotation vectors. Eq. 14 shows the loss of the first p-LSTM model, which predicts the global positions and rotations of the pelvis joint. The total loss of k propagated LSTMs is,

$$L = \lambda_3 L_{pelvis} + \lambda_4 \sum_k \frac{1}{j_n - 4} \sum_{j=5}^{j_n} (\hat{\theta}_a(j) - \theta_a(j))^2. \tag{15}$$

In our study, the hyperparameters are $\lambda_1 = 1$, $\lambda_2 = 1$, $\lambda_3 = 0.1$, and $\lambda_4 = 0.01$.

VI. EXPERIMENTS

In this section, we first explain the implementation details in Sect. VI-A, and introduce metrics used in our experiments in Sect. VI-B. Using metrics, we compare our method's accuracy with previous methods qualitatively and quantitatively in Sect. VI-C. We compare our method's running time with previous methods in Sect. VI-D. We demonstrate our method's robustness for an application in VR systems in Sect. VI-E. We end with the limitations in Sect. VI-F.

A. IMPLEMENTATION DETAILS

We use two public datasets for training and testing, namely CMU [27] and HDM05 [28] datasets, which are two widely used motion capture datasets for 3D human motion reconstruction. We split each dataset into 90% (831,234 frames) and 10% (92,359 frames) of random training and testing sets. We unified the frame rate of these two datasets to 60 Hz.

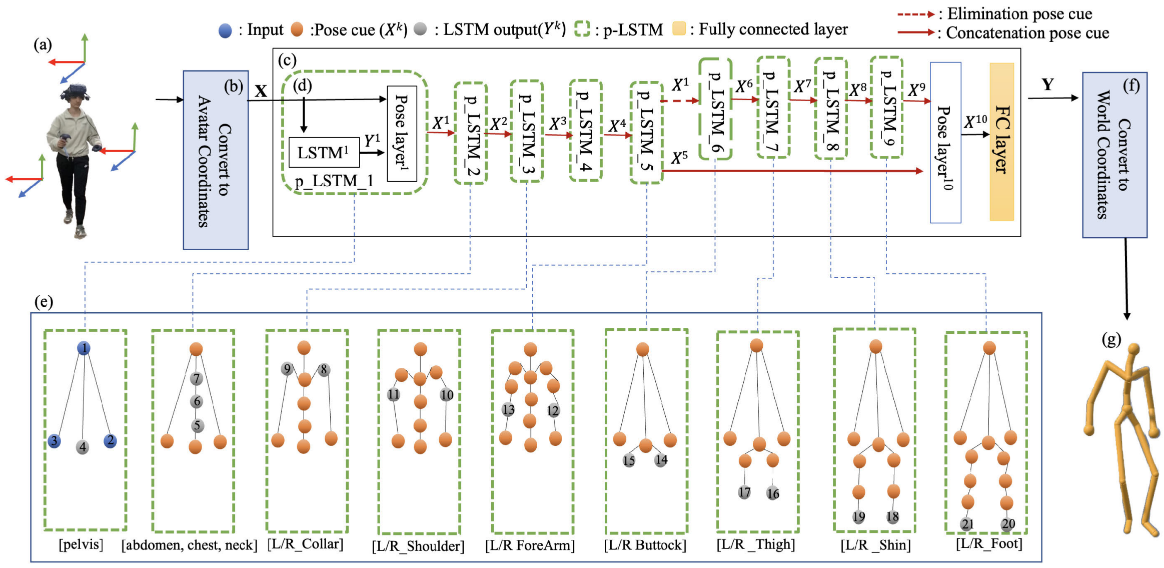

FIGURE 2. The framework of our proposed GeoPose. (a) Input trackers (b) Conversion world to avatar coordinates (c) Pose regression p-LSTMs (d) A unit of p-LSTM (e) Procedure of constructing avatar pose via p-LSTMs (f) Conversion avatar to world coordinates (g) Output pose.

Our datasets comprised a diverse range of walking, running, dancing, and exercising.

We transform two datasets to obtain the representations with respect to the avatar frame using Python and then use these transformed datasets to train our model with PyTorch. One LSTM block consists of one LSTM cell with 100 hidden units and one FC with 150 hidden units. In addition, the last FC layer has 256 hidden units after the p-LSTMs. We use the Adam optimizer [29] with a batch size of 256, a window size of 35 and a learning rate that starts from 1×10^{-3}, which decay by a factor of 0.99 every 300 iterations. It takes about four and a half hours to train our method for up to 3100 iterations on NVIDIA GeForce GTX1080 GPU.

B. METRICS

We use the following metrics: 1) mean per-joint position error (MPJPE) measures the mean Euclidean distance error of all estimated joints in centimeters with the root joint aligned; 2) mean per-joint rotation error (MPJRE) measures the mean global rotation error of all body joints in degrees; 3) root position error (rootPE) measures the Euclidean distance error of the pelvis joint in centimeters; 4) root rotation error (rootRE) measures the global rotation error of the pelvis joint in degrees.

MPJPE, and MPJRE represent the pose accuracy and are independent of the global position and rotation of an avatar instance. Since real applications are sensitive to global motion, rootPE and rootRE are crucial metrics to denote the accuracy of avatar reconstruction in MR applications. Our rotation error $\mathrm{RE}(s,t)$ of joint s at timeframe t is a Euclidean norm of the rotation vector $\mathbf{e}_{rot}(s,t)$ which corresponds to the following rotation matrix:

$$\mathbf{E}_{rot}(s,t) = \hat{\mathbf{R}}^g_w(s,t)\mathbf{R}^g_w(s,t)^T. \tag{16}$$

where $\hat{\mathbf{R}}^g_w$ and $\mathbf{R}^g_w$ are the ground truth and reconstructed values of the global rotation matrix with respect to the world frame. In all our metrics, the lower numbers are better.

C. COMPARISONS

There have been significant efforts toward generating full-body poses from sparse inputs. Among them, we compare our proposed GeoPose with AvatarPoser [14] and LoBSTr [10]. These methods use three or four inputs, consider the global translation of a root joint, try to generate a wide range of poses, and aim for real-time in commercial VR systems. Since LoBSTr does not provide public source codes, we implement their network architecture, use four inputs, including the pelvis joint, and run Final IK in Unity for the upper body joints. As AvatarPoser provides public source codes in GitHub, we run the source codes to train and test and add additional rendering in Unity. For a fair comparison, we train all the methods on the same training and testing datasets and optimize the parameters by adopting the Adam solver with batch size 256. Non-linear optimizers or different batch sizes might change the errors and running time.

1) POSE COMPARISON

The state-of-the-art methods are compared in Fig. 3 and Table 1 in terms of the generated pose accuracy. Fig. 3 depicts visual comparisons of different methods based on given sparse inputs for various motions. Avatars are color-coded to show errors in red. The superiority of our approach is evident in the qualitative results, where GeoPose yields the least error compared to other techniques.

Table 1 summarizes the accuracy evaluation of the generated poses using MPJPE and MPJRE metrics. Our MPJPE result achieves an accuracy improvement of about

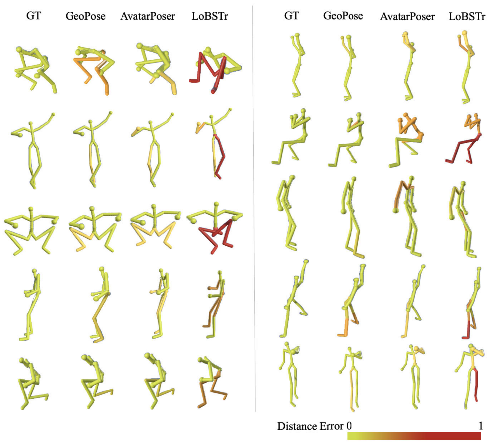

FIGURE 3. Visual results. The first column shows the ground truth. Generated poses using GeoPose, AvatarPoser, and LoBSTr are color-coded to show the large errors in red.

TABLE 1. Pose generation comparison of GeoPose with the state-of-the-art methods on CMU and HMD05 datasets. For each dataset, the best results are highlighted in boldface.

Method	Datasets	MPJPE	MPJRE
GeoPose (Ours)	CMU	**7.20**	**5.44**
	HDM05	8.16	6.92
AvatarPoser	CMU	8.37	5.93
	HDM05	**8.05**	**6.39**
LoBSTr	CMU	10.53	12.32
	HDM05	9.27	10.46

1.17 *cm* (13.9 %) compared with AvatarPoser and 3.33 *cm* (35.9 %) compared with LoBSTr on the CMU dataset. Our method gives slightly worse results, about 0.11 *cm*(1.3 %), compared with AvatarPoser on the HDM05 dataset, but the relative error is negligible. We gain about 1.11 *cm* (11.9 %) compared with LoBSTr on HDM05 dataset. Regarding MPJRE, GeoPose achieves the best performance on the CMU dataset, and AvatarPoser achieves the best on the HDM05 dataset at about 10 %. The proposed method outperforms state-of-the-art methods in MPJPE metric and performs competitively in MPJRE metric. AvatarPoser achieves the second-best performance, and LoBSTr is low.

2) ROOT JOINT COMPARISON

Table 2 reports the performance comparisons for an avatar's global translation and rotation. The pelvis joint is the root of the articulated joints, and thus the location and orientation of the pelvis joint represent those of an avatar instance. The third column shows performance comparisons for the location in a world coordinate system. Our result gains 0.41 cm (16.6 %) compared with AvatarPoser and 1.83 cm (42.5 %) compared with LoBSTr on the CMU dataset. Also, our method gives better results, about 1.51 cm (63.9 %), compared with AvatarPoser, and about 0.86 cm (26.7 %) compared with LoBSTr on the HDM05 dataset. Regarding rootPE, our method achieves the best results on both datasets and surpasses other methods. The fourth column depicts performance comparisons for the global rotation in a world coordinate system. Regarding rootRE, our method gains 1.2° compared with AvatarPoser and shows better results

TABLE 2. Global motion comparison of GeoPose with the state-of-the-art methods on CMU and HMD05 datasets. For each dataset, the best results are highlighted in boldface.

Methods	Testing Dataset	rootPE	rootRE
GeoPose(Ours)	CMU	**2.47**	**3.71**
	HDM05	**2.36**	**6.38**
AvatarPoser	CMU	2.88	4.91
	HDM05	3.87	7.58
LoBSTr	CMU	4.30	8.25
	HDM05	3.22	6.86

TABLE 3. Time (ms) comparison between GeoPose (Ours), AvatarPoser, and LoBSTr in VR. The best results are highlighted in boldface.

Methods	Inference	Generation
GeoPose(Ours)	1.27	**7.58**
AvatarPoser	1.48	16.52
LoBSTr	**1.03**	13.66

at about 2.5° compared with LoBSTr. GeoPose achieves the best performance regarding rootPE and rootRE metrics outperforming.

The errors of the root joint are accumulated through the kinematic chains and affect all the joints in the human body model. Therefore the errors of the root joint are noticeable and easily perceived by humans. Full body pose generation from sparse trackers needs to compare the root errors as well as the widely used MPJPE metric for pose accuracy. Therefore, we compared ours with the methods which reconstruct both global and local motions. We exclude FLAG in our comparison, which yields low full body error on AMASS dataset similar to AvatarPoser and assumes no global motion [13].

D. RUNNNING TIME ANALYSIS

We conducted performance evaluations. Table 3 compares a GeoPose model with LoBSTr and AvatarPoser. We measured the average running time using Unity Profiler on nVidia GeForce RTX3070 GPU. For a fair comparison, we calculate both network inference time and pose generation time. Our method and AvatarPoser show similar inference speeds, while LoBStr presents a faster inference time. AvatarPoser requires executing an IK algorithm to correct the joint angles after the network forward passes, and LoBStr requires an IK for the upper body joints. Due to the computation complexity of a Final IK algorithm [30], our method has a much shorter avatar generation time per each timeframe.

Regarding running time, reinforcement learning approaches with physics-based properties have reported relatively poor performance. Neural3Points reported a 100 ms delay [15], and QuestSim mentioned 160 ms [16]. Also, a VAE framework method [12] is slower than AvatarPoser and LoBSTr, according to the experiments conducted by Jiang et al. [14].

Compared to state-of-the-art methods, the GeoPose achieves a fast generation time, 7.58 ms, which is approximately 130 fps. Table.3 shows the network inference and pose generation time of ours, AvatarPoser, and LoBSTr methods, and our method provides fair results. Considering VR systems require 60 − 80 fps to avoid sickness, GeoPose meets the performance requirements far beyond.

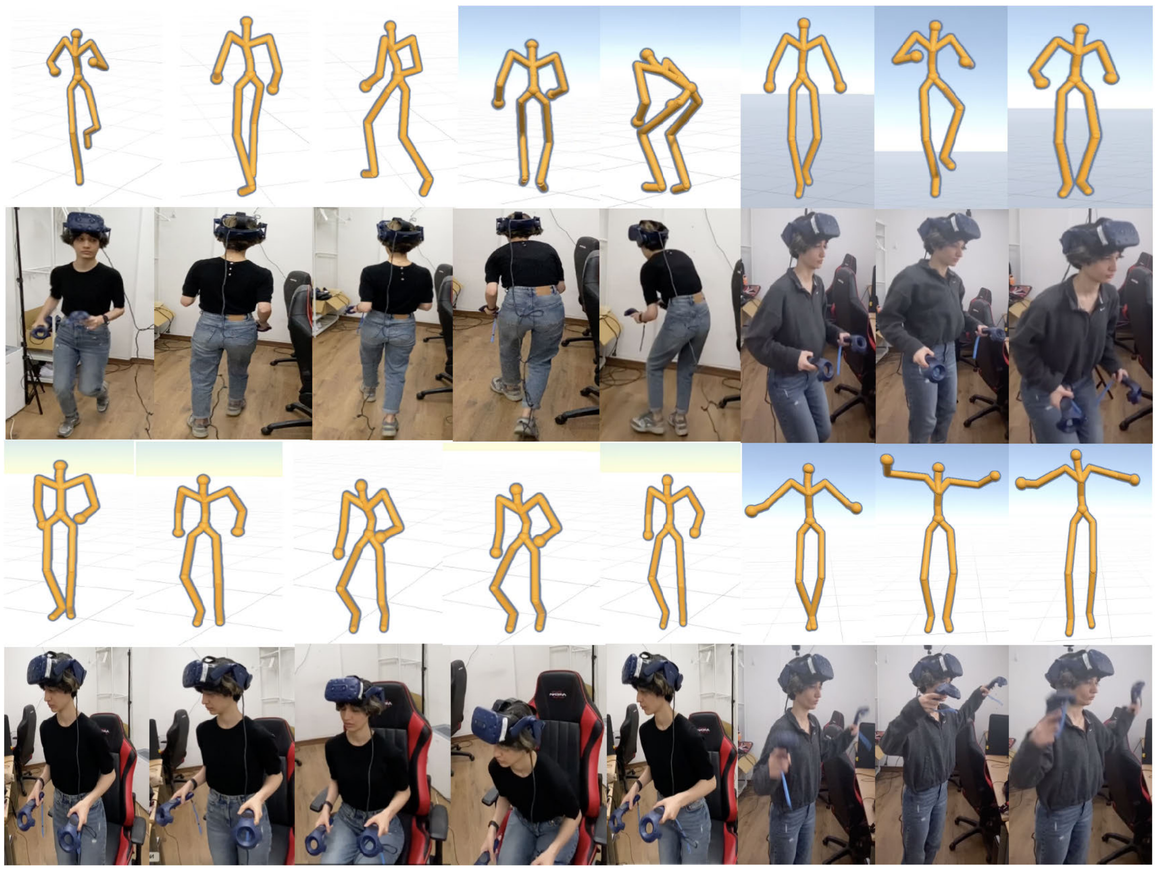
FIGURE 4. Comparison of the generated avatar using GeoPose and the VR user.

E. TEST ON A COMMERCIAL VR SYSTEM

We examine the robustness of our method in real applications where a user moves around in space and acts in various poses. We ran our algorithm online in Unity VR applications. We use HTC Vive Pro HMD and two hand-held controllers to perform a real-time test in VR. Our test motions are walking, running, sitting, jumping, bending, turning around, squatting, and so on. Figure 4 illustrates our GeoPose avatar generated from the HMD and two hand-held controllers of the user in the physical world. With tests on a commercial VR system, GeoPose proves its stability in a wide range of motions, including the user's global and local translation and rotation motions. More tests and animation videos are available in supplementary materials. See the videos for global translation and rotation motion in jumping and running.

F. LIMITATIONS

Although GeoPose generates plausible full-body poses and locates an avatar in a globally feasible position, GeoPose may produce unnatural lower-body movements, such as jittering, foot sliding, and feet penetrating the floor. Since the same sensor inputs correspond to multiple poses, the generated pose is the most possible pose rather than the accurate pose.

VII. CONCLUSION

We presented GeoPose, a novel geometry-incorporated method for a full-body pose generation of an avatar from an HMD and two controllers. GeoPose first transforms the datasets with respect to the avatar frame and then uses nine connected LSTMs model to distinguish the global motion and local pose and generate a full-body pose independently from global motion, taking into account the known kinematic information. We show that our geometry-incorporated approach is accurate, fast enough for practical MR applications, and robust under a wide range of global motions and the local pose of a user. The experimental evaluation demonstrates that our method outperforms state-of-the-art methods and is practically applicable to commercial VR systems.

There are various avenues for future research. In terms of visual representation details, the SMPL model [31] may enhance embodiment with the skinned representation of an avatar. Second, our animated avatars display unnatural artifacts on their lower legs, such as jittering and unstable foot contact. Physics-based constraints [8], [10] might improve motion quality by removing unnatural artifacts. Next, it is worthwhile to improve the accuracy further by adapting pose priors [32] and employing network models, such as variational autoencoders [13] or Transformer [33], [34], [35] for GeoPose.

REFERENCES

[1] J. Jerald, "Immersion, presence and reality trade-offs," in *The VR Book: Human-Centered Design for Virtual Reality*, 5th ed. New York, NY, USA: ACM and Morgan & Claypool, 2015, ch. 4.

[2] M. Parger, J. H. Mueller, D. Schmalstieg, and M. Steinberger, ''Human upper-body inverse kinematics for increased embodiment in consumer-grade virtual reality,'' in *Proc. 24th ACM Symp. Virtual Reality Softw. Technol.*, Nov. 2018, pp. 1–10, doi: 10.1145/3281505.3281529.

[3] G. Liu, J. Zhang, W. Wang, and L. Mcmillan, ''Human motion estimation from a reduced marker set,'' in *Proc. ACM SIGGRAPH Sketches*, 2006, pp. 35–42, doi: 10.1145/1179849.1179860.

[4] H. Liu, X. Wei, J. Chai, I. Ha, and T. Rhee, ''Realtime human motion control with a small number of inertial sensors,'' in *Proc. Symp. Interact. 3D Graph. Games*, Feb. 2011, pp. 133–140, doi: 10.1145/1944745.1944768.

[5] T. von Marcard, B. Rosenhahn, M. J. Black, and G. Pons-Moll, ''Sparse inertial poser: Automatic 3D human pose estimation from sparse IMUs,'' *Comput. Graph. Forum*, vol. 36, no. 2, pp. 349–360, May 2017, doi: 10.1111/cgf.13131.

[6] Y. Huang, M. Kaufmann, E. Aksan, M. J. Black, O. Hilliges, and G. Pons-Moll, ''Deep inertial poser: Learning to reconstruct human pose from sparse inertial measurements in real time,'' *ACM Trans. Graph. (TOG)*, vol. 37, pp. 1–15, Dec. 2018, doi: 10.1145/3272127.3275108.

[7] X. Yi, Y. Zhou, and F. Xu, ''TransPose: Real-time 3D human translation and pose estimation with six inertial sensors,'' *ACM Trans. Graph. (TOG)*, vol. 40, pp. 1–13, Jul. 2021, doi: 10.1145/3450626.3459786.

[8] X. Yi, Y. Zhou, M. Habermann, S. Shimada, V. Golyanik, C. Theobalt, and F. Xu, ''Physical inertial poser (PIP): Physics-aware real-time human motion tracking from sparse inertial sensors,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 13157–13168. [Online]. Available: https://xinyu-yi.github.io/PIP/

[9] M. Kaufmann, Y. Zhao, C. Tang, L. Tao, C. Twigg, J. Song, R. Wang, and O. Hilliges, ''EM-POSE: 3D human pose estimation from sparse electromagnetic trackers,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 11490–11500. [Online]. Available: https://ieeexplore.ieee.org/document/9710700

[10] D. Yang, D. Kim, and S. Lee, ''LoBSTr: Real-time lower-body pose prediction from sparse upper-body tracking signals,'' *Comput. Graph. Forum*, vol. 40, no. 2, pp. 265–275, May 2021, doi: 10.1111/cgf.142631.

[11] K. Ahuja, E. Ofek, M. Gonzalez-Franco, C. Holz, and A. D. Wilson, ''CoolMoves: User motion accentuation in virtual reality,'' *Proc. ACM Interact., Mobile, Wearable Ubiquitous Technol.*, vol. 5, no. 2, pp. 1–23, 2021.

[12] A. Dittadi, S. Dziadzio, D. Cosker, B. Lundell, T. Cashman, and J. Shotton, ''Full-body motion from a single head-mounted device: Generating SMPL poses from partial observations,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2021, pp. 11667–11677. [Online]. Available: https://ieeexplore.ieee.org/document/9710218

[13] S. Aliakbarian, P. Cameron, F. Bogo, A. Fitzgibbon, and T. J. Cashman, ''FLAG: Flow-based 3D avatar generation from sparse observations,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2022, pp. 13243–13252. [Online]. Available: https://www.microsoft.com/en-us/research/publication/flag-flow-based-3d-avatar-generation-from-sparse-observations/

[14] J. Jiang, P. Streli, H. Qiu, A. Fender, L. Laich, P. Snape, and C. Holz, ''AvatarPoser: Articulated full-body pose tracking from sparse motion sensing,'' in *Proc. Eur. Conf. Comput. Vis.*, 2022, pp. 443–460. [Online]. Available: https://siplab.org/projects/AvatarPoser

[15] Y. Ye, L. Liu, L. Hu, and S. Xia, ''Neural3Points: Learning to generate physically realistic full-body motion for virtual reality users,'' *Comput. Graph. Forums*, vol. 41, no. 8, pp. 183–194, 2023, doi: 10.1111/cgf.14634.

[16] A. Winkler, J. Won, and Y. Ye, ''QuestSim: Human motion tracking from sparse sensors with simulated avatars,'' in *Proc. SIGGRAPH Asia Conf. Papers*, Nov. 2022, pp. 1–8, doi: 10.1145/3550469.3555411.

[17] Vicon. (2022). *Vicon Motion System*. Accessed: May 8, 2023. [Online]. Available: http://www.vicon.com/

[18] F. J. Wouda, M. Giuberti, N. Rudigkeit, B.-J.-F. van Beijnum, M. Poel, and P. H. Veltink, ''Time coherent full-body poses estimated using only five inertial sensors: Deep versus shallow learning,'' *Sensors*, vol. 19, no. 17, p. 3716, Aug. 2019, doi: 10.3390/s19173716.

[19] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: www.deeplearningbook.org

[20] S. Hochreiter and J. Schmidhuber, ''Long short-term memory,'' *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[21] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, ''Empirical evaluation of gated recurrent neural networks on sequence modeling,'' 2014, *arXiv:1412.3555*.

[22] A. Graves, ''Supervised sequence labelling with recurrent neural networks,'' Ph.D. dissertation, TUM School CIT., Garching Bei Mnchen., Bayern, Garching, Germany, 2012.

[23] M. Xu, M. Gao, Y.-T. Chen, L. Davis, and D. Crandall, ''Temporal recurrent networks for online action detection,'' in *Proc. IEEE/CVF Int. Conf. Comput. Vis. (ICCV)*, Oct. 2019, pp. 5531–5540, doi: 10.1109/ICCV.2019.00563.

[24] C.-B. Lin, Z. Dong, W.-K. Kuan, and Y.-F. Huang, ''A framework for fall detection based on OpenPose skeleton and LSTM/GRU models,'' *Appl. Sci.*, vol. 11, no. 1, p. 329, Dec. 2020, doi: 10.3390/app11010329.

[25] K. Lee, I. Lee, and S. Lee, ''Propagating LSTM: 3D pose estimation based on joint interdependency,'' in *Proc. ECCV*, 2018, pp. 119–135.

[26] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li, ''On the continuity of rotation representations in neural networks,'' 2018, *arXiv:1812.07035*.

[27] (2004). *CMU Mocap Dataset*. Accessed: May 8, 2023. [Online]. Available: http://mocap.cs.cmu.edu

[28] (2007). *Mocap Database HDM05*. Accessed: May 8, 2023. [Online]. Available: https://resources.mpi-inf.mpg.de/HDM05/

[29] D. P. Kingma and J. Ba, ''Adam: A method for stochastic optimization,'' 2014, *arXiv:1412.6980*.

[30] (2018). *Root Motion FinalIK*. Accessed: May 8, 2023. [Online]. Available: http://assetstore.unity.com/packages/tools/animation/final-ik-14290

[31] M. Loper, N. Mahmood, J. Romero, G. Pons-Moll, and M. J. Black, ''SMPL: A skinned multi-person linear model,'' *ACM Trans. Graph.*, vol. 34, no. 6, pp. 1–16, Nov. 2015, doi: 10.1145/2816795.2818013.

[32] G. Pavlakos, V. Choutas, N. Ghorbani, T. Bolkart, A. A. Osman, D. Tzionas, and M. J. Black, ''Expressive body capture: 3D hands, face, and body from a single image,'' in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2019, pp. 10967–10977.

[33] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, ''Attention is all you need,'' in *Proc. Adv. Neural Inf. Process. Syst.*, Long Beach, CA, USA, 2017, pp. 1–11.

[34] J. Devlin, M. Chang, K. Lee, and K. Toutanova, ''Pre-training of deep bidirectional transformers for language understanding,'' in *Proc. NAACL-HLT*, Minneapolis, MN, USA, 2019, pp. 4171–4186.

[35] E. Aksan, M. Kaufmann, P. Cao, and O. Hilliges, ''A spatio-temporal transformer for 3D human motion prediction,'' in *Proc. Int. Conf. 3D Vis. (3DV)*, Dec. 2021, pp. 565–574, doi: 10.1109/3DV53792.2021.00066.

TARAVAT ANVARI received the B.S. degree in information technology and computer engineering from the Sadjad University of Technology, Iran, in 2018. She is currently pursuing the joint master's and Ph.D. degree with Chung-Ang University, Seoul, South Korea. Her research interests include virtual reality, deep learning, and computer graphics.

KYOUNGJU PARK (Member, IEEE) received the B.E. degree in computer engineering from Ewha Womans University and the M.S. and Ph.D. degrees in computer and information science from the University of Pennsylvania, Philadelphia, PA, USA, in 2004. From 2004 to 2005, she was a Research Professor with Rutgers University, New Brunswick, NJ, USA. From 2005 to 2007, she was a Researcher with the Samsung Electronics Multimedia Laboratory, Suwon, South Korea. Since 2007, she has been an Assistant Professor and an Associate Professor with the Computer Science and Engineering Department, Chung-Ang University, Seoul, South Korea. Her research interests include virtual reality, avatar embodiment, artificial intelligence, computer graphics, and computer vision.