Retargeting Human-Object Interaction to Virtual Avatars

Yeonjoon Kim, *Student Member, IEEE*, Hangil Park, Seungbae Bang, and Sung-Hee Lee, *Member, IEEE*

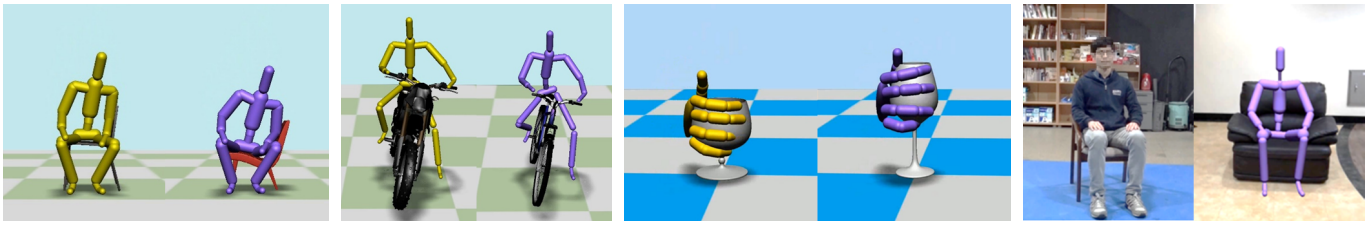

Fig. 1: Our method based on a spatial map retargets the interaction motions with respect to an object (left) to a similar object with a different shape in real-time (right).

Abstract—In augmented reality (AR) applications, a virtual avatar serves as a useful medium to represent a human in a different place. This paper deals with the problem of retargeting a human motion to an avatar. In particular, we present a novel method that retargets a human motion with respect to an object to that of an avatar with respect to a different object of a similar shape. To achieve this, we developed a spatial map that defines the correspondences between any points in the 3D spaces around the respective objects. The key advantage of the spatial map is that it identifies the desired locations of the avatar's body parts for any input motion of a human. Once the spatial map is created offline, the motion retargeting can be performed in real-time. The retargeted motion preserves important features of the original motion such as the human pose and the spatial relation with the object. We report the results of a number of experiments that demonstrate the effectiveness of the proposed method.

Index Terms—Motion retargeting, human-object interaction, augmented reality, telepresence, avatar animation.

1 Introduction

Rapid advances of AR technologies are realizing a vision in which people in different locations can interact seamlessly with each other in a shared space. Recently, there has been considerable research on virtual avatar-based 3D telepresence in which a user wearing a head mounted display communicates with the image of a remote person overlaid on the local space of the user [20, 21]. In this case, the physical space that a user belongs to becomes the telepresence space: The local space is augmented with the remote person's avatar, which moves in the local space by following the remote person's motions made in the remote place. In addition, as envisioned in [13], we can even make a mirrored telepresence space in the remote space by sending the local user's avatar to the remote place.

If the shapes of the local and remote environments are identical, we can generate the avatar's motion simply by copying that of the remote user. Otherwise, the avatar's motion needs to be created by adapting the remote user's motion to the local environment. In particular, a challenging problem arises when the remote user engages in a physical interaction (e.g., contact) with an object. Let us assume that there exists a local object corresponding to the remote object. If the two objects have different shapes, how can we transfer the remote user's motion to the avatar? For example, let us assume that the two spaces are equipped with chairs with different shapes. When the remote user sits on a chair, the avatar in the local space should be able to perform a sitting motion appropriate for the shape of the local chair.

In this paper, we confront the problem of retargeting a motion interacting with an object to a different object that has a similar but different shape. Such motion retargeting should preserve two features: one is the context of the motion with respect to the object; the other is the original pose of the human. The difficulty of motion retargeting increases with the dissimilarity of the function and shape of the two objects. For instance, retargeting a sitting motion on an office chair to a sofa is easier than it would be to the saddle of a bicycle.

We introduce a novel motion retargeting technique for two objects that have similar functions and shapes. More specifically, we assume that there is a correspondence between the surfaces of the objects such that for every point on one surface there exists a corresponding point on the other, and that an arbitrary continuous curve on one surface can be mapped to a continuous curve on the other. In other words, it is assumed that a continuous and bijective map $f : S_1 \rightarrow S_2$ exists between the two surfaces S_i. This assumption allows for a continuous movement of contact points between a user and an object to be transferred to a continuous contact motion on the other object. Then, how can a motion close to yet not in contact with an object be transferred to the nearby space around the other object? This problem has been more or less overlooked except for a few studies, e.g., [2, 10, 24].

To solve this problem, we develop a novel motion retargeting technique by creating a spatial map that defines the correspondences between 3D spaces surrounding the respective objects. The map is constructed to specify the desired position of an avatar's body part with respect to a local object, equivalent to the remote user's relative position with respect to the remote object. Thus, even if there is no contact between the user and the object, the spatial relation between each body part of the user and the object can be preserved when retargeted to the avatar interacting with the target object. Figure 1 shows examples that human-object interaction motions are retargeted to different objects such as chairs, bikes, and cups.

In an AR telepresence scenario in which each user uses her local space as a telepresence space and interacts with the avatar of the remote participant, real-time performance is an important requirement, because otherwise the local user may misunderstand the remote user's intention due to the delay. Our motion retargeting technique is well suited to such AR telepresence applications. The spatial map is created only from the geometries of the two corresponding objects, without input motions of the human. Therefore, it can be constructed offline in a preprocessing step, which allows for an efficient computation of avatar motion that can be achieved by solving the inverse kinematics

• *All authors are with Korea Advanced Institute of Science and Technology (KAIST). E-mail: {yeonjoon, phgphg, be2848, sunghee.lee}@kaist.ac.kr.*

online. Besides telepresence applications, our method can be used to control an avatar by demonstration, especially when the avatar needs to interact with objects.

The rest of the paper proceeds as follows. After reviewing related work in Sec. 2, we detail our motion retargeting technique in Sec. 3. We verify the effectiveness of our method through various experiments in virtual environments and in a prototype telepresence environment in Sec. 4. Section 5 discusses future work and concludes the paper.

2 Related Work

Our study is related to research in the fields of telepresence, character animation, and geometric mesh processing.

2.1 Telepresence Using a Virtual Avatar

To achieve a more immersive telepresence experience, researchers have made efforts to display the image of a remote person seamlessly in the local physical space or in a shared virtual space. There have been studies to create remote user's images as seen from arbitrary viewpoints [20], to enable eye contact or hand contact between the remote users [14, 22], and to control lighting and occlusion to enhance the illusion of a remote user's presence [21]. Group-to-group telepresence has also been studied [4].

The approach of displaying a remote user's appearance as it is has the advantage of delivering the expressions and motions of the remote user realistically. However, the underlying assumption of this approach is that the remote and local environments are more or less the same, and this assumption can create unnatural results as the shape of the remote environment becomes different from that of the local environment. Our motion retargeting technique modifies the remote user's motion to fit the local environment, and thus a user can be provided with a realistic illusion, as if the remote user were coexisting in the same space, even when there are large differences between the spaces.

Recently, researchers have begun to investigate the motion retargeting problem for telepresence. Jo et al. [13] applied an existing motion retargeting method between different environments [2, 10] to a telepresence system. While the motivation and goal of our work are similar to theirs, we develop a new motion retargeting technique that overcomes some of the limitations of the existing techniques, as discussed in Sec. 2.2.

2.2 Motion Retargeting

The motion retargeting problem has been widely studied in computer graphics. Most research concerns transferring the motion of a character to another character of different size and structure. Using control techniques [7], filtering [29], and optimization [9], new motions are generated to preserve certain features such as end-effector trajectories and to satisfy kinematic and dynamic constraints. Laplacian editing has also been employed to create multi-character interaction motions [15].

We are interested in retargeting motions interacting with an object to another object with a different shape. The interaction mesh technique proposed by Ho et al. [10] can deal with this problem by constructing a mesh from the sample points on the character and the object, and creating a new motion by deforming the mesh according to the changed shape of the object or the character. Laplacian deformation energy of the mesh is minimized to maintain the spatial relation between the character and the object. The interaction mesh technique produces outstanding results but it is mainly for the offline motion editing process because it requires both the character motion and the object shape as input and performs optimization over the motion sequence. Al-Asqhar et al. [2] significantly improved the speed of retargeting, but their method also requires both the character motion and the object shape to generate a *relationship descriptor* as a preprocessing stage, which makes it difficult to use for telepresence applications. In contrast, our preprocessing stage uses only the geometry information of the objects, and thus the online stage allows for arbitrary input motions to be processed in real-time. Sandilands et al. [24] used the electric coordinates [30] for motion transfer, but the capability of their method with respect to preserving the geometric and functional features between objects is somewhat limited. In summary, the unique advantage of our method is that it retargets human-object interaction motion to other objects in a single pass with a capability of preserving spatial correspondences between spaces around objects.

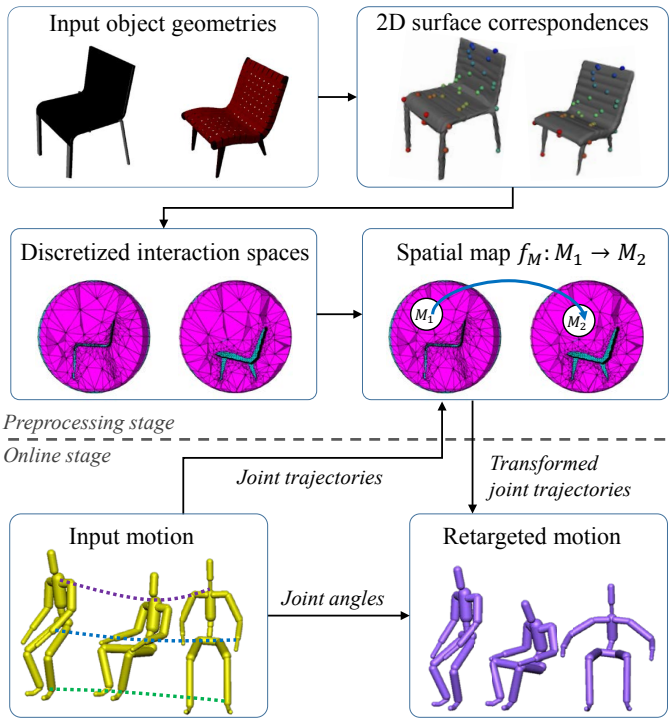

Fig. 2: Overview of the proposed retargeting method.

2.3 Mesh deformation and parameterization

Our approach to constructing the spatial map is based on the techniques of mesh deformation and parameterization that have been studied extensively in computer graphics. Interested readers are referred to [5] for a survey of the mesh deformation and [8, 26] for a survey of the mesh parameterization.

To obtain a reasonable deformation of the 3D space, we use Laplacian deforming (or Dirichlet) energy for mesh deformation. Specifically, the cotangent formula, a simplification of the Dirichlet energy, is used to make the deformation process efficient. The formula was developed in [23] for a 2-dimensional triangular mesh and in [31] for a 3-dimensional tetrahedral mesh, which is used in our work.

Besides the Dirichlet energy, researchers have developed many methods for geometric mesh deformation. The As-Rigid-As-Possible (ARAP) energy for triangular meshes was introduced in [3], and many variations have been developed for different objectives of deformation [11, 28]. Liu et al. [18] explained a general framework for the ARAP energy and performed a triangular mesh parameterization. Chao et al. [6] extended this to a tetrahedral mesh to perform deformation, parameterization, and physical simulation of a deformable body.

3 Motion Retargeting Method

Our motion retargeting technique finds a bijective correspondence map between the surrounding spaces of the two objects and retargets an interaction motion with an object from one space to the other. This approach brings the benefit of preserving the spatial relation between the human body and the object whether they are in contact or not. Figure 2 provides an overview of the motion retargeting process, which consists of a preprocessing stage that constructs a spatial map and an online stage that generates the avatar motion given the stream of human motion data. In the preprocessing stage, we first find the surface correspondences between two given objects (Sec. 3.1.2) and then

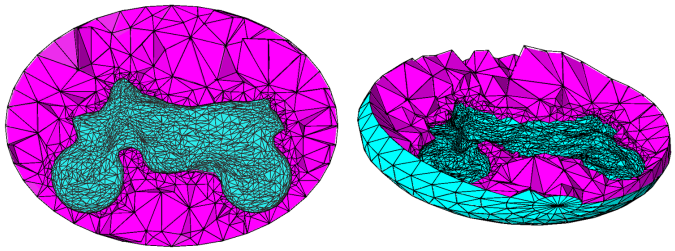

Fig. 3: Two cross sectional views for a volumetric mesh. An interaction space between a motorcycle and an ellipsoid is shown in magenta. The internal and external boundary surfaces are colored cyan.

create volumetric meshes surrounding the objects, from which we define the spatial map, the one-to-one mapping between the neighboring spaces around the objects (Sec. 3.1.3). In the online stage (Sec. 3.2), to preserve the spatial relationship with the target object, the spatial map transforms the joint trajectories of the input motion. By referring to the transformed joint trajectories, the retargeting algorithm determines the avatar's motion such that it optimally preserves both the human pose and the spatial relationship with the objects.

3.1 Spatial Map

Let us define a 3-dimensional space enclosing an object as an interaction space. Each interaction space consists of internal and external boundaries as well as an interior volume. The internal boundary I is the surface of an object. The external boundary E is a closed surface homeomorphic to the sphere and is set to be far enough from the object so that motion outside E can be safely considered to be unrelated to the object. Thus, human motion performed outside the interaction space can be directly applied to the avatar.

Our goal is that for two given objects O_1 and O_2, we define the interaction spaces M_1 and M_2 and construct a spatial map $f_M : M_1 \to M_2$. The construction of the spatial map f_M needs a map $f_I : I_1 \to I_2$ between the object surfaces (or internal boundaries), a map $f_E : E_1 \to E_2$ between external boundaries, and a map $f_V : V_1 \to V_2$ between interior volumes. The external boundaries of the interaction spaces should have the same shape in order to smoothly retarget a motion passing through the boundary. For this, in our implementation we simply model the external boundaries as spheres (or half spheres) of equal radius; our f_E is simply an identity map between the two spheres. We have assumed that there exists a continuous bijective map f_I, and Sec. 3.1.2 explains how to obtain the map f_I. A reasonable choice for the map f_V would be a map that smoothly interpolates f_I and f_E and minimizes the distortion of correspondence between the two spaces. To this end, in Sec. 3.1.3, we define an energy that measures the extent of deformation from a source space to a target space, and obtain f_M as a solution to the optimization problem that minimizes this energy under the hard constraints described by the boundary conditions f_I and f_E.

3.1.1 Discretization of Interaction Spaces

The spatial map f_M should be able to determine the position in M_2 corresponding to any point $p \in M_1$. This is achieved by finding correspondences for some sample points. The correspondences of the remaining points are obtained by interpolating the correspondences of the sample points. To this end, we decompose M_1 into a tetrahedral mesh and then use the vertices as the sample points of M_1. For convenience, we will let M_1 denote both the real continuous space as well as the discretized tetrahedral mesh. The coordinates of a point inside a tetrahedron is expressed by the barycentric coordinates with respect to the four vertices of the tetrahedron. We first model the internal and external boundaries of M_1 as triangular meshes, and then, using the Tetgen software package, obtain the tetrahedral meshes (Fig. 3) with the boundaries [27].

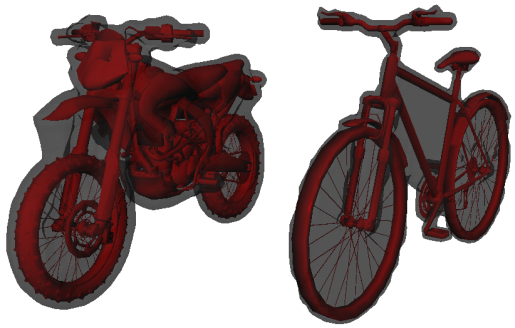

Fig. 4: Newly generated genus-zero water-tight mesh (grey) and its original unclean mesh (red).

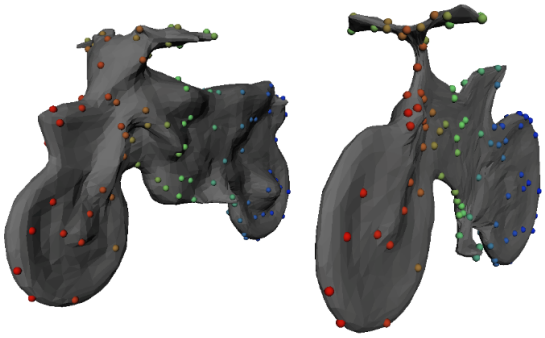

Fig. 5: Correspondence results between a motorcycle and a bicycle.

3.1.2 Correspondences between Objects

The procedure for creating a triangle mesh for the internal boundary is detailed here. To generate a tetrahedral volume mesh, the boundaries must be given as a water-tight mesh. In addition, to generate valid correspondences for our purpose, the two surface meshes for the internal boundaries I_1 and I_2 must have the same topology. However, most 3D models, either those found on the Internet or those obtained from 3D scanners, do not satisfy the conditions. Thus, we designed a framework for finding surface correspondences of two objects by creating watertight triangle meshes enclosing the original objects.

We make new objects O_1' and O_2', which have shapes similar to those of the original objects O_1 and O_2 but with a genus-zero water-tight single mesh. To this end, we first construct a voxel grid that covers sufficient space of the object mesh. Subsequently, for each grid point, we compute the signed distance to the closest surface of the object using a generalized winding number [12], which can robustly generate inside-outside segmentation of unclean meshes. This will convert the mesh object to an implicit surface with zero value on the object surface. We increase the offset of this implicit surface by adding a small value until we find a genus-zero surface, which is then converted to a triangle mesh using the marching cube method [19]. As a result, we obtain a genus-zero water-tight mesh with a small offset distance from the given object (Fig. 4).

Next, we use the non-rigid ICP algorithm [16] to shrink the newly generated water-tight mesh to the original mesh. In this iterative process, we take a small step to the target deformed position with a large number of iterations (≈ 150). The iteration is stopped if convergence is reached. Sometimes a flipped edge can occur, in which case we stop the iteration and use the last deformed mesh without any flipped edge. An alternative to our approach would be to employ constraints penalizing inverted elements [25] in the non-rigid ICP. Our initial experiment with this led to too many iterations to converge, resulting in a huge computation time. Instead of improving this part, we used a

moderate result that had an insignificant influence on the final result. The resulting meshes contain distance offsets of 0.1 - 5 cm for the regions of interest such as handlebars and saddle.

With the newly generated object meshes O'_1 and O'_2, we use the non-rigid ICP algorithm [16] to find correspondences between the two objects. The non-rigid ICP algorithm deforms O'_1 to create O''_1, which matches O'_2. Since we need two meshes with the same topology, we replace O'_2 with O''_1 (Fig. 5). In some cases the two objects differ significantly and it is necessary to manually align semantically matching regions (e.g. bike handlebars, armrest of chair). In such cases, before running the ICP algorithm, we performed manual alignment of the object using modeling tools such as Maya. Figure 5 shows a result of finding correspondences between a motorcycle and a bicycle, wherein correspondences of some sample points are visualized with colored spheres.

3.1.3 Finding interior correspondence

Our approach to finding the interior correspondences is to generate a new tetrahedral mesh M_2 that has the same connectivity as that of M_1 by deforming M_1 such that the boundaries coincide with the target boundaries I_2 and E_2. Once the tetrahedral mesh M_2 is acquired this way, the correspondences between the vertices of M_1 and M_2 are clear: the i-th vertex of M_1 is mapped to the i-th vertex of M_2. Then, the map $\mathbf{y} = f_M(\mathbf{x})$ for any $\mathbf{x} \in M_1$ can be defined as a piecewise affine map as follows:

$$\mathbf{y} = \sum_{i=1}^{4} b_i(\mathbf{x}) \mathbf{v}_{id(\mathbf{x},i)} \tag{1}$$

where $b_i(\mathbf{x})$ are the i-th barycentric coordinates of $\mathbf{x} \in M_1$ in the associated tetrahedron of which the vertices are identified by $id(\mathbf{x},i)$. $\mathbf{v} \in R^3$ denotes the position of the corresponding vertex in M_2. In other words, $\mathbf{x} \in M_1$ is mapped to $\mathbf{y} \in M_2$ using the same barycentric coordinates, but with respect to the vertices of the transformed tetrahedron. Now we explain how to determine the vertex positions of M_2. Basically, we set the vertex positions so as to minimize the deformation energy. Since we construct f_M as a piecewise affine map, the restricted map f_α of the map f_M to a tetrahedron T_α in M_1 is simply an affine map. For an affine map, one of the simplest ways of measuring the magnitude of distortions is to use the Dirichlet energy, defined by $\int_{T_\alpha} \| \bigtriangledown f_\alpha \|^2 dvol$. Simplifying this energy for the total volume M_1, we can obtain a quadratic function, called the cotangent formula, of the vertex positions. For a detailed analysis of this cotangent formula, see [23] and [31]. Computation of the vertex positions that minimize this quadratic energy is performed very efficiently using linear solvers such as the conjugate gradient method.

A solution that minimizes the Dirichlet energy can be understood physically as a solution of a static spring system [27], in which the distortion of each element tends to spread out evenly on the whole, but not to be concentrated on a particular local area. Therefore, the solution achieves smooth deformation.

We have additionally tested a more advanced method [6] for mesh deformation using the As-Rigid-As-Possible energy, which attempts to deform the mesh by translating and rotating the tetrahedrons as much as possible. We found that the resulting deformations are not very different in our experiments despite the longer processing time, probably because the deformation of the space is not extreme. Hence we chose to use the Dirichlet energy.

3.2 Online Motion Generation

In this section, we present our optimization-based method to retarget a pose. Let the generalized coordinates of an avatar be denoted by $\mathbf{q} = (\mathbf{r}_0^T, \theta^T)$, where $\mathbf{r}_0 \in \mathbb{R}^3$ is a root position and θ is a joint angle vector. We will assume that the virtual character has the same structure and size as the user, and that their poses are identical if they have the same generalized coordinates.

The retargeted pose should try to preserve both the original pose and the spatial relationship between the original pose and the object; the two objectives conflict increasingly with each other as the difference between the original and target objects increases, making it necessary to find a suitable compromise between the two objectives. It should be noted that, although the human motion is always physically correct and continuous, the retargeted motion may not be. Thus, we need to make sure that the resulting motion looks physically plausible and smooth. Regarding the physical plausibility, we focus only on the static balance because it is easily violated due to the spatial mapping and is keenly perceived by humans.

To satisfy these objectives, we define four energy terms (i.e., object interaction, pose preservation, balance, and smoothness); the retargeted pose is determined by minimizing the weighted sum of the four energies. The optimization is performed on a per-frame basis and, as will be explained later, the balance energy may create a temporally discontinuous pose. To guarantee continuity, our strategy is to let the avatar smoothly follow the *reference pose* $\mathbf{q}^*$, obtained by the optimization, instead of directly applying the reference pose to the avatar. We implement this simply by determining the avatar's pose as the interpolation between its pose at the previous time step $\tilde{\mathbf{q}}$ and the reference pose:

$$\mathbf{q} = \tilde{\mathbf{q}} + w(\mathbf{q}^* - \tilde{\mathbf{q}}) \tag{2}$$

where the parameter w $(0 < w < 1)$ controls how fast the pose follows its reference. The reference pose is determined in such a way as to minimize the following objective function:

$$\mathbf{q}^* = \arg\min_{\mathbf{q}} \lambda_p E_p + \lambda_d E_d + \lambda_\theta E_\theta + \lambda_b E_b \tag{3}$$

where λ controls the importance of each energy. The optimization problem is solved by using the Levenberg-Marquardt algorithm. The energy terms are described next.

Object interaction This energy term guides the avatar to occupy the portion of the interaction space corresponding to the space occupied by the user. This is achieved by letting the avatar follow the desired positions $\bar{\mathbf{p}}_i$ of every joint; these positions are determined by mapping the joint positions of a user in M_1 to M_2 using the spatial map f_M. Due to the differences between the interaction spaces, the projected joint positions $\bar{\mathbf{p}}_i$ usually violate the constraints inherent in the structure of the character, such as the bone length constraints, range of motion constraints, etc. Thus, we prioritize the joints such that the end-effector joints and the joints in contact with the objects are respected more than other joints by defining the energy as follows:

$$E_p(\mathbf{q}) = \sum_i \alpha_i ||\mathbf{p}_i(\mathbf{q}) - \bar{\mathbf{p}}_i||^2, \tag{4}$$

where the weight α_i is set larger for more important joints. In our implementation, the weights are set to 10 for the end-effectors and joints in contact with an object, and to 0.8 for the other joints.

Pose preservation The pose preservation energy term makes the avatar try to follow the human pose. Rather than preserving the joint space vector θ, we preserve the directions $\mathbf{d}_i$ of each body part in the Cartesian space to ensure that each body part can move in the desired direction $\bar{\mathbf{d}}_i$ independently from the body parts proximal to it:

$$E_d(\mathbf{q}) = \sum_i ||\mathbf{d}_i(\mathbf{q}) - \bar{\mathbf{d}}_i||^2. \tag{5}$$

Smoothness To consider temporal continuity between the retargeted poses, we add the smoothness energy:

$$E_\theta(\mathbf{q}) = \sum_i ||\theta_i - \tilde{\theta}_i||^2. \tag{6}$$

where $\tilde{\theta}_i$ is the i-th joint angle in the previous time step.

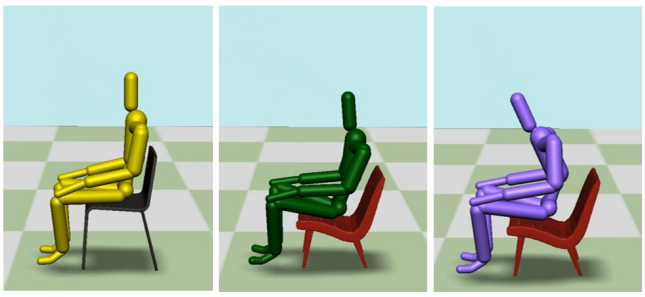

Fig. 6: Effect of the balance energy term. (Left) Original pose captured in the middle of a sitting motion, just before leaning against the back of a chair. (Middle) Retargeted pose without the balance term. The upper body leans back because of the tilted back of the target chair, which causes the loss of balance. (Right) Retargeted pose with the balance term. The trunk is adjusted to maintain balance.

Balance While the physical correctness of dynamic human motion is not accurately judged by the human eye, the static balance is keenly perceived by humans. Thus, *if* the human's pose is statically balanced, we encourage the avatar's pose to satisfy the static balance. The static balance is determined to be satisfied if $\mathbf{p}_{com}$, the ground projection of the center of mass (CoM), lies inside the support polygon, a convex hull made by the ground projection of contact points between the human (or avatar) and the object as well as ground.

During the online retargeting process, we examine whether the human's pose is statically balanced by checking the inclusion of the ground projection of the CoM in the support polygon. If the pose is not statically balanced (e.g., the human is about to sit on a chair), we set the balance energy to zero. If the pose is statically balanced, we use the following energy function to make sure that the $\mathbf{p}_{com}$ of the avatar lies inside the support polygon:

$$E_b(\mathbf{q}) = \begin{cases} 0, & \text{if the original pose is not statically balanced} \\ \max(0, sd(S(\mathbf{q}), \mathbf{p}_{com}(\mathbf{q})))^2, & \text{otherwise} \end{cases} \quad (7)$$

where $S(\mathbf{q})$ denotes the support polygon and $sd(\cdot,\cdot)$ is the signed distance function, which outputs a positive number if the point lies outside the support polygon. The effect of the balance energy term is shown in Fig. 6. Since the balance energy may cause a discontinuity in the reference motion when the state of the static balance of the original motion changes, Eq. (2) interpolates between the previous pose with the reference pose to create a smooth motion.

4 Experiments

We examined the effectiveness of the proposed method in a number of scenarios. The whole body motion is retargeted for the chair and motorcycle environments and the hand motion is retargeted for the glass, in either the VR or AR scenarios. The number of links and the degrees of freedom of the character are 22 and 69, respectively; those values for the hand model are 16 and 35.

4.1 Retargeting Sitting Motions

Chairs are the objects that humans make contact with most frequently in daily life, and they have diverse structures and shapes. We tested the proposed retargeting method with two chair models that have different heights and back angles, as shown in Fig. 7. The spatial map transfers an arbitrary trajectory in the source interaction space to a corresponding trajectory with respect to the target space, so that any motion about the original chair can be easily retargeted to the target chair. Figures 8 and 9 show the retargeted motions of the virtual avatar for sitting motions in different styles. In this experiment, $\lambda_p, \lambda_d, \lambda_\theta$, and λ_b are set to 1.3, 1.0, 0.9, and 1.5, respectively.

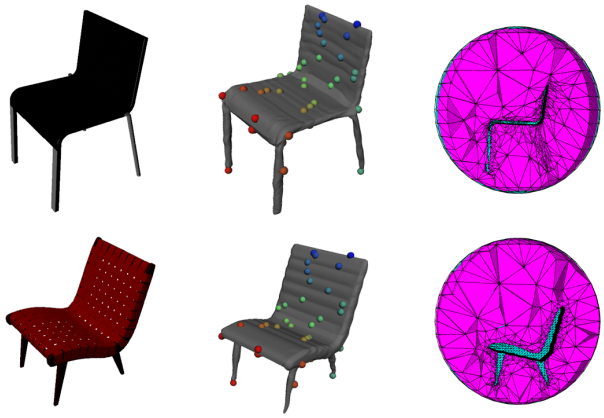

Fig. 7: (Left) Chair models with different heights and back angles. (Center) Internal boundaries of the interaction spaces. (Right) Cross sections of the interaction spaces. The triangles shown in the figures do not necessarily have one-to-one correspondences.

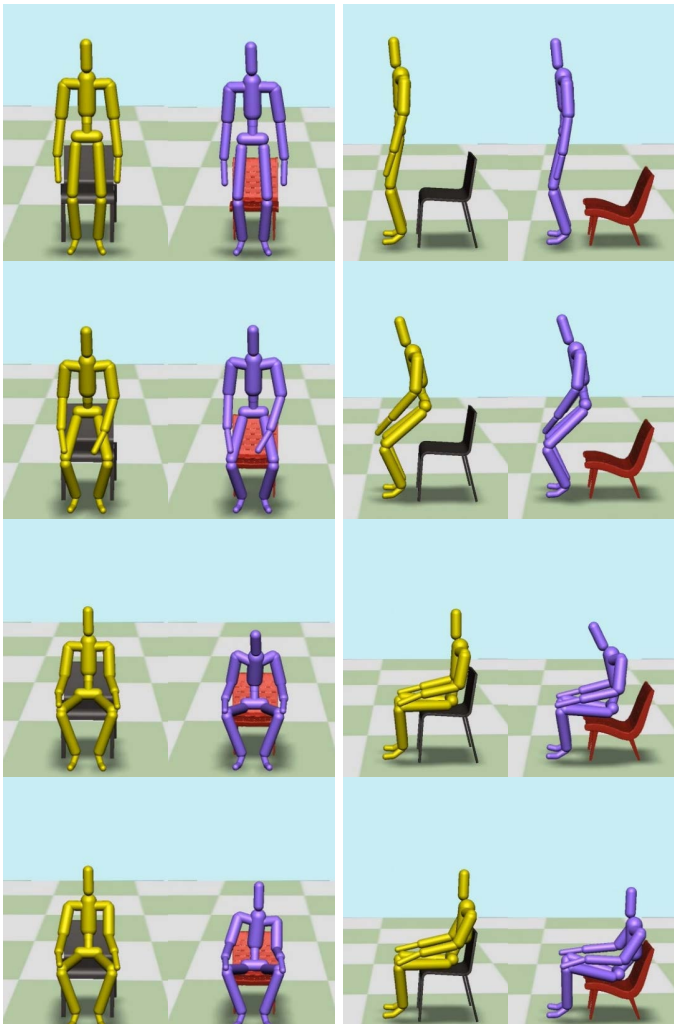

Fig. 8: Sitting motion, shown in front (left) and side (right) views, of the yellow character is retargeted to the purple character. The purple character creates appropriate contacts with the seat and back of the chair.

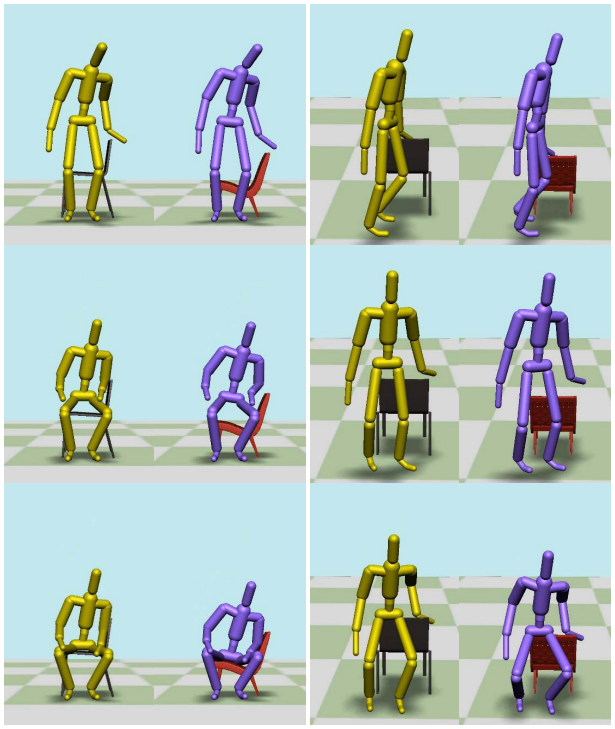

Fig. 9: Retargeting of the sitting motions in different styles for the same objects as those in Fig. 8. (Left) Side sitting with the arm contacting the back. (Right) Sitting on the back while contacting chair with the hip and the hand. The spatial relationships between the object and the character are appropriately preserved.

4.2 Retargting Riding Motions

We retargeted a riding motion on a motorcycle to other bikes (Figs. 10 and 11). While the shapes of the three objects are significantly different, they have common structural parts such as a saddle, two handlebars, and two pedals. Once the internal boundary meshes are created to preserve the correspondences between these parts, the motion retargeted by the spatial map represents suitably changed positions of the end-effectors such as the hips, hands, and feet. For this experiment, we used values of 1.5, 0.9, 0.8, and 0.8 for $\lambda_p, \lambda_d, \lambda_\theta$, and λ_b. The slightly increased weight for the object interaction helped the hands contact the handlebars properly.

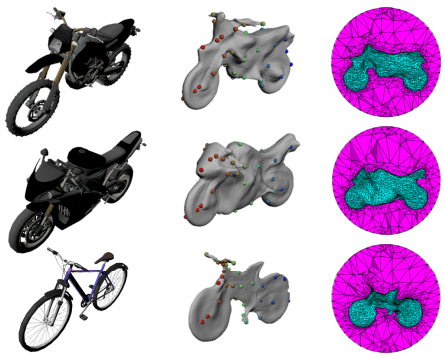

Fig. 10: Two motorcycles and a bicycle with different shapes, especially the locations of the handlebar and pedals. (Center) Internal boundaries with correspondences for sample points are shown in different colors. (Right) Cross sections of the tetrahedral meshes for the interaction spaces around the models.

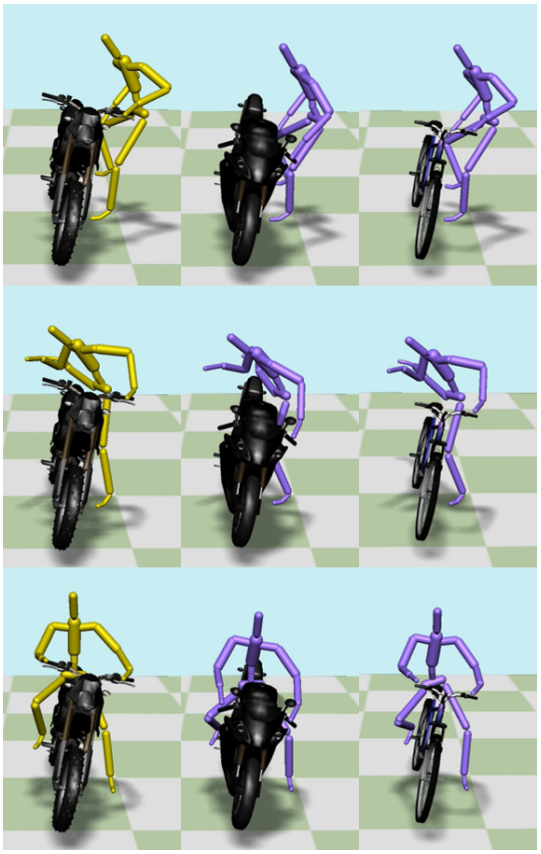

Fig. 11: Riding motion of the yellow character is retargeted to the purple characters. The positions of the hands and feet are transformed to the proper locations.

4.3 Retargeting Hand Motions

The proposed method can be applied not only to whole body motion but also to hand motion, as demonstrated by experiments with two glass models (Fig. 12). The hand motion of grasping a short wine glass is retargeted to a taller and thinner glass. The spatial map computes the corresponding positions of every finger joint and the wrist joint and Eq. (3) determines the optimal hand pose. In this experiment, $\lambda_p, \lambda_d, \lambda_\theta$, and λ_b were set to 2.5, 1.2, 1.0, and 0. It should be noted that the balance energy was ignored in retargeting the hand motion as it is not needed.

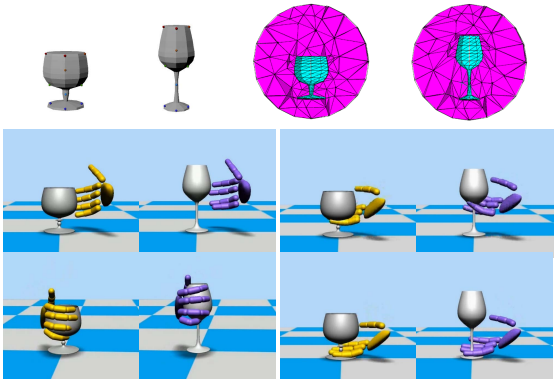

Fig. 12: (Top) Two glass objects with different heights and widths, and their interaction spaces. (Bottom) The motion of the yellow hand is retargeted to the purple hand. The positions of every finger and wrist are changed with the variation of the shapes.

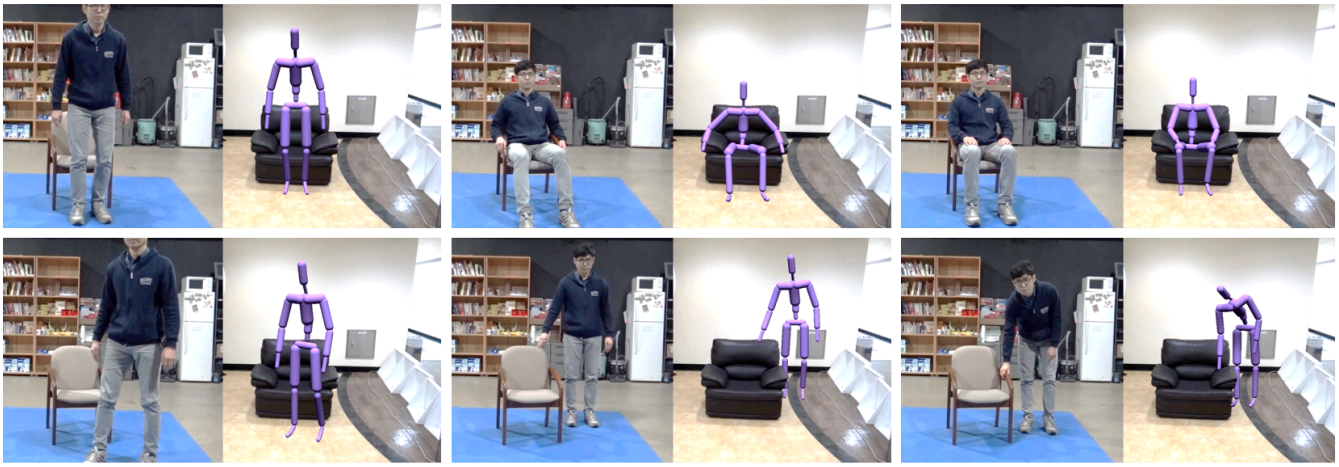

Fig. 13: Transferring a human-object interaction motion to an AR avatar in a different location.

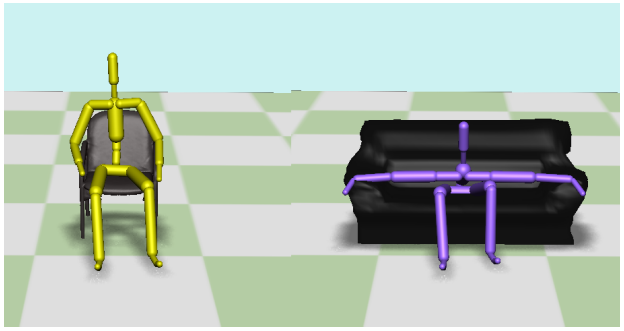

Fig. 14: A failure case due to significant difference of the object sizes.

Experiment	Object		Offline time (sec.)	Retargeting time min.~max. (sec./frame)
	#Vertices	#Tets		
Chair (Sec. 4.1)	1624	10.4K	3.647	0.02-0.072
Bikes (Sec. 4.2)	3521	24.0K	14.167	0.02-0.071
Wine glass (Sec. 4.3)	344	2.4K	0.379	0.03-0.05
Real chair (Sec. 4.4)	3448	23.4K	13.298	0.02-0.06

Fig. 15: Number of vertices of the internal boundary surface and the number of tetrahedrons in the interaction space. Offline time shows only the compute time required to find the interior correspondences (Sec. 3.1.3) and does not include the time for creating the interior surface mesh, which requires human intervention. The last column shows the time for online motion generation.

4.4 Motion Retargeting for an AR Avatar

Our next experiment involved applying the proposed method to the AR environment (Fig. 13). In the offline process, we obtain the 3D scan data of two different chairs by using a Kinect 2 device and Microsoft 3D Builder software; we then construct interaction spaces around them. The human motion is captured using the Kinect 2 device and the iPi Motion Capture software (ipisoft.com); its relative pose with respect to the reference frame of the chair mesh is computed and retargeted to the different chair using the proposed method. Figure 13 shows a screen capture in which a human's sitting motion on a chair in one location is transferred to an AR avatar that makes a natural sitting motion on a sofa in a different location. In addition, the hand contact of a human is transferred to a suitable contact position between the avatar and a sofa (Fig. 13, bottom).

4.5 Failure case

Even if the surface correspondence map between two objects is continuous and bijective, our method may not generate suitably retargeted motion if the sizes of the two objects differ significantly. Figure 14 shows a failure case in which a sitting pose on a chair is retargeted to a wide sofa. Since the hands contact the arm rests of the chair, our algorithm makes the retargeted pose reach the arm rests of the sofa, which creates excessive bending of the trunk joints. This absurd pose can be prevented if we enforce the admissible range of joint angles to the online retargeting process; however, in this case, the retargeted pose with a plausible trunk pose will extend its arms towards the arm rests, creating another awkward pose. A reasonably retargeted pose would not attempt to put the hands on the arm rests if such an action turns out to be impossible; the retargeted pose would rather include laying the hands comfortably on the seat of the sofa. This requires a somewhat intelligent decision and remains an interesting point for future work.

4.6 Performance

Figure 15 shows the complexity of the objects and the computation time for motion retargeting in our experiments. Naturally, the offline computation time depends greatly on the complexity of the objects. The time for the online process is independent of the object complexity but is determined by the number of iterations required for the optimization, which will be affected by the differences of the interaction spaces. Besides the online computation time shown in Fig. 15, there is an additional latency between the original and the retargeted poses of about 25 milliseconds due to the interpolation scheme (Eq. (2)). This latency is a cost for the smoothness of motions.

We performed our experiment using a desktop PC with a 3.4 GHz Intel i7 CPU and 8 GB RAM. Our software implementation uses only a single core, and there is room for faster online computation using parallel processing.

5 Conclusion and Future Work

In this paper, we presented a novel motion retargeting technique for two objects that have similar functions and shapes. The important feature of our method is the spatial map that defines the correspondences between any points in the 3D spaces around the respective objects, and thus identifies the desired locations of the avatar's body parts for any input motion of the human. The spatial map is constructed using the correspondences between the surfaces of the objects and the Laplacian mesh deformation. Once the spatial map is created offline, the motion

retargeting can be performed in real-time.

When a motion in one space is retargeted to another space of different shape, we cannot preserve both the speed of the motion and the spatial relation with the object. Since our method places higher priority on the latter, time warping may occur. This was not problematic in our experiments in which the motion retargeting has been done between similar objects, but could decrease the naturalness of the retargeted motion for a significantly different object.

There are many interesting venues for future work to improve the proposed method. The spatial map constructed by minimizing the Dirichlet energy gives plausible results for mild differences between objects, but it may produce undesirable results for exceedingly different shapes. In particular, a tetrahedron in the domain can become degenerate (i.e., zero volume) or folded over in the target space. These phenomena can occur when an object in the target space has an excessively concave shape. The folded-over region in the target space may induce jiggling motions, and thus need to be removed. Recent research in mesh processing presents methods to prevent such artifacts by adding hard constraints in the mesh deformation computation [1, 17, 25].

The performance of the spatial map depends heavily on the quality of the surface correspondences. Our method to generate watertight meshes to find correspondences between different shapes may require manual adjustments for complex surfaces, as described in Sec. 3.1.2. In fact, the problem of finding correspondences between surface meshes is one of the core problems in geometry processing, and an automatic method that robustly finds the surface correspondences will greatly enhance the utility of our method.

We have demonstrated that the proposed method is effective for retargeting motions between objects with similar shapes and functions. Perhaps the most interesting future work will be to extend the range of object pairs that allow for natural retargeting: how can we retarget poses appropriately if the surface correspondence map is not continuous or not bijective (e.g., a chair and a stool, a coffee cup and a cup without a handle)? As in the case of significant difference in object sizes (Fig. 14), for these more challenging problems we will need more advanced retargeting algorithms that consider not only the spatial relation but also the naturalness of the human behaviors.

ACKNOWLEDGMENTS

This work was supported by the Global Frontier R&D Program funded by NRF, MSIP, Korea (2015M3A6A3073743).

REFERENCES

[1] N. Aigerman and Y. Lipman. Injective and bounded distortion mappings in 3D. *ACM Trans. Graph.*, 32(4):106:1–106:14, July 2013.

[2] R. A. Al-Asqhar, T. Komura, and M. G. Choi. Relationship descriptors for interactive motion adaptation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, SCA '13, pages 45–53, 2013.

[3] M. Alexa, D. Cohen-Or, and D. Levin. As-rigid-as-possible shape interpolation. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '00, pages 157–164, 2000.

[4] 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, April 2013.

[5] M. Botsch and O. Sorkine. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):213–230, Jan. 2008.

[6] I. Chao, U. Pinkall, P. Sanan, and P. Schröder. A simple geometric model for elastic deformations. *ACM Trans. Graph.*, 29(4):38:1–38:6, July 2010.

[7] K.-J. Choi and H.-S. Ko. On-line motion retargetting. In *Computer Graphics and Applications, 1999. Proceedings. Seventh Pacific Conference on*, pages 32–42, 1999.

[8] M. Floater and K. Hormann. Surface parameterization: a tutorial and survey. In *Advances in Multiresolution for Geometric Modelling*, Mathematics and Visualization, pages 157–186. 2005.

[9] M. Gleicher. Retargetting motion to new characters. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '98, pages 33–42, 1998.

[10] E. S. L. Ho, T. Komura, and C.-L. Tai. Spatial relationship preserving character motion adaptation. *ACM Trans. Graph.*, 29(4):33:1–33:8, July 2010.

[11] T. Igarashi, T. Moscovich, and J. F. Hughes. As-rigid-as-possible shape manipulation. In *ACM SIGGRAPH 2005 Papers*, SIGGRAPH '05, pages 1134–1141, 2005.

[12] A. Jacobson, L. Kavan, and O. Sorkine-Hornung. Robust inside-outside segmentation using generalized winding numbers. *ACM Trans. Graph.*, 32(4):33:1–33:12, July 2013.

[13] 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.

[14] P. Kauff and O. Schreer. An immersive 3D video-conferencing system using shared virtual team user environments. In *Proceedings of the 4th International Conference on Collaborative Virtual Environments*, CVE '02, pages 105–112, 2002.

[15] M. Kim, K. Hyun, J. Kim, and J. Lee. Synchronized multi-character motion editing. *ACM Trans. Graph.*, 28(3):79:1–79:9, July 2009.

[16] H. Li, R. W. Sumner, and M. Pauly. Global correspondence optimization for non-rigid registration of depth scans. In *Proceedings of the Symposium on Geometry Processing*, SGP '08, pages 1421–1430, 2008.

[17] Y. Lipman. Bounded distortion mapping spaces for triangular meshes. *ACM Trans. Graph.*, 31(4):108:1–108:13, July 2012.

[18] L. Liu, L. Zhang, Y. Xu, C. Gotsman, and S. J. Gortler. A local/global approach to mesh parameterization. In *Proceedings of the Symposium on Geometry Processing*, SGP '08, pages 1495–1504, 2008.

[19] W. E. Lorensen and H. E. Cline. Marching cubes: A high resolution 3D surface construction algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH '87, pages 163–169, New York, NY, USA, 1987. ACM.

[20] A. Maimone and H. Fuchs. Encumbrance-free telepresence system with real-time 3D capture and display using commodity depth cameras. In *Proceedings of the 2011 10th IEEE International Symposium on Mixed and Augmented Reality*, ISMAR '11, pages 137–146, 2011.

[21] 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 *Virtual Reality (VR), 2013 IEEE*, pages 23–26, March 2013.

[22] J. Oh, Y. Lee, Y. Kim, T. Jin, S. Lee, and S.-H. Lee. Hand contact between remote users through virtual avatars. In *Proceedings of the 29th International Conference on Computer Animation and Social Agents*, CASA '16, pages 97–100, 2016.

[23] U. Pinkall and K. Polthier. Computing discrete minimal surfaces and their conjugates. *Experiment. Math.*, 2(1):15–36, 1993.

[24] P. Sandilands, V. Ivan, T. Komura, and S. Vijayakumar. Dexterous reaching, grasp transfer and planning using electrostatic representations. In *2013 13th IEEE-RAS International Conference on Humanoid Robots (Humanoids)*, pages 211–218, Oct 2013.

[25] C. Schüller, L. Kavan, D. Panozzo, and O. Sorkine-Hornung. Locally injective mappings. In *Proceedings of the Symposium on Geometry Processing*, SGP '13, pages 125–135, 2013.

[26] A. Sheffer, E. Praun, and K. Rose. Mesh parameterization methods and their applications. *Found. Trends. Comput. Graph. Vis.*, 2(2):105–171, Jan. 2006.

[27] H. Si. Tetgen, a Delaunay-based quality tetrahedral mesh generator. *ACM Trans. Math. Softw.*, 41(2):11:1–11:36, Feb. 2015.

[28] O. Sorkine and M. Alexa. As-rigid-as-possible surface modeling. In *Proceedings of the Symposium on Geometry Processing*, SGP '07, pages 109–116, 2007.

[29] S. Tak and H.-S. Ko. A physically-based motion retargeting filter. *ACM Trans. Graph.*, 24(1):98–117, Jan. 2005.

[30] H. Wang, K. A. Sidorov, P. Sandilands, and T. Komura. Harmonic parameterization by electrostatics. *ACM Trans. Graph.*, 32(5):155:1–155:12, Oct. 2013.

[31] Y. Wang, X. Gu, and S.-T. Yau. Volumetric harmonic map. *Commun. Inf. Syst.*, 3(3):191–202, 2003.