Real-time Retargeting of Deictic Motion to Virtual Avatars for Augmented Reality Telepresence

Jiho Kang* Dongseok Yang† Taehei Kim‡ Yewon Lee§ Sung-Hee Lee¶

Graduate School of Culture Technology, KAIST

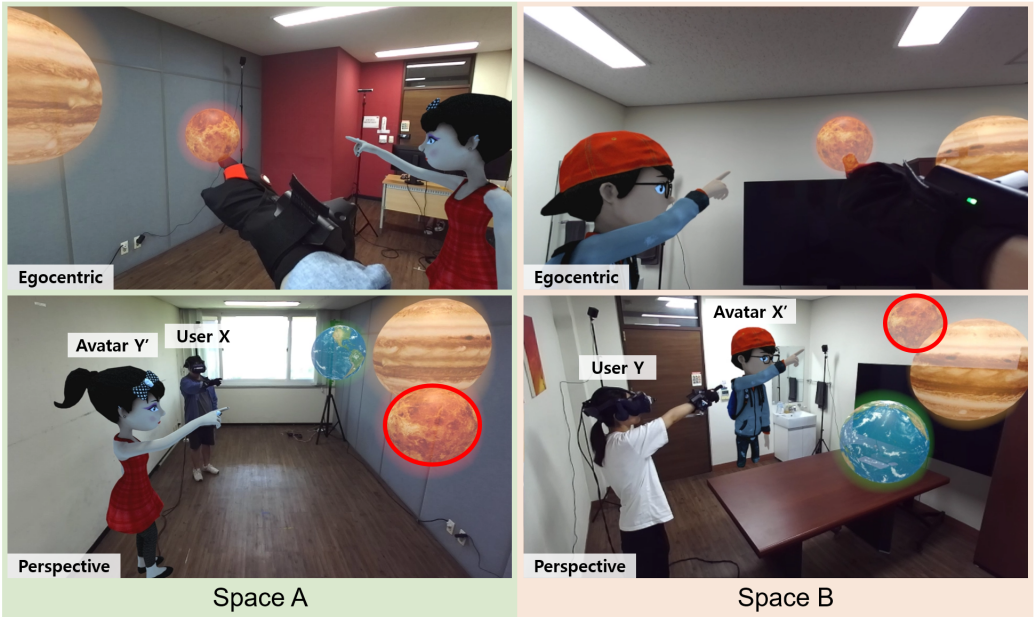


Figure 1: User egocentric (top) and room perspective (bottom) view images of the two spaces A (left) and B (right) involving in an example AR telepresence scenario. User X and Y are in spaces A and B, respectively. Avatar X′ in space B represents user X, while avatar Y′ in space A represents user Y. Spaces A and B have different sizes and arrangements of objects. When the user points at a particular object (Venus marked with a red circle), our system retargets the user's motion for their avatar to appropriately point at the corresponding object in the remote space, facilitating two-way deictic interaction between distant users.

Abstract

Avatar-mediated augmented reality telepresence aims to enable distant users to collaborate remotely through avatars. When two spaces involved in telepresence are dissimilar, with different object sizes and arrangements, the avatar movement must be adjusted to convey the user's intention rather than directly following their motion, which poses a significant challenge. In this paper, we propose a novel neural network-based framework for real-time retargeting of users' deictic motions (pointing at and touching objects) to virtual avatars in dissimilar environments. Our framework translates the user's deictic motion, acquired from a sparse set of tracking signals, to the virtual avatar's deictic motion for a corresponding remote object in real-time. One of the main features of our framework is that a single trained network can generate natural deictic motions for various sizes of users. To this end, our network includes two sub-networks: AngleNet and MotionNet. AngleNet maps the angular state of the user's motion into a latent representation, which is subsequently converted by MotionNet into the avatar's pose, considering the user's scale. We validate the effectiveness of our method in terms of deictic intention preservation and movement naturalness through quantitative comparison with alternative approaches. Additionally, we demonstrate the utility of our approach through several AR telepresence scenarios.

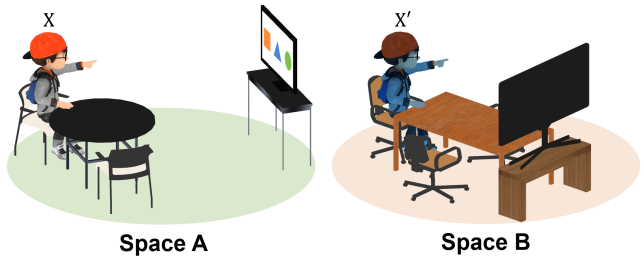
Index Terms: Computing methodologies—Computer graphics—Animation; Human-centered computing—Human computer interaction (HCI)—Interaction paradigms—Mixed / augmented reality

1 Introduction

Recent advancements in Augmented Reality (AR) technology have enabled the realization of avatar-mediated telepresence, allowing geographically separated users to interact in their own spaces through virtual copies of themselves. This promising medium for next-generation communication offers a unique level of immersion and presence, enabling users to engage in more intuitive and natural interactions with others.

One remaining challenge in implementing avatar-mediated AR telepresence is to ensure that virtual avatars accurately represent the intention of the users despite spatial discrepancies between the spaces involved. Previous approaches focused on finding mutually usable spaces [12] or relocating virtual avatars [34] to appropriate placements for given interaction with the remote user. Nonetheless,

*e-mail: jhkang0408@kaist.ac.kr
†e-mail: dsyang@kaist.ac.kr
‡e-mail: hayleyy321@kaist.ac.kr
§e-mail: yeone22@kaist.ac.kr
¶Corresponding author. e-mail: sunghee.lee@kaist.ac.kr



(a)

(b)

Figure 2: (a) An example problem in AR telepresence between dissimilar spaces, specifically, the TVs in space A and space B have different sizes and locations. Suppose the user X in the space A points at the center of the TV. If we simply copy the motion of user X to the avatar X′ in space B, the avatar X′ will not point at the center of the TV, failing to convey the meaning of the user's motion. (b) While the user's body parts are moving towards the target for pointing, the avatar needs to move towards a corresponding target, which can be in a different egocentric position than the user's target, synchronously such that the pointing action completes at the same time.

finding mutually usable spaces or an optimal placement can be challenging when the identical spatial relationship between the user, avatar, and interaction target object, is unavailable by the different space layout; naive mirroring of motion between the user and avatar can lead to misalignment that results in communication breakdowns.

Figure 2 (a) demonstrates an example telepresence scenario; avatar X′ in space B is placed on a chair that best represents the user X in space A. However, merely copying and pasting the user's motion into avatar X′ results in the avatar appearing to point at the air, which fails to convey the intention of user X's motion pointing at the TV. Therefore, the avatar's deictic motion needs to be modified to accurately convey the intention between users. Such real-time motion retargeting is challenging because the user intention is not explicitly given, and the degree of discrepancy between space layouts varies. Additionally, varying scales of user and avatar must be considered as well for general use.

In this paper, we introduce a novel retargeting framework that addresses spatial dissimilarity and allows the virtual avatar to clearly transfer the meaning of the user's deictic motion. Among various types of motions, we choose the deictic motion, specifically pointing and touching, as they are mainly used for non-verbal communication during remote interaction. For effectively retargeting deictic motions in practical applications, it is essential to possess information about object correspondences between two spaces, as well as the target object that a deictic motion is directed towards. Ideally, this information should be automatically acquired through object and action recognition. However, in our study, we manually predefine such information and concentrate on the development of the retargeting motions.

Given the interaction target, we extract features between the user joints and the target object. Then, our model solves the problem of determining the corresponding avatar joint transformations with respect to the remote space targets to animate the avatar in real-time. This problem is significantly more challenging than it initially seems. When a user is statically pointing at a certain point, it is trivial to determine the corresponding avatar pose that points at a corresponding point. However, as seen in Figure 2 (b), if the user is in the middle of moving their head, eye, and hand towards a target point, it is uncertain when the user's gaze and pointing will reach the target. In such a case, it is therefore not straightforward to determine the corresponding avatar pose that moves towards the remote target and eventually reaches the target at the same time as the user.

To solve this problem, we take a data-driven approach. After collecting a dataset of deictic motions that move towards various target points, we train a deep neural network that learns to predict a corresponding avatar pose given the time window of the user's deictic motion towards a target.

Our model is designed to robustly generate avatar motion regardless of different user scales. To this end, we encode the user and avatar motions through angle-based features using AngleNet. We demonstrate that our angle-based features are more robust than using other features, such as the position and orientation of joints. Then the latent vector is combined with the user's scale and decoded into transformations of the avatar's head and right hand and position of the right index fingertip. These joints are used as end-effectors by the inverse kinematics (IK) to generate the avatar's deictic motion.

To the best of our knowledge, this work is the first data-driven approach to retarget a user's deictic motion to the virtual avatar in a dissimilar space layout. We thoroughly validate our method by quantitative evaluation in terms of deictic intention preservation and movement naturalness. Additionally, we implement a prototype AR telepresence application to demonstrate the utility of our method.

In this study, the spatial discrepancy between spaces is limited to the case that the spaces have corresponding objects of the same type but their placement and shape differ. The target objects include avatars, TV, and virtual objects. According to Mayer et al. [20], pointing errors are same regardless of whether a person is sitting or standing. Therefore, we narrow our focus to deictic motions performed while standing in place.

To summarize, this paper makes two main contributions:

- The first learning-based approach for retargeting of deictic motion to virtual avatars in dissimilar environments.
- A neural network-based framework, trained on single person data, that can translate deictic motions from various users to their avatars.

2 Related Work

2.1 AR Telepresence

AR Telepresence has been an active research area in recent years for its ability to maintain a sense of closeness and personal interaction without physical presence. Several research studies have explored the potential of this technology, including methods to address technical challenges, improve the user experience, and enhance the effectiveness of remote collaboration.

Pioneering works proposed prototype systems to realize AR telepresence with projectors and displays [2,16] to render captured images of the remote space in local space. Follow-up research utilized advanced devices such as head-mounted displays (HMDs), multiple projectors, depth cameras, and haptic devices [17, 27] to improve user immersion. As real-time 3D capture and reconstruction become possible, Orts et al. [21] introduced Holoportation, an end-to-end telepresence system transmitting high-quality 3D representations of remote people and objects, thereby edging the remote audio-visual communication closer to face-to-face. Recent research has advanced AR telepresence technology, enabling it to be effectively

used for remote collaboration. For example, mini-Me [23] introduced a scaled avatar representation that accurately transfers the user's gaze and body gestures while staying within the partner's field of view. Loki [28] incorporated multiple user interfaces such as 2D video, 3D visualization of remote spaces, and interactive annotations. An inherent challenge for AR telepresence would be the discrepancy in spatial layout between the distant spaces; virtual avatars need to be placed and move adaptive to the dissimilar formation of the interaction targets (i.e. user and objects) to correctly deliver the intention of corresponding users.

Lehment et al. [15] pioneered the concept of a shared workspace to maximize common features in participants' physical surroundings. Extending this concept to an optimal mutual virtual space, Keshavarzi et al. [12] introduced a method for suggesting object movements, aiming to expand the mutual space with minimal physical effort. Fink et al. [4] proposed a user-defined workspace that relocates avatars based on the interaction target objects while Grønbæk et al. [7] further improved MR collaboration productivity by partially aligning distant physical spaces.

As shared or defined spaces inevitably result in a reduction of usable space, many researchers have explored ways to optimize avatar placement and motion for effective remote interaction. Jo et al. [11] established spatial and object-level matches between spaces to adapt avatar position and motion accordingly. Pejsa et al. proposed Room2Room [22], projecting remote participants onto physically plausible locations for natural conversational formations. Kim et al. [14] introduced an object-level correspondence map for retargeting user-object interaction to avatars with varying shapes and sizes. Yoon et al. [34] proposed a data-driven approach, measuring the similarity between local and remote placements based on surrounding interactable entities. Wang et al. [30] extended the research by predicting user arrival locations and controlling avatar locomotion speed. Our work is on the same line to expand space usability by retargeting the deictic motion of the user to the avatar.

2.2 Deictic Motion in XR

Deictic motion, which encompasses gestures like pointing and touching, is a critical component of non-verbal communication and has gained significant attention in the fields of eXtended Reality (XR). Previous research has primarily focused on the interpretation, modeling, and retargeting of these gestures, with a particular emphasis on their application in human-computer interaction.

Kim et al. [13] proposed a method to first find the optimal placement of the remote avatar, then retarget the remote user's deictic gesture. Yoon et al. [33] went one step further and tackled the placement, arm gesture, and head movement of the local user to the avatar, in order to preserve the environment and interaction context of the local user. Ullal et al. [29] preserved the pose while redirecting the gesture using a multi-objective optimization framework. Fidalgo et al. [3] distorted the remote user's gestures in a 3D plane to correctly reflect them from the local user's perspective, particularly in face-to-face situations. Their method significantly improves gesture recognition.

The second area of research focused on the interpretation of deictic gestures since it involves two parties; the deictic host and the observer. The exact interpretation of deictic gestures received attention due to the importance of exact meaning conveyance [31, 32]. Therefore, researchers focused to improve the method to better interpret the deictic gesture. Plaumann et al. [24] demonstrated that acknowledging users' handedness and ocular dominance can significantly improve pointing accuracy. Sousa et al. [26] warped the gesture direction for better interpretability when the user cannot distinguish the distortion. Mayer et al. [19] suggested a model that improves the interpretation of deictic gestures at targets from all directions of the user. Some work focused on haptic or reaching position retargeting which usually deceives the visual sense by haptic manipulation [1] to improve the presence within the VR environment or by adopting a sensorimotor model [6]. All these studies showed impressive advancement in deictic gesture generation and interpretation. Our work tackled retargeting users' deictic gestures in remote spaces that have a different layout in real-time.

3 Dataset

3.1 Data Capture and Labeling

Plaumann et al. [24] noted considerable variability in pointing gestures across individuals. Similarly, users' touch behavior can vary, often influenced by the part of the hand they use. To address this variation, we defined the completion state (CS) of deictic motions to collect data, minimizing individual differences. CS refers to the pose when a user accurately points at or touches the target. For pointing, CS is defined as the user positioning the tip of the right index fingertip at the target location within their field of view, assuming the eye-finger ray cast (EFRC) intersects the target. EFRC has shown better pointing accuracy of the pointer compared to other techniques such as index finger ray cast (IFRC), forearm ray cast (FRC), and head ray cast (HRC) [19]. For touching, CS is achieved when the right index fingertip makes contact with the target. With these rules, we recorded four action categories: pointing at a single target, pointing at two targets in sequence, touching a single target, and touching two targets in sequence. Subjects remained stationary while performing these actions towards a target presented in a virtual environment created using the Unity3D engine. We captured the joint transformations of the head, right hand, and right index fingertip, along with the target's position, tracked with an HTC Vive Pro headset, and Noitom Hi5 VR Gloves as shown in Figure 3.

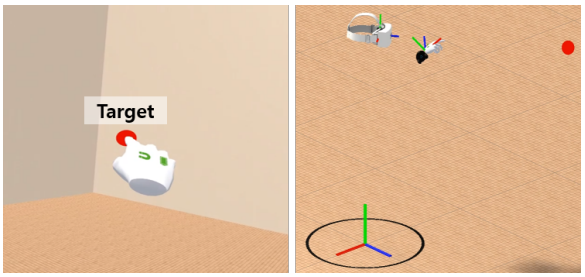


Figure 3: Example egocentric (left) and perspective (right) view images of our data capture scene.

To make our dataset include evenly distributed data over the possible range of deictic motions, we captured motions by arranging target objects in grids. For pointing at and touching a single target, the targets were arranged in a $9 \times 3 \times 1$(azimuth $\times$ height $\times$ distance) grid for pointing and a $5 \times 3 \times 2$ grid for touching in cylindrical coordinates. The azimuth angle covered a span of 140° (-70° to 70°) with the interval of 17.5° for pointing and 35° for touching. Height variations were set at -0.5m, 0m, 0.5m from the eye-level for pointing, and 0m, -0.3m, and -0.6m for touching. Distance was set at 1.5m for pointing and at 0.4m and 0.5m for touching. In the case of actions involving two targets, the targets were arranged in a $5 \times 3 \times 1$ grid for both pointing and touching, with the same grid span as the single-target scenario. Variations in azimuth angle were sampled every 35° for both pointing and touching. Height variations were identical to the single-target scenario. Distance varied by 1.5m for pointing and by 0.45m for touching. In each trial, two different targets were selected: all cases were captured for the training data, and 8 randomly selected cases were captured for the test data.

The training data was obtained from an individual who stands 170cm tall. This individual received instructions to perform the actions naturally, with an emphasis on minimizing personal movement styles to maintain data neutrality. On the other hand, the test data

was collected from three subjects with heights of 161cm, 172cm, and 179cm. For the test subjects, no specific instructions were given apart from providing the CS of deictic motions. Among the training and test subjects, two (170cm, 172cm) are right-eye dominant, and the others are left-eye dominant.

During actions involving two targets, we applied linear interpolation to determine the target position during the transition. Table 1 presents an overview of our recorded data length for subjects of different heights.

Table 1: Length of each action category for different subjects.

Category	Height			
	170cm	161cm	172cm	179cm
Pointing at single target	236s	91s	97s	103s
Pointing at two targets	1895s	49s	39s	32s
Touching single target	283s	125s	140s	96s
Touching two targets	2030s	46s	48s	28s

3.2 Motion Pairing

To enable seamless deictic interactions without time delay, it's crucial to synchronize the CS moments between the user and the avatar. This synchronization necessitates matching the avatar's motion with that of the user. To this end, we generated paired motion sequences that belong to the same action category but have different target positions. The motion pairs serve as the ground truth data for the input user motion and its corresponding avatar motion.

To make a motion pair, we randomly select two motions of the same action category, and then time-warp the motions such that they take actions synchronously. Specifically, we define six states in a deictic motion: (Idle) looking straight ahead with right hand down, (Gaze) turning the head towards the target, (Begin) starting to point at or touch object, (Hold) maintaining CS, (End) returning to Idle state, and (Transition) transitional motion during a target change. For each motion, we manually divide it based on the defined states. Then, we compared the lengths of each state to synchronize the motion timing between the two sequences. To match the duration of the shorter states with the longer ones, we employed monotone piecewise cubic interpolation [5]. Lastly, we segmented the synchronized sequences into multiple motion clips, each comprising 30 frames with an overlap of 10 frames.

3.3 Data Augmentation

We augmented our training data (170cm subject) to ensure that our network can account for users of different scales. We scaled the positions of the right hand, right index fingertip, and target with respect to the head joint transformation. Figure 4 shows the result of scaling the pointing and touching poses. As can be seen, the scaled pointing and touching motions remain consistent with the scaled target position. The scaling values used for training are specified in Sections 4.3 and 5.1.

4 Method

Figure 5 illustrates the overview of our framework, which consists of two distinct models of AngleNet and MotionNet. We begin by obtaining input angle features for both the user and the avatar; angle features are derived from the history of transformations of the head, hand, index fingertip joints, and target position. AngleNet takes the angle feature sequences from both the user and the avatar and maps them into latent representations. These latent sequences are then passed to a recurrent module to generate a control vector for MotionNet. From the control vector along with scale information of the user, MotionNet predicts transformations of the avatar's head, right-hand joint, and index fingertip. Subsequently, the upper-body pose of the avatar in the current frame is computed by an IK solver [25].

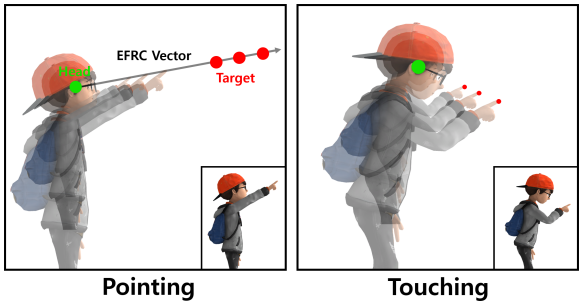


Figure 4: Data augmentation by scaling target position at completion state (CS) pose.

4.1 Input and Output

Our framework is designed to translate the deictic motion of various users into their avatars, utilizing the motion data from a small number of people (even a single person in our work). To achieve this generalization, we employ a user-invariant input representation based on angles and an avatar-invariant output representation (end-effector transformations). We emphasize that our angle-based input representation is more robust against variations in deictic motion style and body size, surpassing the limitations of a position-rotation representation. Moreover, representing the output with end-effector transformations enables us to effortlessly generate the deictic motion of avatars with varying upper-body shapes using an IK solver. For network training and experiments, we use the widely adopted 6D representation for rotation [35].

Before introducing input and output representations, we define reference frames and important direction vectors (see Fig. 6 (a)). The head (H) joint is located at the mid-point of two eyes. The root (O) joint transformation is defined as the head joint transformation when the user/avatar is in a default, upright standing posture. Specifically, the root position corresponds to the head position, and the orientation of the root is determined by the head forward direction projected onto the ground ($y = 0$) and the global up direction. We define the base directions from head to target ($\overrightarrow{\text{HT}}$), from head to index fingertip of the right hand ($\overrightarrow{\text{HI}}$), and the forward vector of the head ($\overrightarrow{\text{HF}}$) to define the input feature.

The user input $\mathbf{x}_t = \{\mathbf{a}_t, \mathbf{t}_t\} \in \mathbb{R}^{15}$ at current time frame t is composed of angle feature $\mathbf{a}_t \in \mathbb{R}^{12}$ and target feature $\mathbf{t}_t \in \mathbb{R}^3$. The angle feature $\mathbf{a}_t = \{\text{T}, \text{F}, \text{I}, \text{TF}, \text{TI}, \text{FI}\}$ consists of angles derived from the base directions, where $\text{T} \in \mathbb{R}^2$ denotes the two angles when $\overrightarrow{\text{HT}}$ is projected to the horizontal and sagittal planes. F and I are defined similarly with respect to $\overrightarrow{\text{HF}}$ and $\overrightarrow{\text{HI}}$. Additionally, $\text{TF} \in \mathbb{R}^2$ denotes the two (horizontal and vertical) angles between $\overrightarrow{\text{HT}}$ and $\overrightarrow{\text{HF}}$, both projected to the horizontal and sagittal planes. TI and FI are defined the same way between $\overrightarrow{\text{HT}}$ and $\overrightarrow{\text{HI}}$, and between $\overrightarrow{\text{HF}}$ and $\overrightarrow{\text{HI}}$, respectively. While $\{\text{TF}, \text{TI}, \text{FI}\}$ are derivable features from $\text{T}, \text{F}, \text{I}$, we empirically found that directly specifying these features as input helps the network produce higher-quality motions. Refer to Figure 6 (b) for further examples regarding the horizontal and vertical angles.

The user's target feature is represented as $\mathbf{t}_t = \{\theta_t, \phi_t, d_t\} \in \mathbb{R}^3$: horizontal θ_t and vertical ϕ_t angles and a distance d_t to the position of the predefined target relative to the root. We set $d_t = 1$ when $d_t > 1\text{m}$ to encourage our model to generate pointing motions in a distance-invariant manner.

The network receives the input sequence as $\mathbf{X}_{t-N+1:t} = \{\mathbf{x}_{t-N+1}, \cdots, \mathbf{x}_t\} \in \mathbb{R}^{N \times 15}$ with a time window from $t - N + 1$ to t. The avatar input sequence $\mathbf{X}'_{t-N:t-1} \in \mathbb{R}^{N \times 15}$ is computed using the same way, but with one time frame shifted backward. The window size N is empirically set to 20 ($= 0.667$ second for 30 fps) for effec-

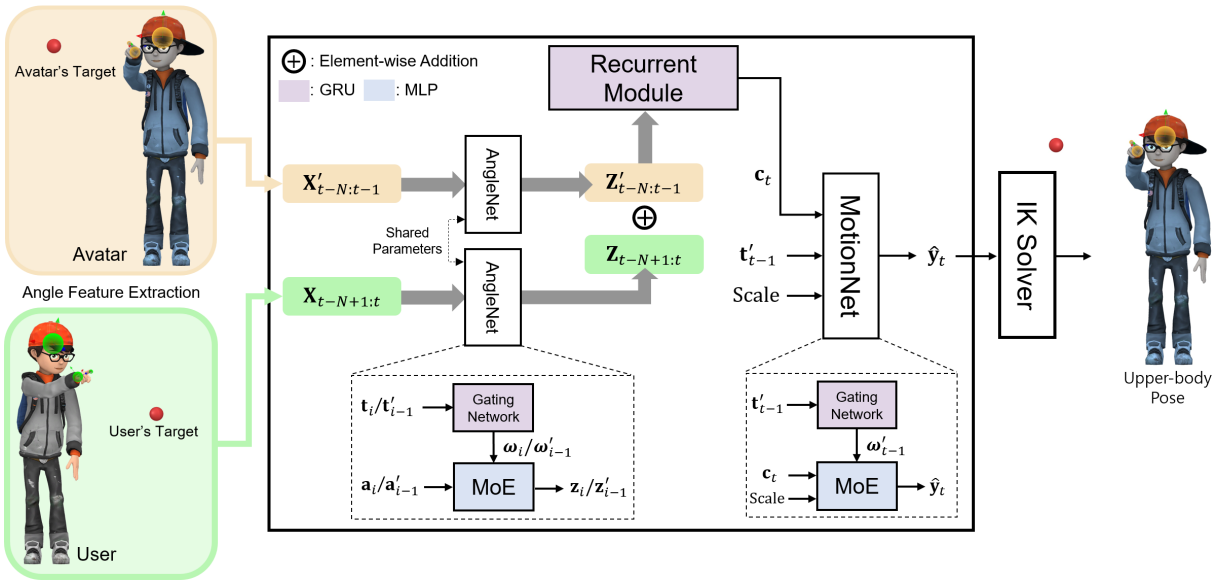


Figure 5: Our neural network-based framework retargeting of deictic motion to virtual avatars in dissimilar environments. The process begins with the calculation of angle features from the user and the avatar. These features are then transformed into latent vectors by AngleNet. Next, a recurrent module maps the sequence of latent vectors to a control vector. With this vector and the user's scale value, MotionNet regresses the avatar's end-effectors. Finally, an IK solver computes the avatar's upper-body pose.

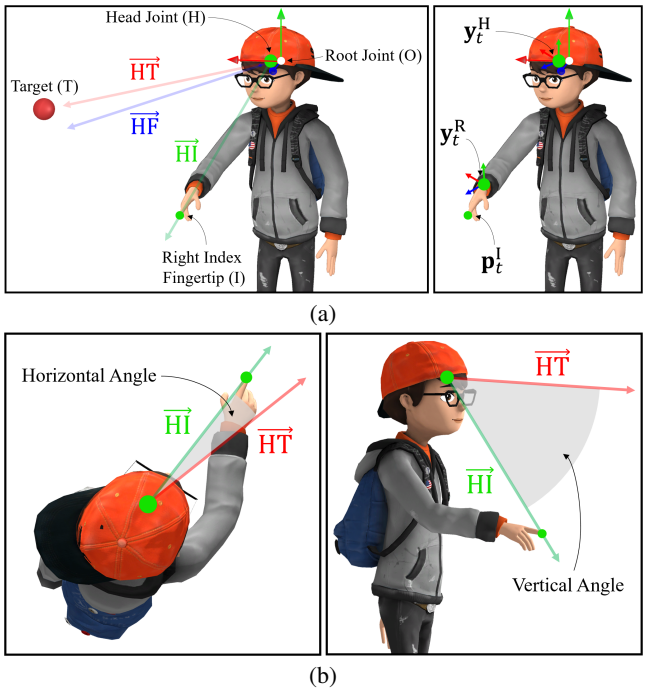


Figure 6: (a) Left: Our angle features are extracted using direction vectors defined at the head (H), root (O), right index fingertip (I) joints, and target (T). Right: Network output comprises the transformations of the head and the right hand as well as the position of the right index fingertip. (b) Illustrations of the horizontal (left) and vertical (right) angles between $\overrightarrow{HT}$ and $\overrightarrow{HI}$ projected onto the horizontal and sagittal planes, which are defined for the root joint transformation.

tively capturing temporal characteristics of deictic motions and the relationship between joints and the target object.

The network output $\hat{\mathbf{y}}_t = \{\hat{\mathbf{y}}_t^{\mathrm{H}}, \hat{\mathbf{y}}_t^{\mathrm{R}}, \hat{\mathbf{p}}_t^{\mathrm{I}}\} \in \mathbb{R}^{21}$ is composed of avatar's target transformations for head $\hat{\mathbf{y}}_t^{\mathrm{H}} \in \mathbb{R}^9$ and the right hand $\hat{\mathbf{y}}_t^{\mathrm{R}} \in \mathbb{R}^9$, and the position of right hand's index finger tip $\hat{\mathbf{p}}_t^{\mathrm{I}} \in \mathbb{R}^3$.

4.2 Network Architecture

Our model functions by dynamically adjusting the network parameters of AngleNet and MotionNet based on the sequence of target features; the target position steers these network parameters, enabling them to serve as smoothly changing regressors. This process constructs a stable latent space, even for unobserved target positions of both user and avatar during training, that connects the angle features of the user with the head and hand joints of the avatar, thereby enabling the generation of seamless deictic motion of the avatar. Despite our training dataset containing deictic motions from only a single person at sparse target positions, this design ensures the robustness and generality of our framework. It is built to handle real-world scenarios with widely varying target positions and to translate the deictic motions of various users to their avatars. Moreover, with only a few milliseconds of inference time ($\approx$ 3.7ms on an Intel i9 processor) and a few kilobytes of memory requirement (49kB), our model is both fast and lightweight, making it suitable for real-time telepresence applications.

We adopt the mixture of expert (MoE) [10] models for AngleNet and MotionNet. A shared gating network dynamically adjusts the weights of both networks according to the given target position. The gating network, a two-layer gated recurrent unit (GRU), receives the target feature $\mathbf{t}$ for the user or $\mathbf{t}'$ for avatar and determines the blending coefficients ω or $\omega' \in \mathbb{R}^6$ ($\omega = \{\omega^{(i)}\}_{i=1}^{6}$) of MoE in AngleNet and MotionNet. The operation of a gating network is defined as:

$$\omega_t = \text{gating network}(\mathbf{t}_t) = \sigma(\text{GRU}(\mathbf{h}_{t-1}^{gat}, \mathbf{t}_t)) \tag{1}$$

where a softmax function σ makes the sum of blending coefficients 1 and $\mathbf{h}_{t-1}^{gat} \in \mathbb{R}^6$ is the hidden state of second layer in the previous frame.

Both AngleNet and MotionNet have 6 experts; each expert is a two-layer multilayer perceptron (MLP). AngleNet takes the input

sequence $\mathbf{X}_{t-19:t}$ from user or $\mathbf{X}'_{t-20:t-1}$ from avatar, and outputs corresponding latent sequences of $\mathbf{Z}_{t-19:t} \in \mathbb{R}^{20\times16}$ or $\mathbf{Z}'_{t-20:t-1}$. The operation of AngleNet can be written as:

$$\mathbf{z}_t = \text{AngleNet}(\mathbf{a}_t; \alpha(\omega_t)) = \mathbf{W}_1\text{ELU}(\text{BN}(\mathbf{W}_0\mathbf{a}_t + \mathbf{b}_0) + \mathbf{b}_1) \quad (2)$$

The weights of AngleNet $\alpha(\omega_t) = \{\mathbf{W}_0 \in \mathbb{R}^{12\times16}, \mathbf{W}_1 \in \mathbb{R}^{16\times16}, \mathbf{b}_0 \in \mathbb{R}^{16}, \mathbf{b}_1 \in \mathbb{R}^{16}\} = \sum_{i=1}^{6} \omega_t^{(i)} \alpha^{(i)}$ are blended by expert weights $\{\alpha^{(i)}\}_{i=1}^{6}$. We utilize batch normalization (BN) and the exponential linear unit (ELU) as the activation function.

We perform element-wise summation for $\mathbf{Z}_{t-19:t}$ and $\mathbf{Z}'_{t-20:t-1}$ and feed them to a two-layer GRU recurrent module. This module captures temporal features to compute a control vector $\mathbf{c}_t \in \mathbb{R}^{16}$ for MotionNet to produce the current end-effector transformations. The operation of the recurrent module is defined as:

$$\mathbf{c}_t = \text{recurrent module}(\mathbf{z}_t + \mathbf{z}'_{t-1}) = \text{GRU}(\mathbf{h}^{rec}_{t-1}, \mathbf{z}_t + \mathbf{z}'_{t-1}) \quad (3)$$

where $\mathbf{h}^{rec}_{t-1} \in \mathbb{R}^{16}$ is the hidden state of the second layer in the previous frame.

Given $\mathbf{c}_t$ and the scale information of the user, MotionNet outputs the desired avatar end-effector transformations $\hat{\mathbf{y}}_t$. The operation of MotionNet is denoted as:

$$\begin{aligned}\hat{\mathbf{y}}_t &= \text{MotionNet}(\{\mathbf{c}_t, \text{scale}\}; \beta(\omega'_{t-1})) \\ &= \mathbf{W}_1^{\dagger}\text{ELU}(\text{BN}(\mathbf{W}_0^{\dagger}\{\mathbf{c}_t, \text{scale}\} + \mathbf{b}_0^{\dagger}) + \mathbf{b}_1^{\dagger}).\end{aligned} \quad (4)$$

The weights of MotionNet $\beta(\omega'_{t-1}) = \{\mathbf{W}_0^{\dagger} \in \mathbb{R}^{(16+1)\times16}, \mathbf{W}_1^{\dagger} \in \mathbb{R}^{16\times21}, \mathbf{b}_0^{\dagger} \in \mathbb{R}^{16}, \mathbf{b}_1^{\dagger} \in \mathbb{R}^{21}\} = \sum_{i=1}^{6} \omega'^{(i)}_{t-1} \beta^{(i)}$ are blended by expert weights $\{\beta^{(i)}\}_{i=1}^{6}$.

4.3 Training

We augmented the training data by applying scale factors of 0.9 (170cm $\rightarrow$ 153cm) and 1.1 (170cm $\rightarrow$ 187cm) to the touching motions, using the techniques in Section 3.3. We did not augment the pointing motions, for which we set the scale to 1. As shown in Figure 4, the EFRC vector of the pointing motion remains invariant to the user scale, so we chose to scale the network output directly after training.

We adopt a curriculum learning strategy that involves gradually transitioning from easier to more challenging tasks. The task difficulty is determined by the number of targets involved in the deictic motion: an easy task with a single target and a hard task with two targets. To ensure diversity in the hard task, we re-sample the data for different target positions at each epoch.

Reconstruction Loss. Reconstruction loss is defined as L1 norm between generated and ground truth end-effector transformations:

$$L_{\text{rec}} = \frac{1}{T}\sum_{i=0}^{T-1} \|\mathbf{y}_i - \hat{\mathbf{y}}_i\|_1 \quad (5)$$

where the output length $T = 10$ is obtained by subtracting the window size $N = 20$ from motion clip length $L = 30$.

Adversarial Loss. To generate natural end-effector trajectories, we leverage an adversarial loss combined with short-term and long-term discriminators. The short-term discriminator D_s assesses the smoothness between frames, while the long-term discriminator D_l evaluates the continuity throughout the sequence. We employ discriminator architecture proposed by [8], which is a fully convolutional network using 1D temporal convolution. The kernel size of the first layer is specifically set to 2 for the short-term discriminator and 10 for the long-term discriminator. The inputs to the discriminator include a sequence of positions, rotations, linear velocities, and angular velocities for both head and right hand joints. Linear velocity and angular velocity are approximated using the finite difference of position and rotation, respectively. We calculate the average of losses for the discriminator, which operates by using a sliding window. Adversarial loss is defined as Least Square GAN equation [18] using past sequence $\mathbf{Y}^{\text{P}}_{\text{adv}}$, ground truth sequence $\mathbf{Y}_{\text{adv}}$ and generated sequence $\widehat{\mathbf{Y}}_{\text{adv}}$:

$$L_{\text{dis}}(D) = (D(\{\mathbf{Y}^{\text{P}}_{\text{adv}}, \mathbf{Y}_{\text{adv}}\}) - 1)^2 + D(\{\mathbf{Y}^{\text{P}}_{\text{adv}}, \widehat{\mathbf{Y}}_{\text{adv}}\})^2 \quad (6)$$

$$L_{\text{gen}}(D) = (D(\{\mathbf{Y}^{\text{P}}_{\text{adv}}, \widehat{\mathbf{Y}}_{\text{adv}}\}) - 1)^2 \quad (7)$$

Total Loss. The entire framework and discriminators are jointly trained with a relative weight λ_{adv} of 0.5 for adversarial loss. The total loss of our framework is:

$$\min_{\text{Framework}} L = L_{\text{rec}} + \lambda_{\text{adv}}(L_{\text{gen}}(D_s) + L_{\text{gen}}(D_l)) \quad (8)$$

and the discriminator loss is:

$$\min_{D_s, D_l} L = \lambda_{\text{adv}}(L_{\text{dis}}(D_s) + L_{\text{dis}}(D_l)) \quad (9)$$

The training epoch is set to 25 and 630 for easy and hard tasks, respectively. We use an SGD optimizer with a momentum of 0.8 and batch size of 32 for training. The initial learning rate is set as 1×10^{-3} with a decaying rate of 0.999 every epoch.

5 Experiment

5.1 Evaluation

To evaluate our framework, we compare it with alternative architectures and do ablation studies for input representation. Quantitative measures are defined in terms of the level of deictic intention preserved and the naturalness of the output motion. All experiments are conducted using our in-house test dataset.

5.1.1 Deictic Intention Preservation

The criteria for assessing whether an avatar has accurately maintained a user's deictic motion are based on the CS. The avatar should ideally match the user's CS on their target. To evaluate the pointing accuracy, we adopted an angular metric proposed by [20], which is invariant to target distance. We calculated the horizontal error (HE [°]) and vertical error (VE [°]) between the EFRC vector and the vector extending from the head to the target. For the assessment of touching, we measured the positional error (PE [cm]) between the target and the right index fingertip. We used the moment t_{cs} when the user reaches CS to measure the error, utilizing the output value $\hat{\mathbf{y}}_{t_{\text{cs}}}$ of our network.

As there is no prior deep learning-based work that performs deictic motion retargeting in real-time, for comparison, we established a **MLP** model as our baseline. We replaced the MoE, which is a key component of AngleNet and MotionNet in our model (**MoE**), with MLP and removed the gating network. Additionally, we conducted an experiment comparing the proposed angle-based input representation (**angle**) with the position-rotation input representation (**pos-rot**). The pos-rot input is composed of the transformations of the head, right hand joints, and the position of the right index fingertip.

To ensure a fair comparison, we carefully matched the number of parameters in the MLP model to those in the MoE model. Specifically, for angle input, the MoE model had 42,112 parameters, closely followed by the MLP model with 42,004. Similarly, for pos-rot input, the MoE model and the MLP model had 45,952 and 45,844 parameters, respectively.

For training the MoE and MLP models on pos-rot input, we augmented the training data's pointing and touching motions with scale factors of 0.9 and 1.1. Each augmented motion was then labeled with the corresponding scale value. In both the MLP and MoE models, AngleNet received a sequence for pos-rot input, comprising the pos-rot sequence, target feature, and scale value. All other training

settings remained consistent throughout the experiment. The error rates run on the test data are presented in Table 2.

Network Architecture Comparison. Across all subjects and for both input representations, the MoE model outperforms the MLP model. When we visualized the results of the MLP model with angle input (refer to supplemental video), we noticed that the avatar mimicked the user's motion regardless of its own target position. However, the MoE model with angle input showed stable retargeting results for the avatar's randomly adjusted target position (see the supplemental video). These findings indicate that, compared to the MLP model, the MoE model more robustly retargets the user's deictic motion towards a variety of target positions.

Input Representation Comparison. Setting the input of the MoE model to angle representation resulted in smaller errors than the pos-rot representation, with the singular exception of the HE of the 172cm subject. This result suggests a tendency of our model to overfit the deictic motion of the 170cm during training with the pos-rot input. As a sensitivity test, when a user varied the orientation of the hand while pointing the same target, the model trained with pos-rot input could not stably point to the target as shown in Figure 7.This finding underscores the sensitivity of the pos-rot representation to rotational changes in the CS. Based on these results, we conclude that our angle representation is significantly more robust to hand posture variations than the pos-rot representation.

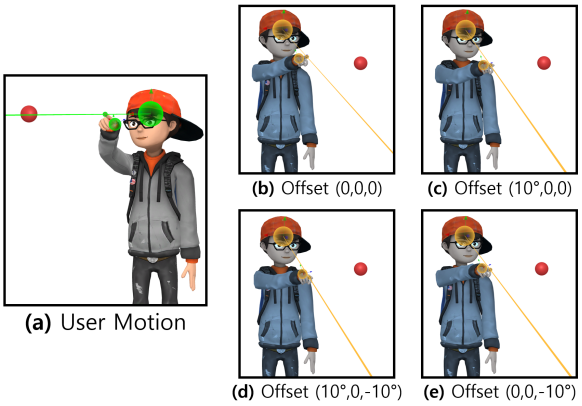


Figure 7: The retargeting results for the MoE model with pos-rot input representation. When the user rotates the hand while pointing at the same target, the network does not produce stable pointing poses. Here offset denotes the angular rotation in Euler angles.

5.1.2 Movement Naturalness

Previous studies have tackled the challenge of retargeting users' deictic movements onto avatars, primarily utilizing IK [13, 23, 29, 33]. However, IK comes with inherent limitations, especially in its ability to faithfully produce human motion dynamics. By contrast, our learning-based approach, trained on real human movements, surpasses the IK in delivering more natural movements. To validate this claim, we compared the naturalness of head and right hand joint motions produced by a consumer-grade IK [25] with those generated by our learning-based method. To generate avatar motion using IK, we utilized the angle that the user's head and right index fingertip formed with the target. We have included recordings of the avatar's motion generated by IK in the supplemental video.

Inspired by the Fréchet Inception Distance (FID) [9], a widely utilized metric in computer vision research for evaluating the fidelity and diversity of generated images, we adopted the Fréchet Motion Distance (FMD) as a quantitative metric, measuring the distance between the feature vectors of real and generated motions. A lower

Table 2: The average horizontal error (HE), vertical error (VE), and positional error (PE) produced by other methods.

Height	Method	Point		Touch
		HE↓	VE↓	PE↓
161cm	Personal offset	1.35	4.94	2.90
	MoE, angle	**1.53**	**6.56**	**8.01**
	MoE, pos-rot	2.08	14.39	10.01
	MLP, angle	26.45	15.05	27.77
	MLP, pos-rot	25.39	22.69	28.48
172cm	Personal offset	2.42	2.16	2.83
	MoE, angle	2.45	**2.12**	**4.96**
	MoE, pos-rot	**1.32**	12.20	9.67
	MLP, angle	26.50	14.23	27.74
	MLP, pos-rot	26.35	21.14	30.22
179cm	Personal offset	1.49	4.33	2.93
	MoE, angle	**2.55**	**4.41**	**7.89**
	MoE, pos-rot	5.33	11.84	12.68
	MLP, angle	26.20	14.67	28.13
	MLP, pos-rot	25.75	23.51	31.78

Table 3: Fréchet Motion Distance (FMD) produced by other methods.

Height	Method	FMD↓
161cm	MoE, angle	**5.27**
	MLP, angle	6.85
	Consumer-grade IK	30.23
172cm	MoE, angle	6.29
	MLP, angle	**6.22**
	Consumer-grade IK	32.85
179cm	MoE, angle	13.29
	MLP, angle	**13.09**
	Consumer-grade IK	21.64

FMD value indicates a smaller discrepancy between the real and generated motions, suggesting a higher naturalness of movement.

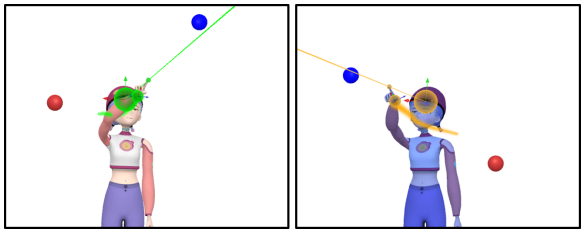
To this end, we trained a two-layer convolutional autoencoder to reconstruct the one-second (30 frames) motion of the head and right hand joints. The pointing and touching motions of a 170cm tall subject were augmented with scale factors of 0.9 and 1.1 and used as training data, while the motions of test subjects (161cm, 172cm, 179cm) were used as validation data. The average reconstruction errors for position and rotation per frame were 1.24cm and 2.16° in the training data, and 5.25cm and 10.56° in the validation data, respectively.

For all possible pairings in the test data, we segmented the motion clips produced by each method into intervals of 30 frames (1s) with a 15 frames (0.5s) overlap. In the case of pointing motions generated by our model, we adjusted the positions of the head, right hand, and right index fingertip by multiplying them with the scale value. Segmented sequences were fed into the encoder of the pre-trained convolutional autoencoder to obtain feature vectors. We calculated the FMD using these vectors.

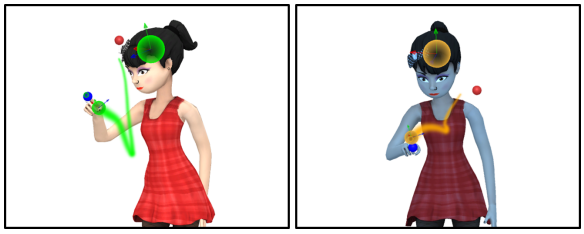
As reported in Table 3, our model (MoE) yielded lower FMD values compared to IK. This result suggests that our model can generate more natural deictic movements of avatars compared to the IK method. Interestingly, the FMD values of the MLP model are similar to those of the MoE, indicating that the MLP model provides a similar level of movement naturalness to the MoE while struggling to preserve the user's deictic intention.

5.2 AR Telepresence Application

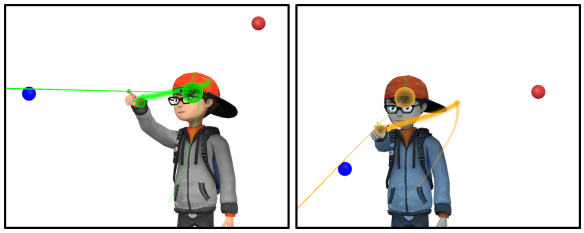
We envision our retargeting method being widely utilized in avatar-mediated AR telepresence. To demonstrate its effectiveness, we showcase two scenarios (Figure 9): education and commerce. These scenarios are further illustrated in our supplementary video.



(a) Subject with 161cm tall, pointing at two targets.



(b) Subject with 172cm tall, touching two targets.



(c) Subject with 179cm tall, pointing at two targets.

Figure 8: Deictic motions (left) from test subjects with different heights and avatar motions (right) retargeted in real-time by our framework. Two targets (red and blue spheres) are in different location configurations for a user and their corresponding avatar.

5.2.1 Implementation

Our setup consists of two identical hardware configurations, one for each remote space. Each configuration integrates a ZED mini RGB-D camera with an HTC Vive Pro headset. This combination facilitates AR rendering and enables real-time occlusion between real and virtual objects. To accurately represent the user's hand actions, we equip the user with additional Vive trackers and Noitom Hi5 VR Gloves. We implemented the entire system using the Unity3D engine (version 2020 3.9f1) and the SteamVR framework. To establish communication between the two remote systems, we used the Photon Unity Network framework. The target positions are predefined and controlled by the system operator. When target objects are switched, the target position is linearly interpolated from the original to the next target for both the user and avatar.

5.2.2 Scenarios

Education. User X teaches user Y about three planets in the solar system: Earth, Venus, and Jupiter. Each user can virtually augment these three planets in their preferred location within their respective spaces. As the user sequentially points to the planets, the movements of the avatar are made to correspondingly point at the planets in the same order.

Commerce. User X seeks user Y's opinion regarding clothing. User Y has the actual clothing item and user X has a virtual 3D replica of it (created by RECON Labs' 3Dpresso). As shown in Figure 9 (b), when the user touches a specific part of cloth (zipper in this scenario), the avatar touches the exact same part, so that the user's focus is accurately relayed.

(a) Education.

(b) Commerce.

Figure 9: Screenshots of AR telepresence scenarios.

6 Limitation and Conclusion

This section discusses several limitations of our approach. One primary limitation of this study stems from the assumption of predefined targets and known object correspondence. For seamless deictic interaction within the dynamic context of real-world AR telepresence situations in which objects can be added or removed and the number of target objects can vary in each space, the user's target and object correspondence needs to be accurately inferred in real-time, through advanced object and action recognition techniques.

Secondly, this study did not consider variations in the user's handedness and dominant eyes. Identifying the patterns associated with handedness and eye dominance can lead to developing a more comprehensive and robust solution suitable for practical applications.

Lastly, our framework needs to be evaluated for its influence on communication and social interaction. This can done through qualitative evaluation using user studies.

In conclusion, we presented a neural network-based framework that retargets deictic motion to virtual avatars for AR telepresence. Our method can retarget the deictic motions of various users to their avatars in dissimilar environments. We validated that our MoE-based architecture reliably learns deictic motion corresponding to target locations and that our angle-based representation effectively extracts user-invariant characteristics of deictic motion. Compared with IK, our learning-based method generates more natural movements of the avatar. Specifically designed for AR telepresence, the effectiveness of our framework has been demonstrated in several scenarios.

Acknowledgments

This work was supported by the NRF, Korea (NRF-2022R1A4A503368912) and KEIT, Korea (20011076). The authors wish to thank anonymous reviewers for their constructive feedback.

REFERENCES

[1] M. Azmandian, M. Hancock, H. Benko, E. Ofek, and A. D. Wilson. Haptic retargeting: Dynamic repurposing of passive haptics for enhanced virtual reality experiences. In *Proceedings of the 2016 chi conference on human factors in computing systems*, pp. 1968–1979, 2016.

[2] S. Beck, A. Kunert, A. Kulik, and B. Froehlich. Immersive group-to-group telepresence. *IEEE transactions on visualization and computer graphics*, 19(4):616–625, 2013.

[3] C. G. Fidalgo, M. Sousa, D. Mendes, R. K. dos Anjos, D. Medeiros, K. Singh, and J. Jorge. Magic: Manipulating avatars and gestures to improve remote collaboration, 2023.

[4] D. I. Fink, J. Zagermann, H. Reiterer, and H.-C. Jetter. Re-locations: Augmenting personal and shared workspaces to support remote collaboration in incongruent spaces. *Proceedings of the ACM on Human-Computer Interaction*, 6(ISS):1–30, 2022.

[5] F. N. Fritsch and J. Butland. A method for constructing local monotone piecewise cubic interpolants. *SIAM journal on scientific and statistical computing*, 5(2):300–304, 1984.

[6] E. J. Gonzalez, E. D. Chase, P. Kotipalli, and S. Follmer. A model predictive control approach for reach redirection in virtual reality. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pp. 1–15, 2022.

[7] J. E. Grønbæk, K. Pfeuffer, E. Velloso, and H. Gellersen. Partially blended realities: Aligning dissimilar spaces for distributed mixed reality meetings. 2023.

[8] F. G. Harvey, M. Yurick, D. Nowrouzezahrai, and C. Pal. Robust motion in-betweening. *ACM Transactions on Graphics (TOG)*, 39(4):60–1, 2020.

[9] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[10] R. A. Jacobs, M. I. Jordan, S. J. Nowlan, and G. E. Hinton. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87, 1991.

[11] D. Jo, K.-H. Kim, and G. J. Kim. Spacetime: adaptive control of the teleported avatar for improved ar tele-conference experience. *Computer Animation and Virtual Worlds*, 26(3-4):259–269, 2015.

[12] M. Keshavarzi, A. Y. Yang, W. Ko, and L. Caldas. Optimization and manipulation of contextual mutual spaces for multi-user virtual and augmented reality interaction. In *2020 IEEE conference on virtual reality and 3D user interfaces (VR)*, pp. 353–362. IEEE, 2020.

[13] M. Kim and S.-H. Lee. Deictic gesture retargeting for telepresence avatars in dissimilar object and user arrangements. In *The 25th International Conference on 3D Web Technology*, pp. 1–6, 2020.

[14] Y. Kim, H. Park, S. Bang, and S.-H. Lee. Retargeting human-object interaction to virtual avatars. *IEEE transactions on visualization and computer graphics*, 22(11):2405–2412, 2016.

[15] N. H. Lehment, D. Merget, and G. Rigoll. Creating automatically aligned consensus realities for ar videoconferencing. In *2014 IEEE international symposium on mixed and augmented reality (ISMAR)*, pp. 201–206. IEEE, 2014.

[16] A. Maimone and H. Fuchs. Encumbrance-free telepresence system with real-time 3d capture and display using commodity depth cameras. In *2011 10th IEEE International Symposium on Mixed and Augmented Reality*, pp. 137–146. IEEE, 2011.

[17] A. Maimone, X. Yang, N. Dierk, A. State, M. Dou, and H. Fuchs. General-purpose telepresence with head-worn optical see-through displays and projector-based lighting. In *2013 IEEE Virtual Reality (VR)*, pp. 23–26. IEEE, 2013.

[18] X. Mao, Q. Li, H. Xie, R. Y. Lau, Z. Wang, and S. Paul Smolley. Least squares generative adversarial networks. In *Proceedings of the IEEE international conference on computer vision*, pp. 2794–2802, 2017.

[19] S. Mayer, J. Reinhardt, R. Schweigert, B. Jelke, V. Schwind, K. Wolf, and N. Henze. Improving humans' ability to interpret deictic gestures in virtual reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pp. 1–14, 2020.

[20] S. Mayer, K. Wolf, S. Schneegass, and N. Henze. Modeling distant pointing for compensating systematic displacements. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pp. 4165–4168, 2015.

[21] S. Orts-Escolano, C. Rhemann, S. Fanello, W. Chang, A. Kowdle, Y. Degtyarev, D. Kim, P. L. Davidson, S. Khamis, M. Dou, et al. Holoportation: Virtual 3d teleportation in real-time. In *Proceedings of the 29th annual symposium on user interface software and technology*, pp. 741–754, 2016.

[22] T. Pejsa, J. Kantor, H. Benko, E. Ofek, and A. Wilson. Room2room: Enabling life-size telepresence in a projected augmented reality environment. In *Proceedings of the 19th ACM conference on computer-supported cooperative work & social computing*, pp. 1716–1725, 2016.

[23] T. Piumsomboon, G. A. Lee, J. D. Hart, B. Ens, R. W. Lindeman, B. H. Thomas, and M. Billinghurst. Mini-me: An adaptive avatar for mixed reality remote collaboration. In *Proceedings of the 2018 CHI conference on human factors in computing systems*, pp. 1–13, 2018.

[24] K. Plaumann, M. Weing, C. Winkler, M. Müller, and E. Rukzio. Towards accurate cursorless pointing: the effects of ocular dominance and handedness. *Personal and Ubiquitous Computing*, 22:633–646, 2018.

[25] Root-Motion. Final-ik. 2017.

[26] M. Sousa, R. K. dos Anjos, D. Mendes, M. Billinghurst, and J. Jorge. Warping deixis: distorting gestures to enhance collaboration. In *Proceedings of the 2019 CHI Conference on Human Factors in Computing Systems*, pp. 1–12, 2019.

[27] A. Steed, W. Steptoe, W. Oyekoya, F. Pece, T. Weyrich, J. Kautz, D. Friedman, A. Peer, M. Solazzi, F. Tecchia, et al. Beaming: an asymmetric telepresence system. *IEEE computer graphics and applications*, 32(6):10–17, 2012.

[28] B. Thoravi Kumaravel, F. Anderson, G. Fitzmaurice, B. Hartmann, and T. Grossman. Loki: Facilitating remote instruction of physical tasks using bi-directional mixed-reality telepresence. In *Proceedings of the 32nd Annual ACM Symposium on User Interface Software and Technology*, pp. 161–174, 2019.

[29] A. Ullal, A. Watkins, and N. Sarkar. A multi-objective optimization framework for redirecting pointing gestures in remote-local mixed/augmented reality. In *Proceedings of the 2022 ACM Symposium on Spatial User Interaction*, pp. 1–11, 2022.

[30] X. Wang, H. Ye, C. Sandor, W. Zhang, and H. Fu. Predict-and-drive: Avatar motion adaption in room-scale augmented reality telepresence with heterogeneous spaces. *IEEE Transactions on Visualization and Computer Graphics*, 28(11):3705–3714, 2022.

[31] N. Wong and C. Gutwin. Where are you pointing? the accuracy of deictic pointing in cves. In *Proceedings of the sigchi conference on human factors in computing systems*, pp. 1029–1038, 2010.

[32] N. Wong and C. Gutwin. Support for deictic pointing in cves: still fragmented after all these years'. In *Proceedings of the 17th ACM conference on Computer supported cooperative work & social computing*, pp. 1377–1387, 2014.

[33] L. Yoon, D. Yang, C. Chung, and S.-H. Lee. A mixed reality telepresence system for dissimilar spaces using full-body avatar. In *SIGGRAPH Asia 2020 XR*, pp. 1–2. 2020.

[34] L. Yoon, D. Yang, J. Kim, C. Chung, and S.-H. Lee. Placement retargeting of virtual avatars to dissimilar indoor environments. *IEEE Transactions on Visualization and Computer Graphics*, 28(3):1619–1633, 2020.

[35] Y. Zhou, C. Barnes, J. Lu, J. Yang, and H. Li. On the continuity of rotation representations in neural networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 5745–5753, 2019.