






# Online Avatar Motion Adaptation to Morphologically-similar Spaces

Soojin Choi<sup>1</sup> , Seokpyo Hong<sup>2</sup> , Kyungmin Cho<sup>3</sup> , Chaelin Kim<sup>1</sup> , Junyong Noh<sup>1</sup> <sup>1</sup>Visual Media Lab, KAIST<sup>2</sup>Samsung Advanced Institute of Technology<sup>3</sup>Anigma Technologies

## Abstract

In avatar-mediated telepresence systems, a similar environment is assumed for involved spaces, so that the avatar in a remote space can imitate the user's motion with proper semantic intention performed in a local space. For example, touching on the desk by the user should be reproduced by the avatar in the remote space to correctly convey the intended meaning. It is unlikely, however, that the two involved physical spaces are exactly the same in terms of the size of the room or the locations of the placed objects. Therefore, a naive mapping of the user's joint motion to the avatar will not create the semantically correct motion of the avatar in relation to the remote environment. Existing studies have addressed the problem of retargeting human motions to an avatar for telepresence applications. Few studies, however, have focused on retargeting continuous full-body motions such as locomotion and object interaction motions in a unified manner. In this paper, we propose a novel motion adaptation method that allows to generate the full-body motions of a human-like avatar on-the-fly in the remote space. The proposed method handles locomotion and object interaction motions as well as smooth transitions between them according to given user actions under the condition of a bijective environment mapping between morphologically-similar spaces. Our experiments show the effectiveness of the proposed method in generating plausible and semantically correct full-body motions of an avatar in room-scale space.

## CCS Concepts

- **Computing methodologies** → **Animation**;

## 1. Introduction

In avatar-mediated telepresence systems, a human-like avatar displayed in a remote space represents the user in a local space. By controlling the avatar, the user can effectively deliver the intention or semantic meaning of their motions to the telepresence participants in a remote space. For example, the local user can make physical contacts with remote participants [OLK\*16, LLL17], point to shared objects [KL20, YYCL21], or interact with environmental elements such as floor or objects [JKK15, KPBL16].

In a typical telepresence setup, one can decide on the type and number of objects as well as the approximate size of the room to ensure a similar environment for involved spaces. However, it is unlikely that the shapes of the objects, their locations, and the size of the rooms are exactly the same. This will create difficulties in the delivery of the semantically correct meaning of the local user's motion to the remote space. For example, an avatar naively imitating the user motion may collide with an object instead of passing it by due to the size and orientation differences. To avoid this, the user motion should be adapted in consideration of the environmental differences between the two spaces when it is transferred to an avatar.

To reproduce the user motion to a remote avatar in a semantically correct way, the following three issues need to be addressed.

First, when the user moves to a different location, the strides and root path should be modified in the remote space according to the environment differences within the same duration, to prevent the avatar from colliding with objects. Second, the action intended by the user should be achieved by the avatar in the same way. For example, if the user puts their hand on a desk, the avatar should do the same despite the height difference of the desks in the two different spaces. Third, the aforementioned issues should be handled on-the-fly for real-time tele-communication, which is fundamentally different from the approaches that assume the availability of the motion in advance.

In the field of telepresence research, there are methods that re-target human motions to an avatar with promising results. Except for Kim et al. [JKK15], none of these approaches address continuous full-body motions and instead focus only on a limited aspect of human motion. For example, several approaches deal with the modification of the user root path during locomotion without supporting the user's interaction with objects [CWS18, TSDS19] or only support human-object interaction motions within a pre-defined area around an object [KPBL16]. Other approaches propose placement retargeting methods for an avatar [YYK\*20] with upper-body gestures only [KL20, YYCL21]. Despite their reported successes, these approaches are not effective in providing *co-presence* to par-

participants in a telepresence system due to the lack of general handling of continuous full-body motions including both locomotion and object interaction motions.

In this paper, we propose a novel motion adaptation method that allows to generate the continuous full-body motion of an avatar, whose size and skeletal structure are the same as those of the user. We assume that the local and remote spaces have morphologically-similar layouts in which the sizes of the spaces and the locations and orientations of the objects can be different to some degree. The reason for our assumption is that retargeting the motion to an avatar in a completely-different environments would be inevitably ill-posed. Under this assumption, our method first modifies the user root path and joint trajectories fitting into the remote space via a thin plate spline (TPS)-driven [Boo89] bijective environment mapping. According to the modified path and foot trajectories, the user strides are properly adjusted. When the avatar approaches an object, it can make a smooth transition from locomotion to object interaction motions by an importance-based approach [SLSG01], followed by proper interaction with the object based on the modified joint trajectories.

The contributions of this paper are three folds: First, our motion adaptation method generates human-like avatar motions on-the-fly by taking a user motion as streaming input. Second, our TPS-based bijective environmental mapping between 3D spaces effectively modifies the user's motions to make the resulting avatar motion semantically correct in morphologically-similar environments. Finally, our method considers locomotions, object interaction motions, and smooth transitions between them in a unified manner using an importance-based approach.

## 2. Related Work

### 2.1. Telepresence using a Full-body Avatar

As a human-like avatar is a useful medium to represent the user in a distant place [KPBL16], much effort have been made to use a 3D avatar in telepresence applications for the past decade. Holoportation [OERF\*16] is an end-to-end telepresence system that represents and transmits the user as an avatar while allowing the avatar to freely move and interact in a remote space. This study focused on reconstructing the user in a remote space with an identical configuration, ignoring the problems derived from the environmental difference between the two spaces.

In practice, because the two spaces involved in a remote communication usually have different environments, many approaches have addressed the problem of adapting the human motion to an avatar resolving environmental differences. In order to ensure that the avatar can interact with an object whose shape is different from that of the object in the user space, Kim et al. [KPBL16] introduced a spatial map that defines the correspondences between the two objects and their surrounding areas. Although the use of spatial map allows to generate avatar motions that are faithful to the original spatial relationship, the user-object interaction is handled only in a pre-defined area around the object.

To retarget a user's deictic gestures to an avatar, gesture retargeting approaches have been proposed [KL20, YYCL21]. The user's

head direction and arm posture are adjusted to make the avatar point to the same position toward the remote object as that toward the corresponding local object. Focusing on the upper-body motion, these studies enable the local user to interact with a remote participant using an avatar and shared objects.

Few approaches have addressed the problem of generating full-body avatar motions including locomotion and object interaction motions altogether for telepresence applications. Jo et al. [JKK15] matched the objects in two different spaces automatically [ATCO\*10], and used the information to define the correspondences for the entire space so that the avatar can be placed at the position corresponding to the user. The pose of the avatar is then adjusted considering the spatial relationship with a remote object [AAKC13]. This is the first work that addresses locomotion and object interaction motions at the same time while resolving environmental differences. Different from this approach, our motion adaptation technique ensures no collision of the avatar with objects while moving and supports proper contacts with interacting objects.

### 2.2. Placement of a Teleported Avatar

When teleporting a user to a remote space as an avatar, deciding where to place the avatar is challenging because the environmental difference between the two spaces can cause unexpected avatar-object penetrations in the remote space. Several studies have suggested solutions to avoid such artifacts by placing the avatar in pre-designated spots such as on a sofa [PKB\*16] or near a table [MYD\*13]. Other studies dedicated a common empty space to which participants can access from multiple dissimilar spaces [LMR14, KYKC20, KZY\*22]. Unlike these approaches, we utilize the entire space for the avatar to freely interact with environmental objects.

To use the entire space for remote communication, some studies applied a bijective mapping between two heterogeneous spaces. Jo et al. [JKK15] created an affine transformation-based mapping function between two spaces by automatically matching objects from each space. Congdon et al. [CWS18] similarly created a mapping function between a pair of spaces to enable users in different physically-walkable spaces to interact with each other in a virtual reality environment. Tomar et al. [TSDS19] presented a pipeline that deforms a local space to fit a remote space using a conformal mapping that minimizes shape distortions. Unlike these approaches that assume a similar layout between two spaces, Yoon et al. [YYK\*20] presented a learning-based approach that allows to cope with large environmental differences by parameterizing the semantic relationship between two spaces. To handle differences in the environment, they move the avatar only after the user finishes locomotion, making the avatar constantly jump to the correct user location. Such abrupt changes in the avatar motion have empirically proven to adversely affect the user's experience [WYS\*22].

These studies suggest that generating continuous avatar movement in largely dissimilar telepresence environments is ill-posed. Therefore, we set the two indoor spaces to have morphologically-similar layouts and focus on generating continuous avatar motions while reducing the artifacts that arise when modifying the global trajectories of full-body motions such as foot-sliding.



### 2.3. Motion Adaptation with Spatial Relationship

Early studies on motion adaptation have focused on retargeting the motion of a character to another with a different size, proportion [CK00, SLSG01], or body shape [LMT08]. To handle interaction with geometrically changing entities such as objects or other characters, Ho et al. [HKT10] proposed to use an *interaction mesh*, which deforms to adapt the motion of an articulated character according to the changed morphology of the interactable entity. Similarly, Al-Asqhar et al. [AAKC13] presented *relationship descriptor*, a static set of points sampled on the surface of an interactable entity. The character motion for the deformed geometry is then reproduced using the relative translation between the body parts and descriptor points. Unfortunately, these methods are not appropriate for telepresence applications because a full sequence of the original motion is required in advance.

There are approaches that enable motion adaptation in real-time scenarios. Using a pre-defined spatial map between two objects, Kim et al. [KPBL16] specified the desired joint trajectories of an avatar from the given stream of user motions. The use of such joint trajectories helps preserve the spatial relationship between the user and an object, thereby making the avatar properly interact with a differently shaped object. Jin et al. [JKL18] suggested an approach that takes advantage of the skin-level spatial relationship using an *aura mesh* to retarget interaction motions to characters with various body shapes. These studies share the same goal with us in that close interaction motions are adapted to differently shaped interactable entities in real-time. However, in room-scale telepresence environments, motions that take place far from interactable entities should also be adapted. We demonstrate how locomotion toward an object, interaction motions with it, and smooth transition between the motions can be adapted to an avatar in a unified manner.

Several learning-based approaches have focused on generating scene-aware motions including locomotion as well as object interaction motions. *Neural State Machine (NSM)* [SZKS19] allows to generate smooth transitions among various object-interaction motions by utilizing a volumetric representation around the character. Inspired by *NSM*, Hassan et al. [HCV\*21] mainly deal with the character motions of sitting and lying down through a scene-aware motion prediction network (*SAMP*). Unlike these approaches whose focus is on the creation of plausible object interaction motions according to a given target action and goal, the main goal of our approach is to preserve the semantics of the user motions when it is reproduced at a remote space, given a streaming input without any pre-defined actions or goals.

### 3. System Overview

An overview of the proposed method is shown in Figure 1. We assume that the two spaces utilized in a remote communication share morphologically-similar layouts in which the size of the spaces and the location and orientation of the objects can be different to some degree. We also assume that the shapes of the floors are rectangular and flat. Therefore, there always exists a corresponding remote object to a local one. Under these assumptions, our online motion adaptation scheme consists of two stages: an offline stage that defines correspondences between the two spaces, and an online stage

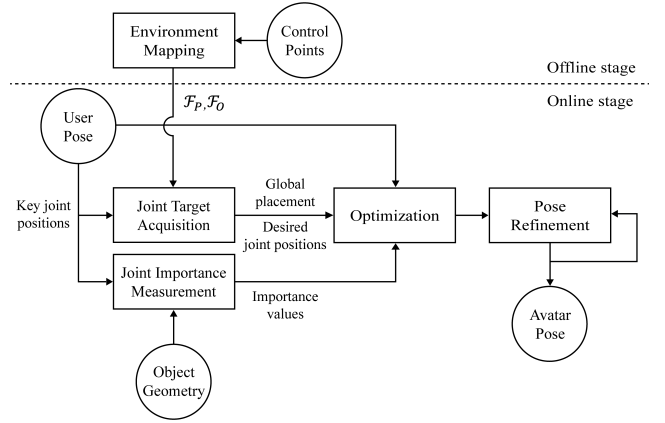


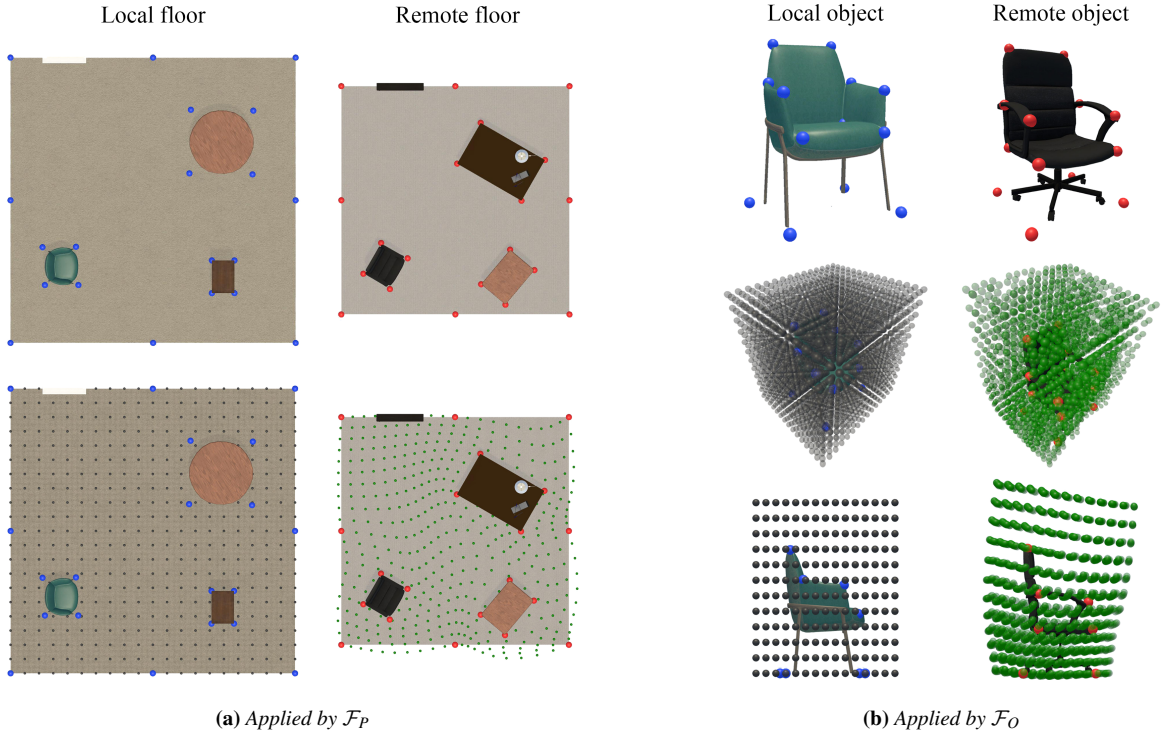
Figure 1: System overview

that generates a full-body motion of the avatar from the user motion on-the-fly. In the offline stage, we manually specify a set of corresponding control points on the surface of the environmental elements in each space. We then pre-define a mapping for each paired object,  $\mathcal{F}_{O_l} : {}^1O_l \rightarrow {}^2O_l$  in the form of TPS-driven bijective mapping using the correspondence points. Here,  $l$  denotes the index for each paired-object and the superscript on the left of the symbol indicates each space. In addition, we also create a mapping for floor  $\mathcal{F}_P : {}^1P \rightarrow {}^2P$ , ignoring the height of the floor in this case.

In the online stage, given the user's pose as streaming input, we determine the desired root placement, including its position and orientation, of the avatar by mapping the corresponding information of the user via  $\mathcal{F}_P$ . To reflect the interaction context to the avatar when the user interacts with an object, we specify the desired positions of the key joints, root and end-effectors, of the avatar. The positions are obtained by mapping the user's key joints to the remote space using  $\mathcal{F}_{O_c}$ , where  $c$  refers to the closest object from the user in a local space. We then generate the pose of the avatar by optimizing the user's pose to satisfy the desired joint constraints while preserving plausible human motion. Using the method introduced in Shin et al. [SLSG01], we measure the interaction probability of each key joint to judge whether the user is interacting with an object or not in a continuous manner. Using the interaction probability values, we dynamically adjust the importance of each term involved in the optimization process, consequently making smooth transitions. Finally, after the pose refinement is performed, the resulting motion is applied to the avatar. The overall online process is performed in real-time.

### 4. TPS-driven Environment Mapping

In the offline stage, we first manually specify a set of control points on environmental elements, such as floors, tables, or chairs in the two spaces as illustrated in Figure 2. We then define the points on the local and remote spaces as sets of source control points  $\mathbf{S} = (\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_{N_C})$  and target control points  $\mathbf{T} = (\mathbf{t}_1, \mathbf{t}_2, \dots, \mathbf{t}_{N_C})$ , respectively, where  $N_C$  is the number of control points. Note that the two sets should be mathematically bijective.



**Figure 2:** Examples of control points and space deformations produced by TPS mappings. Blue and red spheres represent a set of source and target control points, respectively. In order to visualize how each space is deformed, we put regularly-distributed auxiliary gray spheres on a source floor (a) and around a source 3D object (b). Each space is warped by the mapping as shown by green spheres.

Using the paired points, we can define a bijective environment mapping interpolation function that works in 3D domain. This function is based on thin plate spline (TPS) transformation [Boo89], which is known to perform smooth 2D interpolation among two sets of corresponding control points in image domain [RAS17]. To formulate the mapping function, we calculate a set of corresponding weights  $\mathbf{W}$  between the sets  $\mathbf{S}$  and  $\mathbf{T}$  by extending the work of Keller and Borkowski [KB19] in 3D-form as follows:

$$\begin{bmatrix} \alpha & r_{1,2} & \cdots & r_{1,N_C} & 1 & \mathbf{s}_1^T & \\ r_{2,1} & \alpha & \cdots & r_{2,N_C} & 1 & \mathbf{s}_2^T & \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots & \\ r_{N_C,1} & r_{N_C,2} & \cdots & \alpha & 1 & \mathbf{s}_{N_C}^T & \\ 1 & 1 & \cdots & 1 & 0 & \mathbf{0}_{1 \times 3} & \\ \mathbf{s}_1 & \mathbf{s}_2 & \cdots & \mathbf{s}_{N_C} & \mathbf{0}_{3 \times 1} & \mathbf{0}_{3 \times 3} & \end{bmatrix} \begin{bmatrix} \mathbf{w}_1^T \\ \mathbf{w}_2^T \\ \vdots \\ \mathbf{w}_{N_C}^T \\ \mathbf{w}_{N_C+1}^T \\ \mathbf{w}_{N_C+2}^T \\ \mathbf{w}_{N_C+3}^T \\ \mathbf{w}_{N_C+4}^T \end{bmatrix} = \begin{bmatrix} \mathbf{t}_1^T \\ \mathbf{t}_2^T \\ \vdots \\ \mathbf{t}_{N_C}^T \\ \mathbf{0}_{1 \times 3} \\ \mathbf{0}_{1 \times 3} \\ \mathbf{0}_{1 \times 3} \\ \mathbf{0}_{1 \times 3} \end{bmatrix}, \quad (1)$$

where  $\mathbf{s}_i = [x_i, y_i, z_i]^T \in \mathbf{S}$  is the  $i$ -th source control point, and  $\mathbf{t}_i = [x'_i, y'_i, z'_i]^T \in \mathbf{T}$  is the corresponding target control point. When computing a mapping between floors, the values of  $y_i$  and  $y'_i$  are set to zero to ignore the height of the floors.  $r_{i,j}$  is obtained from two source control points, where  $r_{i,j} = \frac{\|\mathbf{s}_i - \mathbf{s}_j\|^2 \ln \|\mathbf{s}_i - \mathbf{s}_j\|}{\|\mathbf{s}_i - \mathbf{s}_j\|^2 \ln \|\mathbf{s}_i - \mathbf{s}_j\|}$ . We set the smoothing parameter  $\alpha$  to 0 in order to satisfy an interpolation spline [KB19]. Then  $\mathbf{W} = [\mathbf{w}_1^T \mathbf{w}_2^T \cdots \mathbf{w}_{N_C+3}^T \mathbf{w}_{N_C+4}^T]^T$ , where  $\mathbf{w}_i \in \mathbb{R}^{3 \times 1}$ , can be obtained by solving the linear equation, Eq. (1).

We refer the readers to the supplementary document for more details of Eq. (1).

Using  $\mathbf{W}^k$ , where  $k$  indicates either the planar floor( $P$ ) or a 3D object( $O$ ), we can finally define the environment mapping function  $\mathcal{F}_k$  for an arbitrary point  $\mathbf{p} \in \mathbb{R}^{3 \times 1}$  as follows:

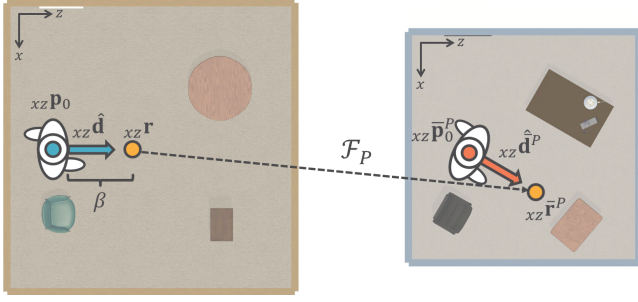
$$\mathcal{F}_k(\mathbf{p}) = \bar{\mathbf{p}}^k, \quad k \in \{O, P\}, \quad (2)$$

$$[r_1 \quad r_2 \quad \cdots \quad r_{N_C} \quad 1 \quad \mathbf{p}^T] \mathbf{W}^k = \bar{\mathbf{p}}^{kT},$$

where  $r_i = \frac{\|\mathbf{s}_i - \mathbf{p}\|^2 \ln \|\mathbf{s}_i - \mathbf{p}\|}{\|\mathbf{s}_i - \mathbf{p}\|^2 \ln \|\mathbf{s}_i - \mathbf{p}\|}$  and  $\bar{\mathbf{p}}^k \in \mathbb{R}^{3 \times 1}$  is the mapped point. Note that the bar placed on the top of each notation represents the value defined in the remote space for the avatar. We refer the readers to the previous work [Boo89, KB19] for more details of TPS transformation.

## 5. Acquisition of Target Joints

We use the environment mapping functions to specify the desired target joint positions of the avatar in the remote space. This process is performed in two steps. We first acquire the desired global placement which includes the position and orientation of the avatar in the planar view point as shown in Figure 3. We then obtain the end-effector positions that represent the target limb postures of the avatar as well as the root joint position with respect to the object.



**Figure 3:** Adaption of the position and orientation of the avatar according to the difference in the given spaces. Please see Section 5.1 for the definitions of the symbols used in the illustration.

### 5.1. Global Placement Determination

The avatar should be placed and oriented in the remote space in the same position and orientation in which the user is placed with respect to the local space before the motion is applied to the avatar, considering the environmental differences between the two spaces as depicted in Figure 3.

**Desired root position** To determine the root position of the avatar  $\bar{\mathbf{p}}_0^P \in \mathbb{R}^3$ , we first project the user's root position onto the planar floor. Superscript  $P$  denotes a space of the planar floor. We then find the corresponding position in the remote space using the pre-computed environment mapping function  $\mathcal{F}_P$  as follows:

$${}_{xz}\bar{\mathbf{p}}_0^P = \mathcal{F}_P({}_{xz}\mathbf{p}_0), \quad (3)$$

where  ${}_{xz}\mathbf{p}_0 \in \mathbb{R}^3$  denotes the user's root position projected onto the ground in y-up coordinate and  ${}_{xz}\bar{\mathbf{p}}_0^P \in \mathbb{R}^3$  denotes the corresponding position to  ${}_{xz}\mathbf{p}_0$  on the remote plane specified by  $\mathcal{F}_P$ . Finally, assuming that the avatar and the user are of the same size, we replace the height value of  ${}_{xz}\bar{\mathbf{p}}_0^P$  with that of the local user.

**Desired root orientation** To obtain the root orientation of the avatar  $\bar{\mathbf{q}}_0^P \in \mathbb{S}^3$ , we first compute the normalized forward direction of the user  ${}_{xz}\hat{\mathbf{d}} \in \mathbb{R}^3$ . The hat indicates that the vector is normalized. As depicted in Figure 3, we then compute reference point  ${}_{xz}\mathbf{r} \in \mathbb{R}^3$  on the local plane as follows:

$${}_{xz}\mathbf{r} = {}_{xz}\mathbf{p}_0 + \beta {}_{xz}\hat{\mathbf{d}}. \quad (4)$$

$\beta$  is a scaling factor that changes the distance between the user and the reference point  ${}_{xz}\mathbf{r}$ . We use the value of 0.2 for  $\beta$  in all of our experiments. We will show how the results change according to the choice of  $\beta$  in Section 9.4. The forward direction of the avatar  ${}_{xz}\bar{\mathbf{d}}^P \in \mathbb{R}^3$  is determined next using the reference point  ${}_{xz}\mathbf{r}$ :

$$\begin{aligned} {}_{xz}\bar{\mathbf{r}}^P &= \mathcal{F}_P({}_{xz}\mathbf{r}), \\ {}_{xz}\bar{\mathbf{d}}^P &= {}_{xz}\bar{\mathbf{r}}^P - {}_{xz}\bar{\mathbf{p}}_0^P. \end{aligned} \quad (5)$$

Finally,  $\bar{\mathbf{q}}_0^P$  is obtained as follows:

$$\bar{\mathbf{q}}_0^P = \exp(\mathbf{v})\mathbf{q}_0, \quad (6)$$

where  $\mathbf{q}_0$  denotes the root orientation of the user and  $\exp(\mathbf{v})$  is the three-dimensional rotation that aligns  ${}_{xz}\hat{\mathbf{d}}$  with  ${}_{xz}\bar{\mathbf{d}}^P$ , which has the

following axis  $\frac{\mathbf{v}}{\|\mathbf{v}\|} \in \mathbb{R}^3$  and angle  $\|\mathbf{v}\| \in \mathbb{R}$ :

$$\begin{aligned} \frac{\mathbf{v}}{\|\mathbf{v}\|} &= {}_{xz}\hat{\mathbf{d}} \times {}_{xz}\hat{\mathbf{d}}^P, \\ \text{where } \|\mathbf{v}\| &= \cos^{-1} \left( \frac{{}_{xz}\hat{\mathbf{d}} \cdot {}_{xz}\hat{\mathbf{d}}^P}{\|{}_{xz}\hat{\mathbf{d}}\| \|{}_{xz}\hat{\mathbf{d}}^P\|} \right). \end{aligned} \quad (7)$$

### 5.2. Desired Joint Positions Determination

When a user interacts with an object, the spatial relationship between the user and object in the local space should be preserved in the remote space in which the avatar performs the same interaction. Given the user's  $j_{key}$ -th joint position  $\mathbf{p}_{j_{key}}$ , where  $j_{key} \in \{\text{root, left hand, right hand, left foot, right foot}\}$  is an index of key joints, we compute the desired joint position of the avatar  $\bar{\mathbf{p}}_{j_{key}}^{O_c}$  using the environment mapping as follows:

$$\bar{\mathbf{p}}_{j_{key}}^{O_c} = \mathcal{F}_{O_c}(\mathbf{p}_{j_{key}}), \quad (8)$$

where  $c$  denotes an index of the closest local object from the user. Note that there are two distinct mapping functions for the root;  $\mathcal{F}_P$  that is responsible for handling locomotion through planar warping and  $\mathcal{F}_O$  that is responsible for object interaction motions by deforming 3D space. The root joint positions computed by the two different mapping functions should be adaptively interpolated depending on the context of user motions. We explain the details in Section 7.1.

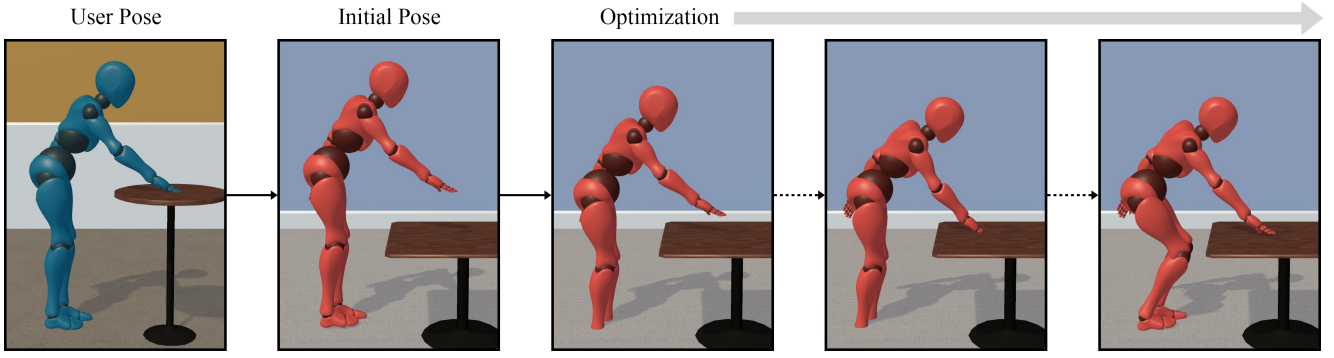
### 6. Joint Importance Measurement

We apply an importance analysis technique [SLSG01] in order to measure object interaction possibility of each body part. Based on the possibility, we infer if the body part is in interaction status. Shin et al. [SLSG01] formulated the importance value to be inversely proportional to the Euclidean distance between the joint and the nearest object. For the estimation of the distance between a user joint and an object, we use a signed distance function (SDF), which computes the shortest distance from the surface of the object. For the details on the computation of the importance value, we refer the readers to Shin et al. [SLSG01].

The measured importance values for the user's key joints, root and end-effectors joints, with respect to the closest local object are represented as follows:

$$\Omega = \{\omega_{root}, \omega_{lh}, \omega_{rh}, \omega_{lf}, \omega_{rf}\}, \quad (9)$$

where subscripts  $lh, rh, lf$ , and  $rf$  stand for the left hand, right hand, left foot, and right foot, respectively. The values are then integrated to the optimization process to dynamically adjust the influence of each objective term to be applied to each key joint according to the user motion. For instance, when the user approaches a chair to sit on while putting hands on the armrests, the root and hand joints are highly influenced by the object interaction relevant terms while the foot joints are barely influenced by those terms. Because the importance value continuously changes within 0 to 1, smooth transitions are generated while being adjusted.



**Figure 4:** Input user motion and procedural motion update. Each image containing the red avatar respectively depicts the initial pose and updated pose after the computation of the root position, spine posture, and limb postures with pose refinement applied.

## 7. Optimization

Following the work of Shin et al. [SLSG01], we divide the entire optimization process into three sub-problems: the computation of the root position, spine posture, and limb postures. This *coarse-to-fine* optimization strategy effectively finds the solutions for each body part in relation to individual objectives. We optimize the displacement map [LS99] of full-body joints at time step  $t$ , which is normally defined as follows:

$$\mathbf{x}^t = (\mathbf{u}_0^t, \mathbf{v}_0^t, \mathbf{v}_1^t, \dots, \mathbf{v}_{N_J-1}^t), \quad (10)$$

where  $\mathbf{u}_0^t \in \mathbb{R}^3$  and  $\mathbf{v}_0^t \in \mathbb{R}^3$  are the translational and rotational displacement of the root joint, respectively.  $\mathbf{v}_j^t \in \mathbb{R}^3$  denotes the rotational displacement of the  $j$ -th joint and  $N_J$  is the number of joints excluding end-effector joints because we do not consider the orientation updates of end-effectors. User motion  $\mathcal{M}_{src}^t = (\mathbf{p}_0^t, \mathbf{q}_0^t, \dots, \mathbf{p}_{N_J-1}^t)$  given in the local space is partially updated at each step in the optimization through a component level operation  $\mathcal{M}_{src}^t \oplus \mathbf{x}$ , as described below:

$$\begin{aligned} \mathbf{p}_0^{t'} &= \mathbf{p}_0^t + \mathbf{u}_0^t, \\ \mathbf{q}_j^{t'} &= \mathbf{q}_j^t \exp(\mathbf{v}_j^t) \quad (0 \leq j < N_J). \end{aligned} \quad (11)$$

To effectively solve our procedural optimization, we rearrange the elements of  $\mathbf{x}$  by using an element shuffle function  $\mathcal{S}$  defined below:

$$\mathcal{S}(\mathbf{x}) = (\mathbf{x}_R^t, \mathbf{x}_S^t, \mathbf{x}_L^t), \quad (12)$$

where  $\mathbf{x}_R^t \in \mathbb{R}^{N_R}$ ,  $\mathbf{x}_S^t \in \mathbb{R}^{N_S}$ , and  $\mathbf{x}_L^t \in \mathbb{R}^{N_L}$  are the translational displacement of the root joint, the rotational displacements of the spine joints including the root joint, and the rotational displacements of remaining limb segments, respectively. Here,  $N_R + N_S + N_L = 3 \times (N_J + 1)$ . For more details of how to use  $\mathcal{S}(\mathbf{x})$ , please see the supplementary document. For simplicity, we will leave out  $t$  in the following paragraphs.

The optimization is performed at every time step  $t$  in order to update the motion to the next state. At the very beginning of each step, we set an initial guess  $\mathcal{M}_{init}$  to  $\mathcal{M}_{src} \oplus (\bar{\mathbf{p}}_0^P - \mathbf{p}_0, \ln(\bar{\mathbf{q}}_0^P \mathbf{q}_0^{-1}), \mathbf{0}_{1 \times (3 \times (N_J - 1))})$ , which represents the user pose whose root position and orientation are transformed as described

in Section 5.1. Starting from the initial pose, the remaining joints are sequentially updated using the displacement as depicted in Figure 4.

### 7.1. Root Position Computation

Because the root orientation is highly correlated with a upper body posture, we first deal with the root position only. In order to place the avatar where it preserves the semantic meaning of the user motion, we define three terms: global positioning ( $E_g$ ), object interaction ( $E_o$ ), and reachability check ( $E_r$ ). The objective function for the root position computation is as follows:

$$\arg \min_{\mathbf{x}_R} \lambda_g E_g + \lambda_o E_o + \lambda_r E_r, \quad (13)$$

where  $\mathbf{x}_R = \mathbf{u}_0$ , and the weights  $\lambda$  determine the relative importance of each term. Note that we set  $\lambda_g$  and  $\lambda_o$  to the same value because  $E_g$  and  $E_o$  are linearly complementary to each other as explained in the following.

**Global positioning ( $E_g$ )** We define the global positioning term as follows:

$$E_g = (1.0 - \omega_{root}) \|(\mathbf{p}_0 + \mathbf{u}_0) - \bar{\mathbf{p}}_0^P\|^2, \quad (14)$$

where  $\bar{\mathbf{p}}_0^P$  denotes the desired root position obtained in Section 5.1 and  $\omega_{root} \in \Omega$  is the importance value for the root.  $E_g$  is responsible for penalizing optimization variable  $\mathbf{x}_R$  when the user only performs locomotions by setting  $\omega_{root} = 0$  and also for interpolating the locomotion with object interaction motions when the user starts interacting with an object.

**Object interaction ( $E_o$ )** This term forces the user root joint position to be adjusted so that the avatar properly interacts with a remote object by following the desired root position  $\bar{\mathbf{p}}_0^O$  computed in Section 5.2:

$$E_o = \omega_{root} \|(\mathbf{p}_0 + \mathbf{u}_0) - \bar{\mathbf{p}}_0^O\|^2. \quad (15)$$

When the user approaches an object, the value of  $\omega_{root}$  becomes higher making the avatar motion gradually transition from locomotion to avatar-object interaction motion by interpolating between  $E_g$  and  $E_o$ . When the user moves away from an object, the same interpolation occurs in the opposite direction.



**Reachability check ( $E_r$ )** In order to place the root so that each end-effector can meet each target, the reachability check term examines if each target is reachable by the straightened limb from the current root position [SLSG01]. We first specify the index sets as follows:

$$\begin{aligned} \mathcal{A} &= \{m = (J_{ball}, e) | m \in \{(ls, lh), (rs, rh)\}\}, \\ \mathcal{L} &= \{n = (J_{ball}, e) | n \in \{(lc, lf), (rc, rf)\}\}, \end{aligned} \quad (16)$$

where both  $m$  and  $n$  consist of indices of the ball joint and the end-effector of the arm ( $\mathcal{A}$ ) and leg ( $\mathcal{L}$ ) sets, respectively. Similar to subscripts  $lh, rh, lf$ , and  $rf$  that we defined in Section 6, subscripts  $ls, rs, lc$ , and  $rc$  stand for the left and right sides of the shoulder and coxa of the avatar. We examine if the end-effector of the avatar can reach the goal position from the given ball joint position by employing the function  $R^k$  [SLSG01]:

$$R^k(J_{ball}, e) = \begin{cases} 0, & \text{if } \|\mathbf{p}_{J_{ball}} - \bar{\mathbf{p}}_e^k\| < l_e \\ (\|\mathbf{p}_{J_{ball}} - \bar{\mathbf{p}}_e^k\| - l_e)^2, & \text{otherwise,} \end{cases} \quad (17)$$

where  $l_e$  is the maximum length of each limb, and  $k \in \{O, P\}$  denotes the index of the space of mapping function. Using the forms of  $R(m)$  and  $R(n)$ , we define the reachability check term as follows:

$$E_r = \sum_{m \in \mathcal{A}} \omega_{e \in m} R^O(m) + \sum_{n \in \mathcal{L}} (\omega_{e \in n} R^O(n) + (1 - \omega_{e \in n}) R^P(n)). \quad (18)$$

We apply  $\omega_e$ , the importance value of each end-effector, to make an end-effector with higher importance strongly forced to reach its goal, compared to the others. Note that the reachability check term is explicitly expressed by target joint positions, but implicitly optimized because all target joints can be expressed with the root by using forward kinematics (FK) along the joint hierarchy. In this step, we set  $\lambda_g, \lambda_o$ , and  $\lambda_r$  as 1.5, 1.5, and 1.0, respectively.

## 7.2. Spine Posture Computation

After the root position is determined, we adjust the spine posture of the avatar to make its upper end-effectors reach the target while preserving a natural human pose. To achieve these objectives, we define two terms:

$$\arg \min_{\mathbf{x}_S} \lambda_S E_S + \lambda_P E_P, \quad (19)$$

where  $\mathbf{x}_S = \{\mathbf{v}_0, \mathbf{v}_1, \dots, \mathbf{v}_{N_S}\}$  is a set of rotational displacements involved in the spine chain starting from the root joint.  $\mathbf{v}_0$  is the rotational displacement of the root,  $\mathbf{v}_h$  ( $1 \leq h \leq N_S$ ) is the rotational displacements of the joints involved in the chain.  $N_S = 3$  in our case.

**Spine refinement ( $E_S$ )** Given the root position, the upper body posture of the avatar should be adjusted if the hand joint cannot reach its goal. This is achieved by moving the shoulder joint to the position so that the hand can reach the target with a fully extended arm. We determine the desired joint position for the shoulder joint,  $G(J_{ball}, e)$  as follows:

$$G(J_{ball}, e) = \begin{cases} \mathbf{p}_{J_{ball}}, & \text{if } \|\mathbf{b}\| < l_e \\ \mathbf{p}_{J_{ball}} + (\|\mathbf{b}\| - l_e)\mathbf{b}, & \text{otherwise,} \end{cases} \quad (20)$$

where  $\mathbf{b} = \bar{\mathbf{p}}_e^O - \mathbf{p}_{J_{ball}}$ .

$\mathbf{p}_{J_{ball}}$  is the current shoulder joint position which can be represented

by forward kinematics (FK) as explained in Eq. (18).  $\bar{\mathbf{p}}_e^O$  is the desired hand joint position determined by  $\mathcal{F}_O$ , and  $l_e$  is the maximum arm length of the user in this upper body case. Then, we define the spine refinement term  $E_S$  by using the form of  $G(m)$  as follows:

$$E_S = \sum_{m \in \mathcal{A}} \omega_{e \in m} \|\mathbf{p}_{J_{ball}} - G(m)\|^2. \quad (21)$$

The importance values of hands  $\omega_{e \in m}$ , are valid only when the user starts to interact with an object with their hands within a distance threshold. We will note the value of the threshold in the implementation details.

**Pose preservation ( $E_p$ )** To preserve the original human pose, we penalize large displacements by defining the pose preservation term as follows:

$$E_p = \|\mathbf{x}_S\|^2. \quad (22)$$

We set  $\lambda_S$  and  $\lambda_P$  to 1.0 and  $10^{-4}$ , respectively.

## 7.3. Limb Posture Computation

With the root position and spine posture fixed, we determine the limb postures. There are multiple goals for limbs to achieve to preserve the semantic meanings of user motions while naturally interacting with environmental elements. We define three terms: object interaction ( $E_o$ ), foot positioning ( $E_f$ ), and pose preservation ( $E_p$ ). The objective function for this step is described as follows:

$$\arg \min_{\mathbf{x}_L} \lambda_o E_o + \lambda_f E_f + \lambda_p E_p. \quad (23)$$

$\mathbf{x}_L$  represents a set of rotational displacements of all limb segments, except for those of end-effectors because we do not consider any orientation updates of end-effectors. Each limb is optimized independently with the following terms.

**Object interaction ( $E_o$ )** This term enforces the avatar to reach the target by forcing the end-effector position  $\mathbf{p}_e$  to match with the desired position  $\bar{\mathbf{p}}_e^O$  determined in Section 5.2:

$$E_o = \omega_e \|\mathbf{p}_e - \bar{\mathbf{p}}_e^O\|^2. \quad (24)$$

Similar to the case above, we express  $E_o$  with the explicit form using FK and make it activated only when the user's limb comes close to an object.

**Foot positioning ( $E_f$ )** When the user performs locomotion, the foot position should be adjusted properly in the remote space. We define a foot positioning term as follows:

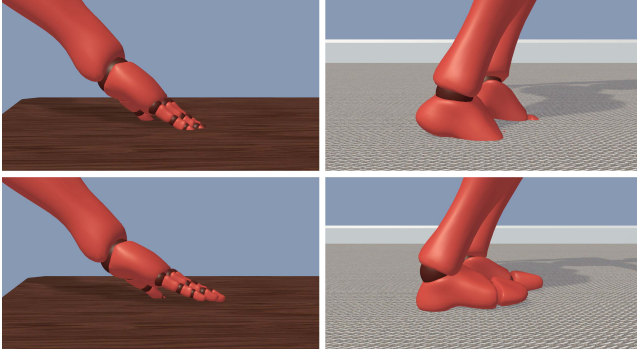
$$E_f = (1 - \omega_e) \|\mathbf{p}_e - \bar{\mathbf{p}}_e^P\|^2, \quad (25)$$

where  $\mathbf{p}_e$  denotes the current foot position which is computed by FK and  $\bar{\mathbf{p}}_e^P$  is the desired foot position determined by projecting the foot position of the user via  $\mathcal{F}_P$  as described in Section 5.1.

**Pose preservation ( $E_p$ )** We penalize large displacements to preserve the original human motion:

$$E_p = \|\mathbf{x}_L\|^2. \quad (26)$$

In this step, we set  $\lambda_o, \lambda_f$ , and  $\lambda_P$  to 10.0, 10.0, and  $10^{-5}$ , respectively.



**Figure 5:** End-effector refinement to remove penetrations into an environmental element (top) by adjusting its orientation (bottom).

The resulting avatar motion  $\mathcal{M}_{tar}$  is obtained by performing procedural updates of the user motion as follows:

$$\mathcal{M}_{tar} = ((\mathcal{M}_{init} \oplus \mathcal{S}_R^{-1}(\mathbf{x}_R)) \oplus \mathcal{S}_S^{-1}(\mathbf{x}_S)) \oplus \mathcal{S}_L^{-1}(\mathbf{x}_L). \quad (27)$$

For more details of the motion update and the definitions of the symbols used here, please see the supplementary document.

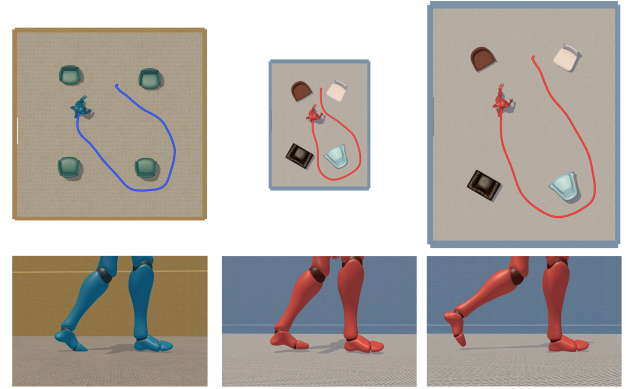
## 8. Pose Refinement

Because we do not consider the orientations of the end-effectors, penetration by the hands and feet into an object may occur in the resulting poses as shown in Figure 5. To alleviate this problem, we go through a refinement step that updates the orientation of each end-effector according to the measured penetration depth of each end-effector. The penetration depth can be measured easily by a height distance between the tip of the end-effector and the surface of floor or objects. Using this value, the end-effector is pushed up by rotation in the same way used for the computation of the desired root orientation described in Section 5.1. This simple refinement brings an additional benefit of reducing the dimension of optimization because the correct values are known right after optimization, we can exclude the rotational vectors of end-effectors in the definition of displacement  $\mathbf{x}$  described in Section 7. Given that our optimization is performed at every time step, we apply a linear motion blending technique between the resulting motion  $\mathcal{M}_{tar}^t$  and  $\mathcal{M}_{tar}^{t-1}$  to guarantee the time consistency [KPBL16,JKL18].

## 9. Results

### 9.1. Implementation Details

Our system is implemented in Unity 3D with C#. All of the experiments were performed using a desktop with an AMD Ryzen™ 9 5900X Processor (3.70GHz, 12 cores), 128GB memory, and Nvidia GeForce RTX 3080 Ti GPU. Our manual specification of the corresponding control points in the local and remote spaces in the offline stage required less than 30 seconds per each element pair through an intuitive GUI tool [Uni]. Our character model is downloaded from Mixamo [Mix] and composed of 24 joints with 72 degrees of freedom in total. A user motion is assumed to be input to our



(a) Local space

(b) Smaller and larger remote spaces

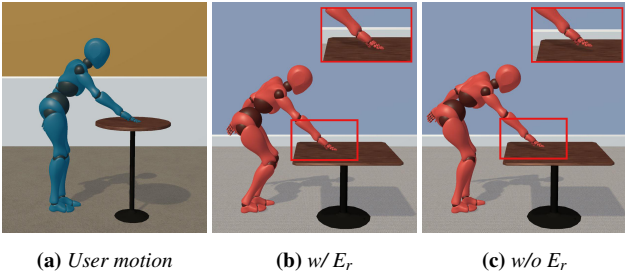
**Figure 6:** Top: The blue and red paths indicate the root path of the user motion and avatar motion, respectively. Bottom: Changes in the user stride according to the scale of the remote space.

system as a stream of motion capture data. The dimensions of optimization variables,  $\mathbf{x}_R$ ,  $\mathbf{x}_S$ , and  $\mathbf{x}_L$  are  $\mathbb{R}^{3 \times 1}$ ,  $\mathbb{R}^{3 \times 4}$ , and  $\mathbb{R}^{3 \times (4 \times 2)}$ , respectively. We optimized the objective functions using the conjugate gradient method in ALGLIB [Boc]. While the computation for the root position was performed numerically due to the discontinuity in Eq. (18), the remaining two steps, computation of spine and limb postures, are solved analytically in closed-forms of the first and second derivatives of our objective functions. In the process of computing the importance value [SLSG01] for avatar-objects interactions, we set the distance threshold to 0.6 for the root joint and 0.3 for the end-effectors. The blending parameter used for the pose refinement is set to 0.6 for the previous time step. Once manual specification of control points is completed in the offline stage, our online motion adaptation works in real-time requiring under 33ms/frame on average.

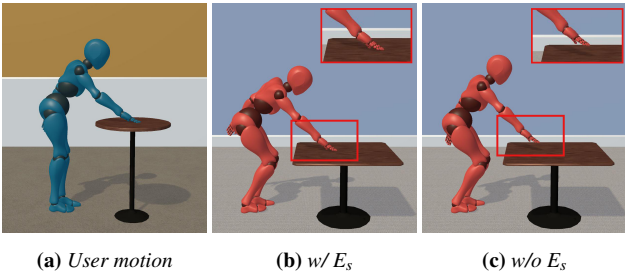
### 9.2. Examples

To examine the effectiveness of the proposed method, we created various paired indoor environments under different scenarios. We demonstrate properly adapted avatar locomotions including walking and running, object interaction motions, and motion transitions in the following. For animation results, please see the supplementary video.

**Locomotion** We performed experiments on our global placement determination of the avatar in remote spaces that have different sizes and shapes from those of the local space. We show that how the user root path illustrated in Figure 6a is adjusted to fit the smaller or larger spaces. In the smaller space depicted in Figure 6b (left), the avatar successfully passes through a narrow gap between two chairs without colliding with them. In the larger space depicted in Figure 6b (right), the avatar moves around one chair with a sufficient distance farther away. With the specified global placement of the avatar, its stride is properly adjusted according to the scale of the remote space as shown in the bottom of Figure 6b. To see how the foot trajectories are adjusted in action, please see the supplementary video.



**Figure 7:** Effectiveness of the reachability check term ( $E_r$ ) in the root position computation step.

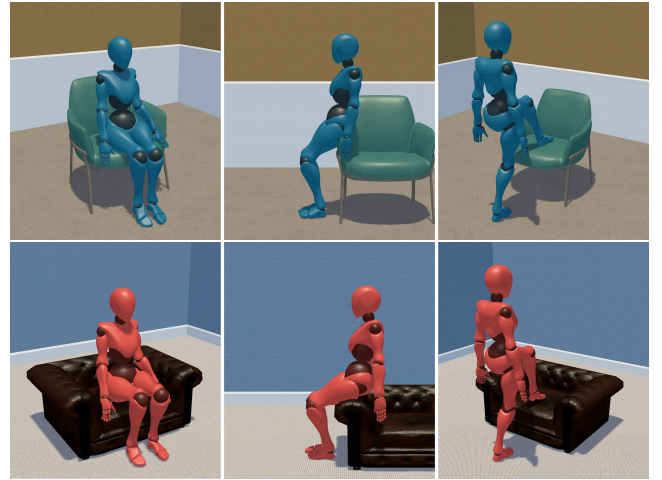


**Figure 8:** Effectiveness of the spine refinement term ( $E_s$ ) in the spine posture computation step.

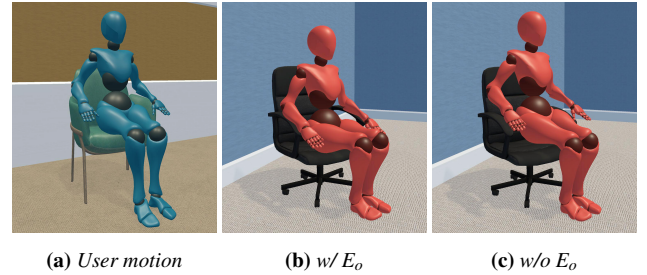
**Table interactions** We show the effectiveness of the reachability check term ( $E_r$ ) in adjusting the avatar root position in Figure 7. The avatar bends its leg to make the hand reach the target point on the table when applying the term as shown in 7b. Otherwise, the avatar tries to bend its upper body to achieve the same goal as depicted in Figure 7c. Figure 8 shows that the avatar successfully touches the target point on the table by properly bending its upper body with the spine refinement term ( $E_s$ ). Without using the term, the hand of the avatar fails to reach the target even though its arm is fully stretched as shown in Figure 8c.

**Chair interactions** Using chairs with different sizes and shapes, we show how the proposed method adapts various chair interaction motions such as sitting on a chair with hands touching the seat, sitting on the armrest, and putting a foot on the seat in Figure 9. In the last scenario in Figure 9, we also show how smoothly our system makes a transition from  $\bar{\mathbf{p}}_e^P$  to  $\bar{\mathbf{p}}_e^O$ , a foot-object interaction in this case, using the foot positioning term. Figures 10b and 10c highlight different resulting motions of the avatar generated with and without the object interaction term ( $E_o$ ) when the user sits on a chair.

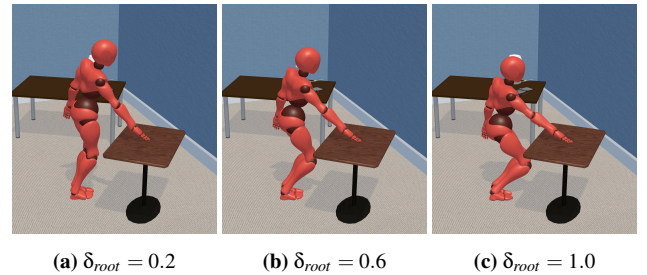
**Motion transition** Taking advantage of the joint importance analysis [SLSG01], our system makes the avatar smoothly transition between locomotion and object interaction motions in both ways. The importance value is activated according to the maximum distance threshold  $\delta_{key}$ , which denotes the maximum range in which a joint is considered to be involved with an interaction motion. We tested how the maximum distance threshold affects the resulting motions. Figure 11 shows how the root position changes according to the different value of  $\delta_{root}$ , the threshold associated



**Figure 9:** Top: User motions. Bottom: Adapted avatar motions. Chair interaction motions such as sitting on a chair (left), sitting on the armrest (middle), and putting a foot on the seat (right) are adapted according the chair with different shapes and heights.



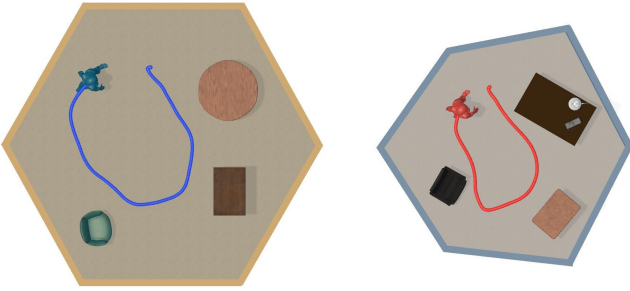
**Figure 10:** Effectiveness of the object interaction term ( $E_o$ ) in the root position and limb posture computation steps.



**Figure 11:** Resulting avatar motions generated using different values of  $\delta_{root}$ . With a fixed value for end-effectors ( $\delta_e=0.3$ ), the root position determined by the optimization differs according to the value of  $\delta_{root}$ .

with the root joint. The result shows that a larger threshold leads to a more sensitive avatar motion. In contrast, a smaller threshold value makes the key joint of the avatar ignore the object until they become relatively closer. For animation results, please see the supplementary video.





**Figure 12:** The blue path indicates the user’s root path (left) and the red path indicates the adjusted root path of the avatar (right).

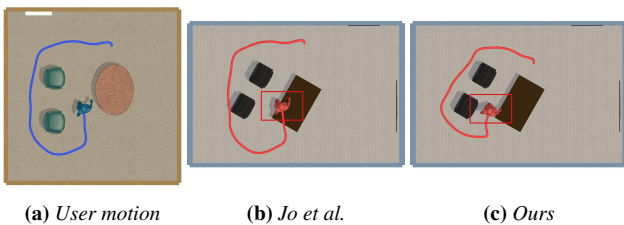
**Non-rectangular spaces** We assume that the spaces are of rectangular shape in general. Our TPS-driven approach, however, can also be applied to a more general case of the room shape as shown in Figure 12. This verifies that a reasonable mapping can be produced as long as the given spaces are morphologically-similar.

### 9.3. Comparison

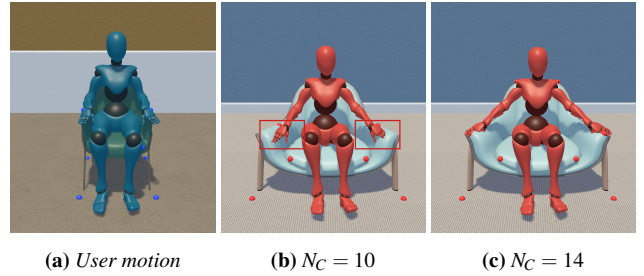
To explore the effectiveness of our global placement determination strategy, we compared our method with Jo et al. [JKK15] who also established a bijective planar mapping between two spaces. Because they use an affine transformation to warp the space to another, the resulting root path collides with the object as shown in Figure 13b. This implies that the use of such a simple transformation may not be suitable for the object rearrangement in space with a different size. In contrast, our global placement determination ensures that the avatar can move around in the remote space without any collision with objects as shown in Figure 13c.

### 9.4. Experiments on Heuristic Design

**Control point specification** When specifying the control points for the floors as described in Section 4, we intuitively chose four points at each corner of the object. We empirically discovered that specifying three points is enough to prevent the avatar from colliding with an object even if these three points may not cover all planar areas that the object occupies on the floor. It is because the user motion is originally given without colliding with a bounding box of the object regardless of the number of control points.



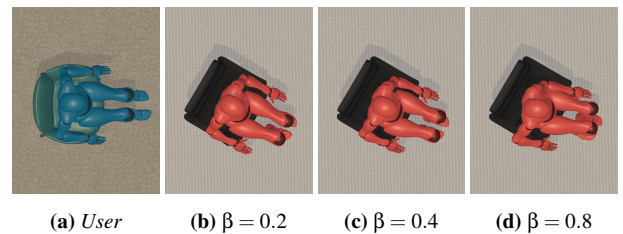
**Figure 13:** Comparison of global placement determination. The user motion (left), the global placement determined by Jo et al. [JKK15] (middle), and that by our method (right).



**Figure 14:** Resulting avatar motion according to the number of control points ( $N_C$ ) specified on the chair. Red boxes show that the desired contact is not made between the hands and the chair.

For an object-level mapping, the control points are strategically specified near high curvature areas so that the overall shape of an object can be revealed. This strategy is similar to that of Kim et al. [JKK15], but the underlying purposes of specifying the points are different. We select the control points to define a bijective mapping between the convex hulls created by the points, which enclose the objects. Kim et al. [JKK15] choose the points considering which points will mainly affect the avatar motions. Using an insufficient number of points can degrade the quality of the resulting avatar motion interacting with an object as shown in Figure 14. We empirically discovered that at least 12 and 6 points are necessary for a chair and a table, respectively, to ensure the preservation of the semantics of user motion. For more detailed animation results, please see the supplementary video.

**Desired root orientation determination** In Section 5.1, the reference point  ${}_{xz}\mathbf{r}$  used for determining the avatar’s forward direction  ${}_{xz}\mathbf{d}^P$  was determined by the scaling factor  $\beta$ . We performed an experiment to find the proper value of  $\beta$ . The forward direction of the avatar deviates from the expected direction as the value of  $\beta$  increases as shown in Figure 15. This is because that the distant reference point from the user tends to be more affected by the TPS-driven transformation than the close one does, resulting in a wrong direction in the remote space. We empirically found that setting the value of  $\beta$  to 0.2 works well for all scenarios. While increasing the value of  $\beta$  negatively affects the resulting motion, any positive value of  $\beta$  less than 0.2 did not make any noticeable difference.



**Figure 15:** Changes in the forward direction of the avatar according to the value of  $\beta$ .





(a) 50% smaller remote space compared to the local space (b) 80% larger remote space compared to the local space

**Figure 16:** Walking motions in much different remote spaces. The avatar’s feet are overlapped in an extremely smaller space (left) and the avatar tends to fly in an extremely larger space (right).

### 9.5. Qualitative Evaluation

We conducted two user studies to find out the operable range of our method. We designed experiments to identify the acceptable range of orientation differences between two objects and size differences between the spaces, which leads to visually plausible animation results. The participants were asked to evaluate the naturalness of the resulting motion on a 5-Likert Scale after watching each video. From the results, we concluded that the resulting motions are acceptable within the range of  $-90^\circ$  to  $+60^\circ$  in terms of the orientation difference and  $-40\%$  to  $+70\%$  in terms of the size difference. We refer the readers to the supplementary document for further details of the analysis of the results and how we conducted the evaluation.

### 10. Limitations and Future Work

Although our method can successfully generate a full-body avatar motion that reflects the semantic meaning of the user motion despite the environmental differences between the two spaces used for tele-communication, it is limited by the condition under which the two spaces should be morphologically-similar and the difference is not too large. If the remote space is much larger or smaller than the local space, the resulting walking motion of the avatar will become unnatural with too big or small strides as shown in Figure 16. It is because our method neither generates new motions nor plans for unknown future user inputs. We show some failure cases that occur due to large differences in the size of the spaces in the supplementary video. We also cannot handle dexterous object interaction motions because we rely on simplified joint configurations for the user and avatar.

If two objects are located too closely together in both spaces, the control points densely-distributed at the bottom of the objects induce improper floor mapping ( $\mathcal{F}_p$ ) in the remote space. This sometimes leads to the wrong orientation or misplaced foot positions of the avatar, failing to reflect the proper semantics of the user motion of object interaction. The orientation mismatch can be alleviated by tuning the heuristic parameter,  $\beta$ , as discussed in Section 9.4. However, the misplaced foot position is hard to be resolved in such a heuristic way. For animation results, please see the supplementary video.

In some cases, jittering is observed in the resulting avatar motion, caused by the frame-wise optimization. By employing a temporal coherence term, this occasional jittering can be suppressed.

The maximum distance threshold for each joint determines when the avatar should react to nearby objects. We used the same set of threshold values for all of our experiments for general applicability. This sometimes led to sudden changes of the joint values in the resulting motion. Individually tuning the threshold values for each object can resolve this artifact. Another approach would be to incorporate a data-driven model that learns human motion features considering future and past motions as well as the current pose. This learning based approach may provide an easier way to produce object interaction motion of an avatar without artifact.

Another assumption we made in this work is that the environmental objects are fixed. We believe that the method can be further extended to handle movable object interactions by dynamically updating our TPS-driven transformation weights. Employing a machine learning-based motion generation scheme may allow to handle more diverse scenarios than we used in this work. For example, our method can be extended to augment human-object interaction motion data and associated weights under diverse scenarios with numerous interior layouts of the involved spaces. Using a network with the augmented data, the proper avatar motion can be generated according to the user behavior in real-time by querying the network.

In the first step of our method, we manually select control points for each paired objects and floors. While it required less than 30 seconds for each pair of objects using a commercial tool in our examples with several interior objects, the manual process will become tedious with more interesting and complex scenarios that utilize a lot of objects. Automatic detection of key points of environmental objects using a volumetric sensor [SZKS19] will be a useful extension, and dynamically calculating the environment mapping weights using this information will be an interesting future work. We assumed that the user motions are provided from a local space as a streaming input. Typically, the user motions are captured using expensive and complex motion capture systems. A promising alternative is to utilize ego-centric motion capture approaches [JG17, YK19, GMSPM21]. The use of such approaches will bring the deployment of a fully pipelined 3D telepresence system one step closer to reality.

### 11. Conclusion

We presented a novel motion adaptation method that allows to generate continuous full-body motions of an avatar by extending the TPS-based transformation to a 3D domain. Using bijective environment mapping functions between two spaces, we demonstrated that human-like avatar motions including locomotion and object interaction motions, as well as smooth transitions between them can be created on-the-fly in real-time. We verified the effectiveness of each step of our adaptation method through various experiments.

### Acknowledgements

We thank the anonymous reviewers for their invaluable comments; Seonghyeon Kim, Jung Eun Yoo, and Dawon Lee for the helpful discussions; Hyerin Koo and Woonjung Kim for editing animation data; Sebastian Starke for sharing AI4Animation framework as open-source which we referred for visualizing results.

This work was supported by Global Research Cluster program of Samsung Advanced Institute of Technology (IO210607-08717-01).

## References

- [AAKC13] AL-ASQHAR R. A., KOMURA T., CHOI M. G.: Relationship descriptors for interactive motion adaptation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2013), pp. 45–53. 2, 3
- [ATCO\*10] AU O. K.-C., TAI C.-L., COHEN-OR D., ZHENG Y., FU H.: Electors voting for fast automatic shape correspondence. In *Comput. Graph. Forum* (2010), vol. 29, Citeseer, pp. 645–654. 2
- [Boc] BOCHKANOV S.: ALGLIB optimization. [www.alglib.net](http://www.alglib.net). 8
- [Boo89] BOOKSTEIN F. L.: Principal warps: Thin-plate splines and the decomposition of deformations. *IEEE Transactions on pattern analysis and machine intelligence* 11, 6 (1989), 567–585. 2, 4
- [CK00] CHOI K.-J., KO H.-S.: Online motion retargeting. *The Journal of Visualization and Computer Animation* 11, 5 (2000), 223–235. 3
- [CWS18] CONGDON B. J., WANG T., STEED A.: Merging environments for shared spaces in mixed reality. In *Proceedings of the 24th ACM Symposium on Virtual Reality Software and Technology* (2018), pp. 1–8. 1, 2
- [GMSPM21] GUZOV V., MIR A., SATTLER T., PONS-MOLL G.: Human positioning system (hps): 3d human pose estimation and self-localization in large scenes from body-mounted sensors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 4318–4329. 11
- [HCV\*21] HASSAN M., CEYLAN D., VILLEGAS R., SAITO J., YANG J., ZHOU Y., BLACK M. J.: Stochastic scene-aware motion prediction. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2021), pp. 11374–11384. 3
- [HKT10] HO E. S., KOMURA T., TAI C.-L.: Spatial relationship preserving character motion adaptation. In *ACM SIGGRAPH 2010 papers*. 2010, pp. 1–8. 3
- [JG17] JIANG H., GRAUMAN K.: Seeing invisible poses: Estimating 3d body pose from egocentric video. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2017), IEEE, pp. 3501–3509. 11
- [JKK15] JO D., KIM K.-H., KIM G. J.: Spacetime: adaptive control of the teleported avatar for improved ar tele-conference experience. *Computer Animation and Virtual Worlds* 26, 3-4 (2015), 259–269. 1, 2, 10
- [JKL18] JIN T., KIM M., LEE S.-H.: Aura mesh: Motion retargeting to preserve the spatial relationships between skinned characters. In *Computer Graphics Forum* (2018), vol. 37, Wiley Online Library, pp. 311–320. 3, 8
- [KB19] KELLER W., BORKOWSKI A.: Thin plate spline interpolation. *Journal of Geodesy* 93, 9 (2019), 1251–1269. 4
- [KL20] KIM M., LEE S.-H.: Deictic gesture retargeting for telepresence avatars in dissimilar object and user arrangements. In *The 25th International Conference on 3D Web Technology* (2020), pp. 1–6. 1, 2
- [KPBL16] KIM Y., PARK H., BANG S., LEE S.-H.: Retargeting human-object interaction to virtual avatars. *IEEE transactions on visualization and computer graphics* 22, 11 (2016), 2405–2412. 1, 2, 3, 8
- [KYKC20] KESHAVARZI M., YANG A. Y., KO W., CALDAS L.: 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)* (2020), IEEE, pp. 353–362. 2
- [KZY\*22] KESHAVARZI M., ZOLLHOEFER M., YANG A. Y., PELUSE P., CALDAS L.: Mutual scene synthesis for mixed reality telepresence. *arXiv preprint arXiv:2204.00161* (2022). 2
- [LLL17] LEE Y., LEE S., LEE S.-H.: Multifinger interaction between remote users in avatar-mediated telepresence. *Computer Animation and Virtual Worlds* 28, 3-4 (2017), e1778. 1
- [LMR14] LEHMENT N. H., MERGET D., RIGOLL G.: Creating automatically aligned consensus realities for ar videoconferencing. In *2014 IEEE International Symposium on Mixed and Augmented Reality (ISMAR)* (2014), IEEE, pp. 201–206. 2
- [LMT08] LYARD E., MAGNENAT-THALMANN N.: Motion adaptation based on character shape. *Computer Animation and Virtual Worlds* 19, 3-4 (2008), 189–198. 3
- [LS99] LEE J., SHIN S. Y.: A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the 26th annual conference on Computer graphics and interactive techniques* (1999), pp. 39–48. 6
- [Mix] Mixamo. <https://www.mixamo.com>. 8
- [MYD\*13] MAIMONE A., YANG X., DIERK N., STATE A., DOU M., FUCHS H.: General-purpose telepresence with head-worn optical see-through displays and projector-based lighting. In *2013 IEEE Virtual Reality (VR)* (2013), IEEE, pp. 23–26. 2
- [OERF\*16] ORTS-ESCOLANO S., RHEMANN C., FANELLO S., CHANG W., KOWDLE A., DEGTAREV Y., KIM D., DAVIDSON P. L., KHAMIS S., DOU M., ET AL.: Holoportation: Virtual 3d teleportation in real-time. In *Proceedings of the 29th annual symposium on user interface software and technology* (2016), pp. 741–754. 2
- [OLK\*16] OH J., LEE Y., KIM Y., JIN T., LEE S., LEE S.-H.: Hand contact between remote users through virtual avatars. In *Proceedings of the 29th International Conference on Computer Animation and Social Agents* (2016), pp. 97–100. 1
- [PKB\*16] PEJSA T., KANTOR J., BENKO H., OFEK E., WILSON A.: 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* (2016), pp. 1716–1725. 2
- [RAS17] ROCCO I., ARANDJELOVIC R., SIVIC J.: Convolutional neural network architecture for geometric matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition* (2017), pp. 6148–6157. 4
- [SLSG01] SHIN H. J., LEE J., SHIN S. Y., GLEICHER M.: Computer puppetry: An importance-based approach. *ACM Transactions on Graphics (TOG)* 20, 2 (2001), 67–94. 2, 3, 5, 6, 7, 8, 9
- [SZKS19] STARKE S., ZHANG H., KOMURA T., SAITO J.: Neural state machine for character-scene interactions. *ACM Trans. Graph.* 38, 6 (2019), 209–1. 3, 11
- [TSDS19] TOMAR Y., SRIVASTAVA A., DEY A., SHARMA O.: Conformal redirected walking for shared indoor spaces. In *The 17th International Conference on Virtual-Reality Continuum and its Applications in Industry* (2019), pp. 1–8. 1, 2
- [Uni] Unity. <https://unity.com>. 8
- [WYS\*22] WANG X., YE H., SANDOR C., ZHANG W., FU H.: Predict-and-drive: Avatar motion adaption in room-scale augmented reality telepresence with heterogeneous spaces. *IEEE Transactions on Visualization and Computer Graphics* (2022). 2
- [YK19] YUAN Y., KITANI K.: Ego-pose estimation and forecasting as real-time pd control. In *Proceedings of the IEEE/CVF International Conference on Computer Vision* (2019), pp. 10082–10092. 11
- [YYCL21] YOON L., YANG D., CHUNG C., LEE S.-H.: A full body avatar-based telepresence system for dissimilar spaces. *arXiv preprint arXiv:2103.04380* (2021). 1, 2
- [YYK\*20] YOON L., YANG D., KIM J., CHUNG C., LEE S.-H.: Placement retargeting of virtual avatars to dissimilar indoor environments. *IEEE Transactions on Visualization and Computer Graphics* (2020). 1, 2