

Surface based motion retargeting by preserving spatial relationship

Zhiguang Liu, Antonio Mucherino, Ludovic Hoyet, Franck Multon

▶ **To cite this version:**

Zhiguang Liu, Antonio Mucherino, Ludovic Hoyet, Franck Multon. Surface based motion retargeting by preserving spatial relationship. MIG '18 - 11th Annual International Conference on Motion, Interaction, and Games, Nov 2018, Limassol, Cyprus. pp.1-11, 10.1145/3274247.3274507 . hal-01919065

HAL Id: hal-01919065
https://inria.hal.science/hal-01919065

Submitted on 16 Nov 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Surface based Motion Retargeting by Preserving Spatial Relationship

Zhiguang Liu
Inria, Univ Rennes, CNRS, IRISA
France
triggerliu@gmail.com

Antonio Mucherino
Univ Rennes, Inria, CNRS, IRISA
France
antonio.mucherino@irisa.fr

Ludovic Hoyet
Inria, Univ Rennes, CNRS, IRISA
France
ludovic.hoyet@inria.fr

Franck Multon
Univ Rennes, Inria, M2S
France
franck.multon@irisa.fr

ABSTRACT

Retargeting motion from one character to another is a key process in computer animation. It enables to reuse animations designed for a character to animate another one, or to make performance-driven be faithful to what has been performed by the user. Previous work mainly focused on retargeting skeleton animations whereas the contextual meaning of the motion is mainly linked to the relationship between body surfaces, such as the contact of the palm with the belly. In this paper we propose a new context-aware motion retargeting framework, based on deforming a target character to mimic a source character poses using harmonic mapping. We also introduce the idea of *Context Graph*: modeling local interactions between surfaces of the source character, to be preserved in the target character, in order to ensure fidelity of the pose. In this approach, no rigging is required as we directly manipulate the surfaces, which makes the process totally automatic. Our results demonstrate the relevance of this automatic rigging-less approach on motions with complex contacts and interactions between the character's surface.

CCS CONCEPTS

• **Computing methodologies** → **Animation**; **Optimization algorithms**;

KEYWORDS

Character Animation, Motion Retargeting

1 INTRODUCTION

Performance-driven animation of human-like characters has been widely explored, especially with the dissemination of cheap and easy-to-use motion capture devices. Users generally wish to control characters with various shapes, ranging from realistic to cartoon characters. Moreover, animations studios have gathered Terabytes of rigged animations accurately adapted to a given character in specific conditions. Being able to automatically reuse this data to make new characters move in the same way is essential to save time and associated costs. The problem is: how to accurately transfer a motion perfectly adapted to character *A* (either an actor/user, or a preprocessed virtual character) to character *B*, whereas *A* and *B* have different shapes and environments to interact with.

To answer this question, motion retargeting has been widely explored [Gleicher 1998]. However, most of previous approaches rely on a rigged skeleton, whereas the contextual meaning of the motion is mainly linked to relationship between skin surfaces. Hence, motions such as applauding, touching the belly, or crossing the legs, need manual editing to correct the errors introduced by skeleton-based motion controllers driven by inverse kinematics. Even though skeleton animation looks fine, manual post-processing is needed to correct artifacts introduced when animating the body surface. Moreover the rigging process is tedious and has to be done for each new character, leading to a reprocessing of all the required motions.

In order to tackle these limitations, this paper proposes a new approach to get rid of the rigging process and skeleton control, as suggested in previous works [Baran et al. 2009; Rhodin et al. 2014]. We aim at pushing the idea further to directly retarget a database of motions to new characters while ensuring that the contextual meaning linked to the body surface is preserved, without manipulation of rigged skeletons. As shown in Figure 1, we input the source calibration pose in Figure 1(a), a sequence of deformed source poses in Figure 1(c) extracted from the prerecorded database, and a target calibration pose in Figure 1(f). Then our method outputs a set of newly synthesized target poses for the target character, preserving the relationship between body surfaces of the deformed source poses in Figure 1(i).

Because of the versatility of computer graphics applications, poses such as those explored in this paper can typically be represented either by manifold mesh or, most of the time, by non-manifold mesh such as multiple-part objects, polygon soup or tetrahedral volumetric meshes. In order to reuse existing shape deformations and deal with various representations of shapes, we therefore propose a new surface-based motion retargeting framework. Our method is general and does not require the source and target calibration meshes to be compatible, namely, to have the same number of vertices, triangles, and connectivity. Our method needs to set a sparse set of joints inside the source and target calibration shapes. These joints can be provided by skeleton joint extraction algorithms automatically or obtained from user specification. Then we generate the retargeted animation based on the joints and available source mesh sequences. In contrast, the rigging process requires the users to perform operations such as setting up the skeleton, painting body part weight, and skeleton adjustment, which is much more tedious and time-consuming. In this sense, our method simplifies the work of the animation process. However, we assume there are clear semantic correspondences between the surface of the two 3D domains.

We use harmonic function as the mapping function, as previous work showed that it results in a smooth and detail-preserving mapping between two 3D domains [Ben-Chen et al. 2009b; Ferrari et al. 2012; Joshi et al. 2007; Zayer et al. 2005]. However, the Laplacian operator and its derivative can be quite complicated functions, which makes it hard to analytically solve these integrals. Inspired by the Method of Fundamental Solutions (MFS) [Fairweather and Karageorghis 1998; Li et al. 2009] that discretized the problem by using a linear combination of component functions, we construct an approximated harmonic mapping using MFS. Results presented in Section 8 show that our method using MFS produces very small reconstruction error.

The major contributions of this paper are summarized as follows:

- We propose a new unified framework for motion retargeting consisting in two steps: 1) extracting deformation descriptors and 2) synthesizing new deformations based on these deformation descriptors. During the deformation descriptor extraction process, we compute the affine transformation by solving a linear system. During the deformation synthesis process, we generate new poses by minimizing the difference between affine transformation descriptors and other constraints such as position, spatial relationship, and volume preservation.
- We propose a new structure to capture the spatial relationship among different body parts on the surface of a mesh, which we call *Context Graph*. By minimizing the deformation of the *Context Graph* between a source animated pose and a synthesized pose, we are able to preserve the contextual meaning for the newly synthesized pose.

The rest of the paper is organized as below. We first review the related work on motion adaptation and shape deformation in Section 2. Section 3 gives an overview of our method. Algorithm details on deformation analysis and synthesis will be discussed in Section 4 and 5, respectively. In Section 6, we describe the construction of our *Context Graph* and how it is deformed to preserve spatial relationships and body parts volume, followed by Section 7 with some implementation details. Experimental analyses and evaluations are conducted in Section 8. Finally, in Section 9, we conclude the paper, as well as discuss the limitations and future research directions.

2 RELATED WORK

2.1 Motion Retargeting

Motion retargeting techniques tackle the problem of adapting an animated motion from one character to another, and they have been widely explored to reuse existing motion capture data in computer graphics [Gleicher 1998; Kulpa et al. 2005].

Ho et al. [2010] proposed a new structure called interaction mesh to preserve the spatial relationships between body parts/objects in the scene by iteratively minimizing a Laplacian deformation energy. Their method generates sensible motions for close interactions as well as those without any contacts. However, their method is not suitable for real-time animation application as the algorithm performs the optimization over the whole animation sequence. Similarly, Al-Asqhar et al. [2013] propose to use relationship descriptors to describe the kinematics of the body parts by the weighted sum of translation vectors with respect to points sampled on the surface of the object in close proximity. Their method adapt the motion of a character to large updates of character morphologies and scenarios on-the-fly. However, re-position of contacts may be invalid due to large deformations. This work was then extended by Tonneau et al. [2016] [Al-Asqhar et al. 2013], who presented a scheme that enhances the detection and re-position of unnatural motion adaptations. Kim et al. [2016] develop a spatial map, which builds a bijection between two domains bounding the source and target objects. During the online motion generation process, the synthesized motion tries to preserve the spatial relationship between the source pose and an object. However, the algorithm needs to tetrahedralize two domains and requires the shape of source and target object to be very similar. Molla et al. [2018] propose an egocentric normalization of body surfaces to preserve the spatial relationships between the limbs and the other body parts. They test the proposed method a variety of characters with and without self-contacts. There are important differences between the method of Molla et al [2018], as they start from the joints with IK solver to reconstruct new postures, while our method works on the surface of the source and target meshes directly.

We are interested in generating realistic animations transfering existing motions of a character onto other characters with very different body morphologies. All of the above-discussed motion retargeting methods work on sparse joints. Thus, the speed and quality may decrease greatly when they are extended to a surface mesh, as the dimension of a surface mesh model is much higher than the sparse joints, e.g. a typical surface mesh model contains 20K vertices compared to approximately 50 sparse joints in a skeleton.

2.2 Surface Deformation

In this Section, we focus on direct surface manipulation deformation methods. Differential coordinates have been widely used to represent local details of a mesh. Lipman et al. [2004] reconstruct the surface by minimizing the difference of Laplacian coordinate

before and after the deformation. This method requires to explicitly find a correct rotation for the Laplacian coordinate of a local vertex, otherwise, distortion would have resulted. Similarly, Sorkine et al. [2004] propose a Laplacian representation by encoding each vertex relative to its neighborhood. However, unlike Lipman et al. [2004], the transformation of the differential coordinate for each vertex is implicitly computed from the transformation of a control handle. However, for transformations that involve large rotation, the deformation will suffer from anisotropic scales and need a post-process to refine it.

Zhou et al. [2005] enhance the Laplacian surface editing scheme by introducing a volumetric graph. Specifically, they embed a cubic lattice inside the shape, as well as an offset surface wrapping the exterior of the shape. The Laplacian editing framework is then employed to preserve the volume while keeping the local surface details. Loper et al. [2014] estimate body shape and pose by using a set of standard motion capture marker data. They successfully approximate the soft-tissue deformation as shape variations within the space of static human body shape. Wang et al. [2015] design linear subspaces for controlling shape deformation, which cuts down the time complexity of variational shape deformation methods. Methods such as [Boukhayma and Boyer 2017; Boukhayma et al. 2017] transfer or generate variations of 3D models by building a probabilistic low dimensional embedding of shape poses with Gaussian Process Models. Casas et al. [2013] propose the parametric motion graphs that enables real-time animation editing based on a database of temporally aligned mesh sequence reconstructions.

Most of direct surface manipulation techniques only work on the surface of a manifold mesh, which are hard to be applied to other mesh structures such as polygon soups and multi-component shape. In this paper, we apply the Laplacian deformation framework [Sorkine et al. 2004] to our *Context Graph* that preserves the volume and spatial relationship among different body parts. We approximate the harmonic mapping by using a linear combination of the fundamental solution of harmonic equation [Li et al. 2009; Poullikkas et al. 1998], which enables us to process both manifold and non-manifold shapes.

2.3 Space Deformation

A space deformation maps a source domain to a target domain inside the Euclidean space while preserving user-defined constraints. Cage-based deformation methods [Ben-Chen et al. 2009b; Joshi et al. 2007; Lipman et al. 2008] are such kinds of approaches that can be used to deform any type of shapes inside the cage, such as polygon soup, point cloud, and manifold mesh. Compared with direct surface based deformation that manipulates a set of dense points, controlling a sparse polyhedral cage simplifies the deformation process.

Shapes inside a cage can be represented as a linear combination of the vertices of the cage by barycentric coordinates [Floater et al. 2005; Huang et al. 2006]. However, barycentric coordinates are not positive for a concave domain, which may result in stretching in the resulting deformation. Harmonic coordinates [Joshi et al. 2007] are then explored as they are always positive inside the cage. Unfortunately, computing a closed-form solution for harmonic Dirichlet is hard. Boundary element method (BEM) [Sauter and Schwab 2010] offers a way to only discretize the boundary of the domain and derive the solution by solving a linear system. Later, in order to avoid computing the integral over the boundary, method of fundamental solution (MFS) [Li et al. 2009; Poullikkas et al. 1998] has been proposed to use only the fundamental solution in computing the solution. Lipman et al. [2008] introduce Green Coordinates for polyhedral cage in both 2D and 3D to extend barycentric coordinates, which leads to space deformation with a shape-preserving property. Ben-Chen et al. [2009b] propose a space deformation method that manipulates a sparse set of position and orientation constraints by finding an appropriate harmonic mapping of the space. They solve a non-linear minimization problem on the cage to find such a map that satisfies the user constraints.

In this paper, we employ harmonic mapping as our deformation function and compute the solution by a linear combination of component functions introduced in [Li et al. 2009; Poullikkas et al. 1998]. Thanks to MFS, our method is a meshless boundary method which does not require to enforce the connectivity of the mesh to function.

2.4 Deformation Transfer

Deformation transfer methods reuse existing animation sequences from the source shape to produce new animations. Given a source calibration shape, a target calibration shape, and a deformed source shape, the algorithm produces a deformed target shape that mimics the source shape deformation.

Sumner and Popovic [2004] present an efficient method to transfer animations from a source shape to a target shape, establishing a bijection mapping between the triangles of the source and target shapes. The deformation gradient for each triangle in the source shape is then computed, which transforms a source triangle into its corresponding target triangle. One limitation is that only manifold mesh can be processed. Zhou et al. [2010] enhance this framework by connecting all adjacent components with a minimal spanning tree. In contrary, Ben-Chen et al. [2009a] project the deformation into a linear space by transfering a sparse set of deformation gradients inside a cage. However, their method requires constructing two polyhedral cages for both the source and target shape. Jin et al. [2018] propose the aura mesh, which is a volumetric mesh that surrounds a character's skin, to preserve the spatial relationships between two characters. The interaction between two characters is expressed with respect to the collision between the skin mesh of one character and the aura mesh of the other. The method is fast and produces interaction motions for characters with different sizes and skin shapes. However, their method requires a rigged skeleton and define the weights of the skin surface whereas our method just needs to specify several joints inside the characters to transfer the deformation descriptors.

Most of these deformation transfer methods just transfer the deformation descriptor of the source shape to the target shape without considering the spatial relationship between different body parts of the source shape. To solve this problem, we define a *Context Graph* on the surface of source/target shape, used to describe the spatial relationship among different body parts.

Compared to the Green coordinates model [Lipman et al. 2008] that needs to move the cage to deform the bounding space, our

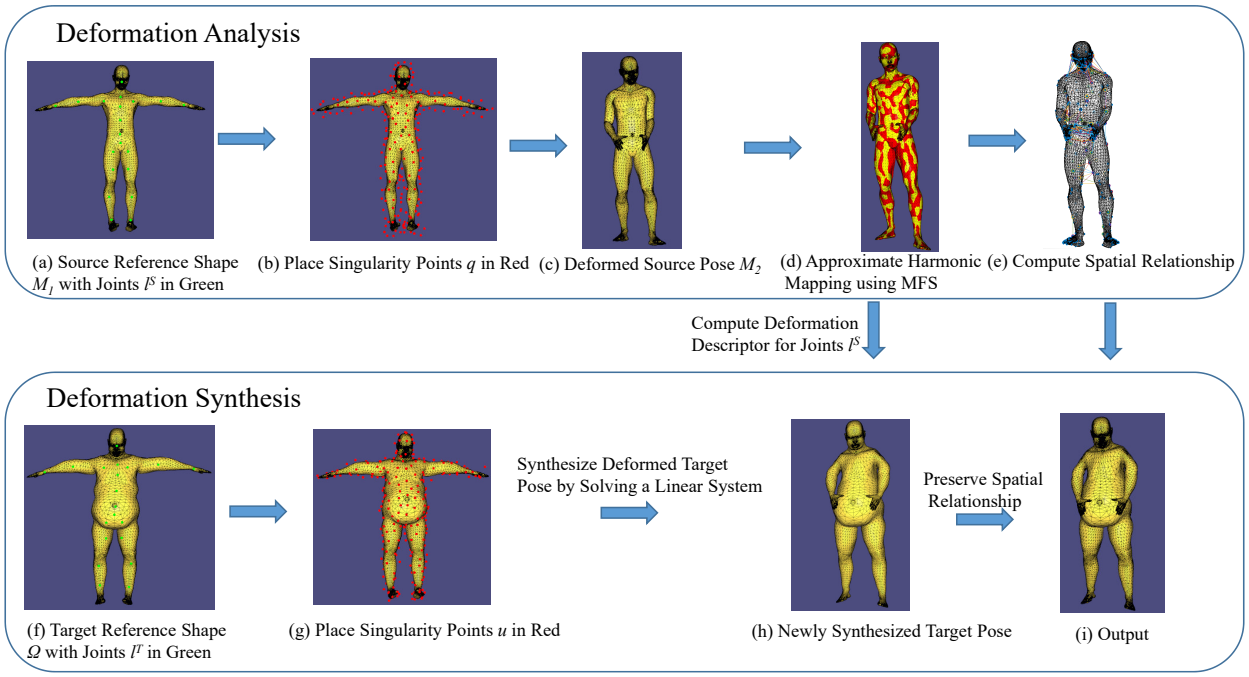


Figure 1: Overview of synthesizing new poses using harmonic mapping. During the deformation analysis process, we calculate the deformation descriptor of the source calibration pose. During the deformation synthesis process, we compute the coefficients of approximating harmonic mapping by assuming that the target calibration pose perform similar deformation to the source calibration pose. We further constrain the synthesized motion to have an as similar spatial relationship as that of the source deformed pose by adding Laplacian framework.

method directly builds volumetric mappings between different 3D domains. Our method is somewhat similar to spatial deformations such as [Ben-Chen et al. 2009a,b], in the sense that we also use the Jacobian of the mapping function as the affine transformation descriptor. However, there are important differences: (1) Our method does not need a cage and we just need to compute some offset points close to the boundary surface of the pose using methods such as marching cubes and Silhouette Clipping [Lorensen and Cline 1987; Sander et al. 2000] and this is completely hidden to the users. In addition to the shape to be deformed, Ben-Chen et al. [2009a; 2009b] require the user to offer a cage surrounding the shape. Creating a cage is not a trivial work. Although some algorithms could simplify this task, the user may still need to do some manual work to construct a high quality cage. (2) We have an analytical least-squares solution to the harmonic reconstruction optimization problem. Methods of [Ben-Chen et al. 2009a; Lorensen and Cline 1987] need multiple 'local/global' iteration steps to converge to a satisfing solution. (3) We concentrate on preserving the spatial relationship among different body parts using the proposed *Context Graph*, which may also benefit other spatial deformation methods such as [Ben-Chen et al. 2009a; Zhou et al. 2010].

3 ALGORITHM OVERVIEW

As shown in Figure 1, our method consists of two phases: the deformation analysis and synthesis. During the deformation analysis process, if we do not have the skeleton together with mesh data, we first compute the joints of both the source and target calibration poses M_1 and Ω as shown in Figure 1(a) and (f), using skeleton extraction methods such as [Au et al. 2008; Cao et al. 2010]. Secondly, as shown in Figure 1(b), we generate an offset surface enclosing the source calibration pose M_1. The points uniformly sampled on this offset surface are auxiliary points used for approximating the mapping function, and we call them singularity point. Then we approximate the harmonic function using the singularity points to map the source calibration shape to the deformed source shape using MFS [Fairweather and Karageorghis 1998; Li et al. 2009]. We use the Jacobian matrices of the harmonic function at these joints as the affine transformation.

During the deformation synthesis process, similar to the deformation analysis step, we first place some offset points close to the target calibration shape as seen in Figure 1(g). Secondly, we assume that the joints of the target calibration pose Ω experience similar deformations than the joints of the source calibration pose M_1 such that we can calculate the coefficients for approximating the harmonic mapping function of the target calibration pose. Thirdly, in order to keep the spatial relationships similar to the deformed source pose M_2 and reduce the loss of volume, we introduce and construct a *Context Graph*, and employ the Laplacian deformation framework to further constrain the system. Finally, we synthesize the new posture by solving a linear system.

4 EXTRACTING DEFORMATION FROM SOURCE SHAPE HARMONIC MAP

4.1 Solving Harmonic Mapping Using MFS

Recent work [Ferrari et al. 2012; Joshi et al. 2007; Zayer et al. 2005] shows that harmonic functions generate pleasing deformations. We therefore choose them as our deformation mapping function, and formulate our harmonic mapping problem as follows.

Let M_1 be an open region of R^3 with a smooth boundary ∂M_1, and M_2 be another open region of R^3 with a smooth boundary ∂M_2. We aim at constructing a mapping $f : M_1 \rightarrow M_2$ between two boundaries with the same topology, that is, the objects have pairs of corresponding boundary surfaces, such that

$$\begin{aligned} &\Delta f(P) = 0, \quad P \in M_1 \\ s.t. \quad &f(P) = P', \quad P \in \partial M_1, P' \in \partial M_2 \end{aligned} \tag{1}$$

where the Δ is the Laplacian operator defined as $\frac{\partial f}{\partial x^2} + \frac{\partial f}{\partial y^2} + \frac{\partial f}{\partial z^2}$.

Since the Laplacian operator Δ is linear, we can construct a linear subspace of functions from R^3 to R^3 using harmonic mappings. Inside this linear space, We would like to find a mapping function that both preserves the local details and the spatial relationships. However, the Laplacian operator Δf and its derivative are too complex to be solved analytically. We follow previous methods, such as [Li et al. 2009], and discretize the boundary using the Method of Fundamental Solutions (MFS) [Fairweather and Karageorghis 1998]. Then the solution to Equation 1 can be approximated by a linear combination of component functions of harmonic equations:

$$f(P) = K^s(P, Q)W^s \tag{2}$$

where

- $P = [p_1, p_2, ..., p_{N_c^s}]$ is a matrix of $3 \times N_c^s$ concatenating all N_c^s surface points of source calibration shape ∂M_1.
- $Q = [q_1, q_2, ..., q_{N_a^s}]$ is a $3 \times N_a^s$ matrix consisting of all N_a^s three dimensional singularity points outside ∂M_1. More details on placing q can be found in Section 4.3.
- $k(p, q) = 1/(4\pi \|p - q\|)$ is the component function of the harmonic function f, which depends on the distance between a point $p \in P$ and singular point $q \in Q$. To simplify the notation, we use $K^s = K^s(P, Q)$ to represent the matrix of components function of size $N_c^s \times N_a^s$ and each row $K^s(p, Q)$ consists of N_a^s components for a point $p \in P$.
- $W^s = [w_1^T, w_2^T, ..., w_{N_a^s}^T]$ is a $N_a^s \times 3$ matrix of harmonic coefficients associated with these N_a^s singularity points, which offers the degree of the freedom to control the boundary fitting. The length of row vector w_i^T is 3 and each of its element corresponds to one dimension of point $p \in P$.

The vanishing Laplacian operator Δ on f is enforced by the component functions $k(p, q)$, we only need to ensure that the function satisfies the boundary condition. To perform boundary fitting on each surface point $p \in \partial M_1$, we evaluate $f(p)$ using Equation 2. The boundary constraints are then $f(p) = \hat{p}$ where $\hat{p}$ is the corresponding point of p laid on the boundary surface of deformed source pose. Enforcing these constraints on all collocation points reduces to a linear system:

$$A^s W^s = b^s \tag{3}$$

where A^s is a $N_c^s \times N_a^s$ coefficient matrix with element $A_{ij}^s = k(p_i, q_j)$, N_c^s is the number of surface points p while N_a^s is the number of singularity points q, b is the image of p on the boundary of deformed source pose ∂M_2 with size $N_c^s \times 3$. Therefore, the above fitting process reduces to a linear system on three axis directions. This is an over-determined linear least-squares problem as usually $N_a^s \ll N_c^s$, which results in a closed-form solution [Sorkine et al. 2004].

4.2 Calculating Descriptor of Affine Transformation

The Jacobian matrix of the mapping function can be used to prescribe an affine transformation for any point $p \in (M_1 \cup \partial M_1)$. Once we have approximated the source deformation f as a linear combination of component functions of Laplacian equation, the Jacobian of the deformation for any point p inside the domain M_1 can be computed using the gradients of the Laplacian equation. The transposed of the Jacobian of the source deformation of f at the point p is:

$$J^s(p)^T = D^s W^s, \; p \in \partial M_1 \cup M_1 \tag{4}$$

where D^s is a matrix whose i^{th} column is the derivative of the kernel and W^s are linear coefficients of f computed in Equation 3. D^s is a $3 \times N_a^s$ matrix and W^s is a $N_a^s \times 3$ matrix.

The Jacobian of the source deformation on each point p prescribes the affine transformation which we will transfer to the target calibration pose. There are many choices for the selection of Jacobian, e.g. we can choose any surface point's Jacobian to be transferred. However, as suggested by Ben-Chen et al. [2009b], points on the medial axis of the domain undergo the smallest distortion. On the other hand, densely sampling points inside source domain M_1 would result in high computation and even artifacts [Zhou et al. 2010]. Therefore, similar to Ben-Chen et al. [2009a; 2009b], we choose the Jacobian of each joint to represent the affine transformation.

4.3 Placing Singularity Points

We apply the MFS to solve the partial differential equation defined in Equation 1. The kernel function defined in Equation 2 is not well defined if a point p inside the shape equals to the singularity point q. On the other hand, the vanishing Laplacian operator cannot always be guaranteed if the singular points are sampled inside the calibration shape. Therefore, we have to place singularity points outside the surface of the calibration shape. The locations of the singularities are either preassigned or determined along with the coefficients w_i of the component functions so that the approximate solution satisfies the boundary conditions as well as possible. If the locations of the singularities are to be determined, the resulting minimization problem is nonlinear and time-consuming. Therefore, in this paper, we preassigned the positions of singularity points and find the approximate solution by the least square fit of the boundary surface using Equation 1. In the current implementation, we use 260 singularity points on average.

We follow the method of Li et al. [2009] to place singularity points uniformly on an offset surface $\partial \hat{M}_1$ outside the boundary surface ∂M_1 as shown in Figure 1(b). Here is the process of how to sample singularity points: (1) We collapse edges of ∂M_1 until the

desired number of faces is achieved but ensures that new vertices are placed outside all previous meshes [Sander et al. 2000]. (2) We uniformly sample some points $Q = [q_1, q_2, ..., q_{N_a^s}]$ on the boundary points of $\partial\hat{M}_1$ by furthest point relaxation as described in Fast Automatic Skinning Transformations [Jacobson et al. 2012]. Similarly, the collocation points are uniformly sampled from the boundary points of ∂M_1.

5 PRODUCING NEW POSTURES BY FINDING APPROPRIATE HARMONIC MAP

Once the deformation descriptor has been extracted from the source deformation, we can apply it to the target calibration pose. We deform the target calibration pose with the same affine transformation (Jacobian constraints) extracted from Equation 4. First, we calculate our deformation descriptor of the harmonic functions of the target calibration pose, with a set of unknown coefficients w^t. Then, we compute the deformed target calibration pose as a linear combination of component functions of Laplacian equation by Equation 6.

5.1 Computing a Raw Pose

Let the target calibration shape be given as a set of N_c^t points $V = \{v_1, v_2, ..., v_{N_c^t}\}$, and corresponding singularity points $U = \{u_1, u_2, ..., u_{N_a^t}\}$. Given the Jacobians J^s at the m joints $\mathcal{J}^s$ inside the deformed source pose, we want to minimize the Jacobian differences between the deformed source pose and newly synthesized target pose by:

$$E_{Affine} = \left\| D^t W^t - J^s \right\|_F^2 \tag{5}$$

where W^t is the unknown linear coefficient of target calibration shape with size $N_a^t \times 3$. $D^t = \nabla K(\mathcal{J}^t, U)$ is the gradient of target kernel function on m joints inside target calibration shape with size $3m \times N_a^t$ and $\mathcal{J}^t$ is a $m \times 3$ matrix that concatenating positions of m joint inside target calibration shape. J^s is a $3m \times 3$ dimensional matrix concatenating matrices of all m affine transformations, computed by Equation 4, of the source shape deformation.

This optimization problem for computing the coefficients W^t can be solved by a linear system. Equipped with the coefficients W^t, the deformed target pose is given as a linear combination of the component function of the harmonic function for the target calibration pose as:

$$\hat{V} = K^t W^t \tag{6}$$

where $K^t = K^t(V, U)$ is a matrix of size $N_c^t \times N_a^t$ that each row computes the inverse of the distance between the point $v \in V$ and all singularity points u_i.

5.2 Constraints

The problem defined in Equation 5 is under-determined because the number of joints m is much smaller than the number of singularity points N_a^t. On the other hand, we need to predetermine a position for any vertex to compute a unique solution. Lastly, we introduce the bone length constraint such that the resulting poses are physically plausible. More details on solving the system with constraints are discussed in Section 7.2.

Surface Smoothness Constraint

Here, we follow [Ben-Chen et al. 2009b] to assume that adjacent points on the surface will be transformed similarly. Thus, the second derivative of the mapping function tends to be very small. In order to make the surface as smooth as possible, we constrain the Hessian matrix $H(f^t)$ of the target mapping function $f^t = K^t W^t$ for a newly synthesized surface to be as small as possible by:

$$E_{Smooth} = \left\| H(K^t W^t) \right\|_F^2 \tag{7}$$

where E_{Smooth} is a linear function of W^t with the coefficients of size $3N_c^t \times N_a^t$.

Positional Constraint

The gradient of a deformation function can only specify the rotation and scaling transformations. Thus, to determine all the absolute positions, we predetermine the position of any point, e.g., v_1, as $v_{position}$ by:

$$E_{Position} = \left\| K^t(v_1, U) W^t - v_{position} \right\|^2 \tag{8}$$

where $K^t(v_1, U)$ is a row vector of length N_a^t and U are singularity points of target calibration shape.

Bone Length Constraint

In an ideal character animation, the bone length between two joints is fixed. We introduce the bone length constraint to avoid stretching the synthesized shape.

Specifically, the bone length l_{ij} between joint $\mathcal{J}_i^t$ and $\mathcal{J}_j^t$ for target calibration shape is prescribed by $(\left\| \mathcal{J}_i^t - \mathcal{J}_j^t \right\| - l_{ij})^2$. We compute the Jacobian Linearization of this non-linear system

$$(\left\| K_i^t W^t - K_j^t W^t \right\| - l_{ij})^2 \tag{9}$$

by:

$$E_{Bone} = J_{Bone} W^t + C \tag{10}$$

where J_{Bone} is the Jacobian matrix of size $3\mathcal{B} \times N_a^t$, $\mathcal{B}$ is the number of bones, and C is a constant matrix.

6 CONSTRUCTING CONTEXT GRAPH

In this Section, we compute the *Context Graph* to preserve the volume and spatial relationship between body parts. We first construct two compatible *Context Graphs* and then simplify the graphs to speed up the computation. Lastly, we deform the graph constructed on the target calibration pose into the one built on the deformed source pose such that we preserve the spatial relationships.

We project some singular points onto the boundary of source calibration pose M_1 to construct a graph such that spatial relationship of a particular pose could be preserved. Here, we project the N_a^s singular points of M_1 onto the surface $P \in \partial M_1$, denoted as $P_a = \{p_1, p_2, ..., p_{N_a^s}\}$. We connect all the points P_a on each deformed source pose to build a fully connected *Context Graph* G^s, which enable us to capture the spatial relationships among points on the boundary surface of deformed source pose.

We use the non-rigid iterative closest point (ICP) algorithm [Li et al. 2008] to find N_a^s corresponding points $V_a \in \partial\Omega$ on the boundary of target calibration poses Ω. Connecting all the points V_a found

on $\partial\Omega$ would result in a compatible graph G^t as the one G^s built on the deformed source pose.

6.1 Pruning the Context Graph

The fully connected *Context Graph* G^s contains too many redundant edges that will greatly slow down the deformation. Thus, we need to prune the *Context Graph*, which we do according to the following rules:

(1) As we can consider human poses to be close to rigid, the distance between any two given points on the same body part rarely changes during the deformation. For this reason, edges connecting two points laid on the same body-part are discarded, and we therefore only consider points distributed on different body parts. To do so, we apply K-means to cluster the body parts by the joints extracted either from the users or the skeleton joint extraction algorithms such as [Au et al. 2008].
(2) As we focus on close interactions and assume that nodes which are far away from each other on the graph have little influence on the others, we remove edges longer than a given threshold. Currently, we set the threshold to 50% of the calibration shape height.
(3) As we only consider interactions on the surface of body parts, edges passing through any body part would be invalid. Edges intersecting with the interior of the mesh are therefore discarded.

Figure 2 shows an example of constructing and pruning *Context Graph*.

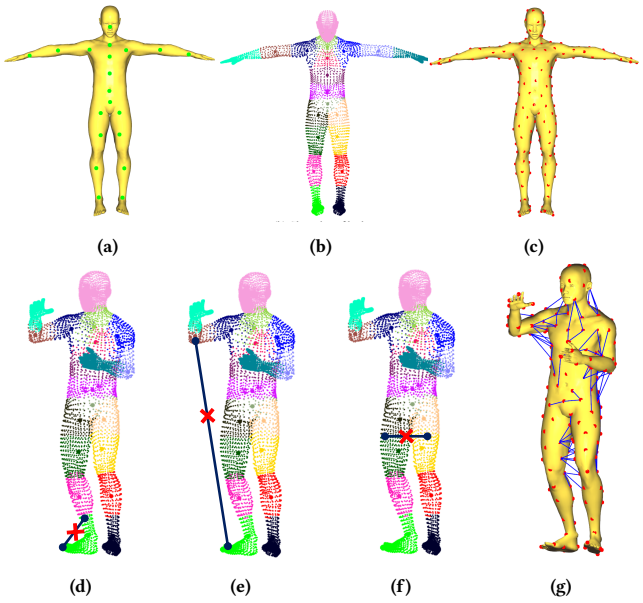


Figure 2: Pruning the *Context Graph*. (a) Source calibration pose with joints. (b) Clustering of body-parts. (c) Projecting singular points onto the boundary surface and extracting them. (d) Removing edges on the same body part. (e) Removing long edges. (f) Removing edges interpenetrating the body parts. (g) The simplified *Context Graph*.

6.2 Deformation of Context Graph

There are many choices to combine the existing differential deformation frameworks (reviewed in Section 2). For a clear presentation, we follow in this paper the method of [Sorkine et al. 2004] to compute Laplacian coordinate, which implicitly computes the affine transformations from the translation of a point.

The Laplacian coordinates of points P_a are defined as $L(P_a) = LP_a = (I - Diag^{-1}Adjacent)P_a$ with the uniform weights, where *Adjacent* is the adjacent matrix of *Context Graph*, *Diag* is the degree matrix, and L is the $N_a^s \times N_a^s$ Laplacian operator matrix constructed from the *Context Graph* of deformed source mesh. The deformation of target *Context Graph* that preserves spatial relationship is computed by:

$$E_{Spatial} = \left\| T_{Spatial} L_{Spatial}(P_a) - L_{Spatial}(\hat{V}_a) \right\|_F^2 \qquad (11)$$

where points $\hat{V}_a \in \partial\Omega$ are the node of spatial *Context Graph* that correspond to $P_a \in \partial M_1$, which are extracted from the surface of target calibration shape Ω, and $L_{Spatial}(P_a)$ are the Laplacian coordinates computed from the *Context Graph* of the source deformed shape. The coefficients of $T_{Spatial}$ are the linear combinations of unknown W^t [Sorkine et al. 2004], which implies $E_{Spatial}$ is a linear function of the unknown W^t with size $N_a^s \times N_a^t$.

Similarly, we define the deformation of volume *Context Graph* that preserves the volume of the target calibration shape by:

$$E_{Volume} = \left\| T_{Volume} L_{Volume}(V) - L_{Volume}(\hat{V}) \right\|_F^2 \qquad (12)$$

where V is a $N_c^t \times 3$ matrix that concatenating the position of point $v \in \partial\Omega$. Similar to $T_{Spatial}$, the coefficients of T_{Volume} are linear functions of unknown W^t with size $N_c^t \times N_a^t$. $L_{Volume}(\hat{V})$ are the Laplacian coordinates computed from the volume *Context Graph* of the target calibration shape

7 IMPLEMENTATIONS

7.1 Solving the Linear System for Extracting Source Deformation

As discussed in Section 4, the deformation analysis process can be optimized by solving a linear system $A^s W^s = b^s$. Each row in the coefficient matrix A^s is the value of the kernel functions evaluated on a collocation point and all the other singularity points. However, the resulting matrix A^s is not sparse because the value of kernel functions is almost never zero. Thus, we cannot apply a sparse solver to speed up the computation. As studied in [Li et al. 2009; Ramachandran 2002], such a matrix might be ill-conditioned and will not produce a stable solution using regular linear system solvers such as Gaussian elimination. We follow previous methods of [Li et al. 2009; Ramachandran 2002] to use Singular Value Decomposition (SVD) because it approaches accurate and stable results even when the coefficient matrix is highly ill-conditioned. The matrix A^s is then decomposed as $A^s = \mathcal{U}\mathcal{S}\mathcal{V}^T$ and the inverse of A^s is computed as $A^{s-1} = \mathcal{V}\mathcal{S}^{-1}\mathcal{U}^T$.

Once we have precomputed the inverse of A^s, we are able to perform a fast fitting of new boundary condition b_*^s (new pose in

the deformed source pose sequence) by:

$$W_*^s = A^{s-1} b_*^s = \mathcal{V}\mathcal{S}^{-1}\mathcal{U}^T b_*^s \tag{13}$$

Then the transpose of the deformation descriptor of joints $\mathcal{J}^s \in M_1$ that fits a new boundary b_*^s is computed by:

$$J^s(\mathcal{J}^s)^T = D^s W_*^s \tag{14}$$

7.2 Solving the Linear System for Synthesizing a New Pose

We define the retargeting problem as the sum of Equation 5, 7, 8, 11, and 12 by:

$$\begin{aligned} E = E_{Affine} + \lambda_{Smooth} E_{Smooth} + \lambda_{Position} E_{Position} \\ + \lambda_{Bone} E_{Bone} + \lambda_{Spatial} E_{Spatial} + \lambda_{Volume} E_{Volume} \end{aligned} \tag{15}$$

where $\lambda_{Position}, \lambda_{Smooth}, \lambda_{Bone}, \lambda_{Spatial}$, and λ_{Volume} are the weights for the energy terms. In our implementation, they are empirically set to be 0.01, 10, 1, 0.8, and 1, respectively. There are some principles to tune the values of the weights. The bone length and volume constraints are hard because we do not want to change the length of bone or lose the volume of the shape during the deformation. The position constraint makes the linear system solvable and it should be with a large weight. The smooth constraint should be small, otherwise, the shape would be flattened out. The spatial constraint should be large such that the spatial relationship between body parts can be preserved

We rewrite Equation 15 as a linear system by:

$$A^t W^t = b^t \tag{16}$$

where W^t is the unknown coefficient matrix of size $N_a^t \times 3$, A^t and b^t represent a concatenation of matrices from Equation 5, 7, 8, 11, and 12 as:

$$A^t = \begin{bmatrix} A_{Affine} \\ \lambda_{Position} A_{Position} \\ \lambda_{Bone} J_{Bone} \\ \lambda_{Smooth} A_{Smooth} \\ \lambda_{Spatial} A_{Spatial} \\ \lambda_{Volume} A_{Volume} \end{bmatrix}, b^t = \begin{bmatrix} J^s \\ \lambda_{Position} v_{position} \\ -\lambda_{Bone} C \\ \vec{0} \end{bmatrix} \tag{17}$$

where b^t is the right hand side matrix containing the affine transformation J^s, position $v_{position}$, and bone length C constraints. The remaining elements are all 0. $A_{Affine} = D^t(\mathcal{J}^t)$ is a coefficient matrix of size $3m \times N_a^t$ defined in Equation 5, where m is the number of joints $\mathcal{J}^t$ and N_a^t is the number of singularity points of the target calibration pose. $A_{Position} = K^t(v_1, U)$ is a row vector of length N_a^t defined in Equation 8. A_{Smooth} is a Hessian matrix of size $3N_c^t \times N_a^t$ defined in Equation 7, where N_c^t is the number of boundary points on the target calibration pose. $A_{Spatial}$ and A_{Volume} are the linear coefficients defined in Equation 11 and 12 that preserve the spatial relationship and volume for the synthesized pose. J^s is the Jacobian matrix of size $3m \times 3$ from m joints of deformed source pose defined in Equation 5 and $v_{position}$ is a row vector of length 3.

The analytical least-squares solution is defined by the following normal equation:

$$W^t = ((A^t)^T A^t)^{-1} (A^t)^T b^t \tag{18}$$

Once we calculate the unknown coefficient W^t, we produce a new pose by using Equation 6.

8 EXPERIMENTAL RESULTS

8.1 Harmonic Mapping Error

In order to quantitatively analyze the correctness of the proposed method, we use the source calibration shape as the target calibration shape. Thus, we treat the source animation as our ground truth data. We compute the average approximation error for each pose from all the animation sequences by:

$$Error_{Mapping}(P) = 1/m \sum_{i=1}^{m} \left\| f_i(P) - \hat{P}_i \right\| \tag{19}$$

where function f_i maps each vertex $P \in \partial M_1$ of i^{th} frame onto deformed source pose $\hat{P}_i$ and m is total number of frames.

To simplify the calculation of reconstruction error, we normalize each calibration shape to a height of 1. Figure 3 shows the resulting error for a set of punching motions. Each pose contains around 18K vertices. As shown in Figure 3b-3d, the reconstruction error is coded with color from blue (0) to red (0.04). We can observe that our method generates meshes close to the ground truth meshes. Note that, as illustrated in Table 1, the average error per pose over the source animation sequence is very small, which shows that the approximated harmonic function is able to map from one domain onto another one precisely.

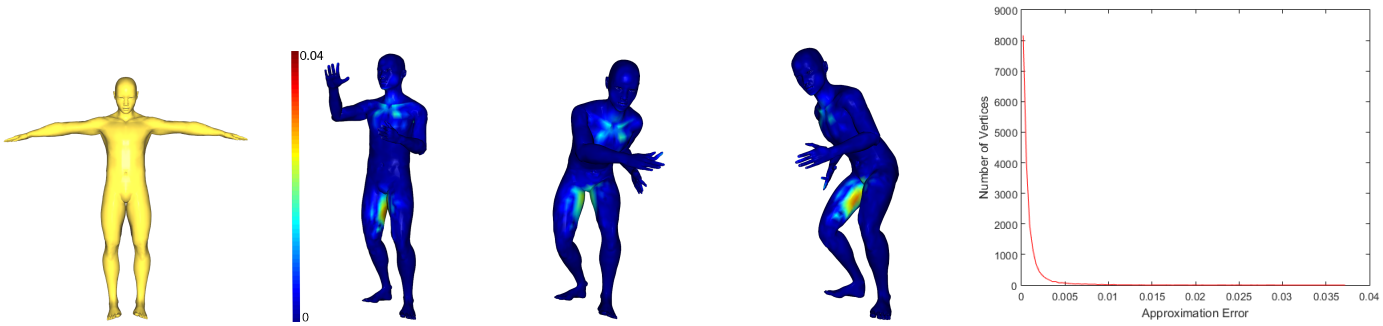
Table 1: Average reconstruction errors over different motion sequences

Motion Name	Number of Frames	Number of Vertices	Mean Error	Std
Punching	371	17996	0.021	0.14
Sitting on a chair	490	19296	0.125	0.26
Tree pose	329	19296	0.193	0.21

8.2 Retargeting Results

All experiments were performed on a PC with an Intel core i7-7700HQ CPU and with 16GB of RAM. Our method takes around 4 seconds to synthesize each frame. The deformation analysis takes around 1 second. We first demonstrate the influence of different terms on producing the resultant poses. As shown in Fig. 5, bone length constraint defined in Equation 10 ensures that the deformation of the target calibration shape is physically correct, while preserving the volume during the deformation prevents unnatural morphological changes. As illustrated in Fig. 4(b), the belly in blue color is collapsed if we do not add the volume constraint. In addition, by preserving the *Context Graph*, the spatial relationship between the legs is more similar to the original source shape as shown in Fig. 4(d).

We test the proposed method on various morphologies. As shown in Fig. 6, we compare the performance of our method to the Linear Blend Skinning using the same skin weights as the source animation, and the results created by an artist. We can see that our method can capture the contextual meaning of the source animation, and the resulting poses are similar to the artist's manual corrections.



(a) Source calibration Shape (b) Deformed Source Pose I (c) Deformed Source Pose II (d) Deformed Source Pose III (e) Reconstruction Error

Figure 3: Reconstruction error of harmonic mapping computed as per pose for 371 frames of punching motion. We map the source calibration pose (a) with 18K vertices onto the other punching motion (b-d). The coded color indicates the scale of approximation error from 0 (blue) to 0.04 (red). The reconstruction error curve shows that most of reconstructed vertices are very close to ground truth points (e).

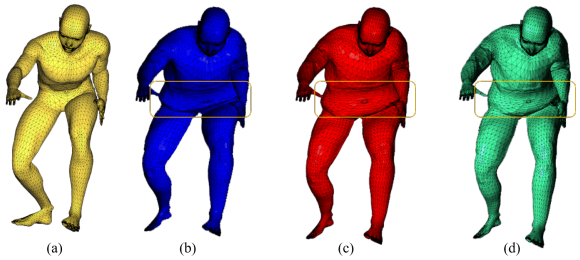


Figure 4: Influence of constraints preserving the volume and using the *Context Graph*: Source shape (a) and Target shape deformed using our method without volume preservation (b), with volume preservation (c), and with both volume and *Context Graph* preservation (d).

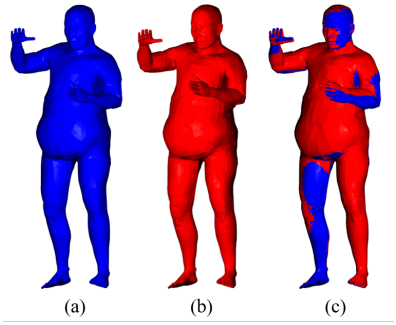


Figure 5: Our method with (a) and without (b) bone length constraints. Overlay of both (c).

Serious artifacts can result if we apply the skin weights of source animation to a new target calibration shape directly. As illustrated in Fig. 6(f), the whole body is twisted and interpenetrated due to the change of morphology and the lack of contextual information. Thus, it is inefficient to produce retargeting animation based on the naive Linear Blend Skinning.

Although our method generates sensible retargeting poses, we can not totally avoid interpenetrations because we use a sparse set of nodes to construct the *Context Graph*. As shown in Fig. 6(d), the right hand may penetrate the left hand due to insufficient nodes of *Context Graph* to capture the spatial relationship. On the other hand, due to the limitation of the bone length and the avoidance of interpenetration, the resulting pose can be slightly different from the deformed source pose. As shown in the fourth example of Fig. 6(d), the position of the two hands relative to the head is a little different from the one in the deformed source shape. Such a kind of deformation is common if the morphology of the target calibration is extremely different from the source calibration shape.

9 CONCLUSIONS

We proposed a new context-aware framework for surface based motion retargeting, which can be applied to both manifold and non-manifold meshes. Our algorithm works in two stages: (1) We learn the affine transformation descriptor from the source animation sequences. (2) We produce a new deformation of the target calibration shape that respects the affine transformation descriptor and spatial relationships among different body parts. The contextual meaning is preserved by a new structure called the *Context Graph*. We solve a linear system to compute the deformation of the target shape such that the resulting motions mimic the source deformation.

Currently, in order to preserve the spatial relationship, our method requires the source and target calibration shapes to be in the same pose. While it makes sense to assume the source and target calibration poses are similar in motion retargeting area, a typical target calibration shape provided by the user could have arbitrary pose. Given an arbitrary target calibration shape, we may have two possible solutions: (1) We deform the given target calibration pose such that it has a similar pose as source using shape deformation methods such as [Ben-Chen et al. 2009a; Lipman et al. 2008; Wang

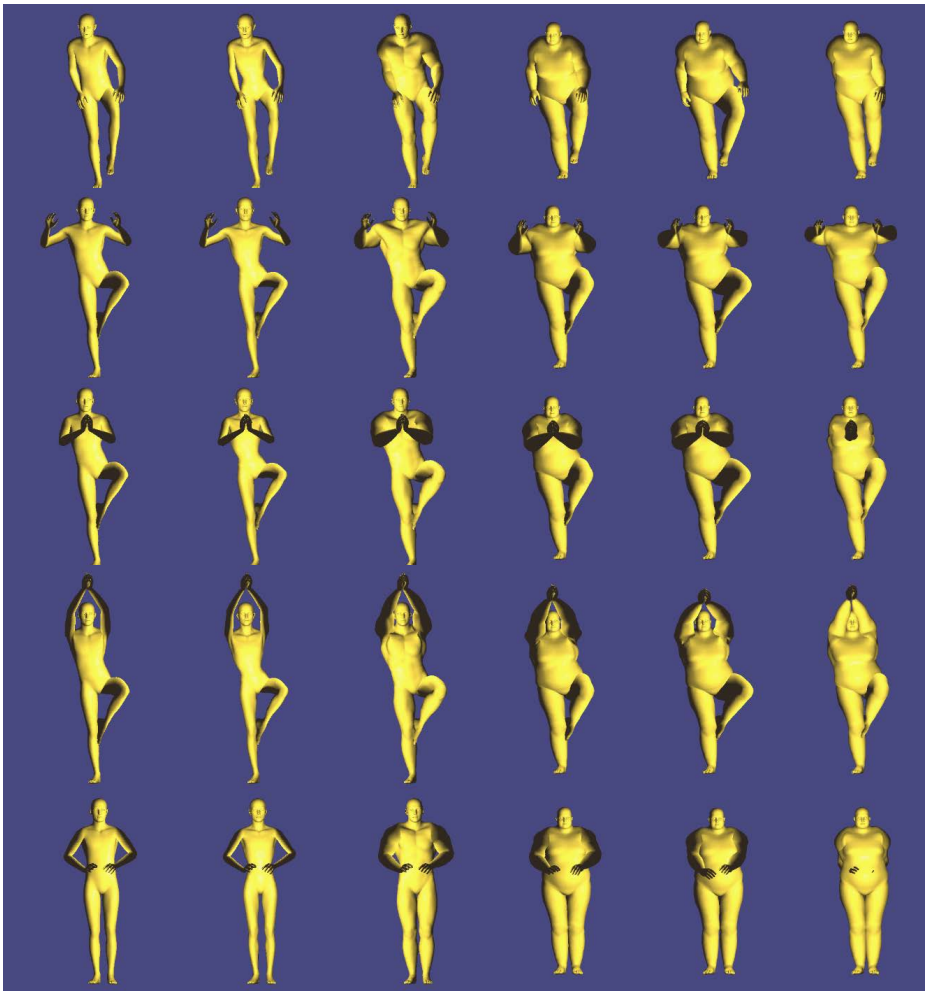


Figure 6: The 'tree pose' retargeting comparison: source poses (a), retargeting results produced by our method (b-d), by an artist (e), and using traditional Linear Blend Skinning (f).

et al. 2015]. (2) We embed the deformation into our retargeting framework and force the spatial relationship constraint to guide the synthesized target pose. While we have shown many retargeting examples both in the paper and supplemental video, we also want to compare our algorithm with the other methods such as the Skinned Multi-Person Linear model [Loper et al. 2015] in the future.

REFERENCES

Rami Ali Al-Asqhar, Taku Komura, and Myung Geol Choi. 2013. Relationship descriptors for interactive motion adaptation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. ACM, 45–53.

Oscar Kin-Chung Au, Chiew-Lan Tai, Hung-Kuo Chu, Daniel Cohen-Or, and Tong-Yee Lee. 2008. Skeleton Extraction by Mesh Contraction. *ACM Trans. Graph.* 27, 3, Article 44 (Aug. 2008), 10 pages.

Ilya Baran, Daniel Vlasic, Eitan Grinspun, and Jovan Popović. 2009. Semantic Deformation Transfer. *ACM Trans. Graph.* 28, 3, Article 36 (July 2009), 6 pages. https://doi.org/10.1145/1531326.1531342

Mirela Ben-Chen, Ofir Weber, and Craig Gotsman. 2009a. Spatial Deformation Transfer. In *Proceedings of the 2009 ACM SIGGRAPH/Eurographics Symposium on Computer Animation (SCA '09)*. ACM, New York, NY, USA, 67–74.

Mirela Ben-Chen, Ofir Weber, and Craig Gotsman. 2009b. Variational Harmonic Maps for Space Deformation. In *ACM SIGGRAPH 2009 Papers (SIGGRAPH '09)*. ACM, New York, NY, USA, Article 34, 11 pages.

Adnane Boukhayma and Edmond Boyer. 2017. Controllable Variation Synthesis for Surface Motion Capture. In *2017 International Conference on 3D Vision (3DV)*. 309–317. https://doi.org/10.1109/3DV.2017.00043

Adnane Boukhayma, Jean-Sébastien Franco, and Edmond Boyer. 2017. Surface motion capture transfer with gaussian process regression. In *CVPR 2017-IEEE Conference on Computer Vision and Pattern Recognition*. 9.

Junjie Cao, Andrea Tagliasacchi, Matt Olson, Hao Zhang, and Zhinxun Su. 2010. Point cloud skeletons via laplacian based contraction. In *Shape Modeling International Conference (SMI), 2010*. IEEE, 187–197.

Dan Casas, Margara Tejera, Jean-Yves Guillemaut, and Adrian Hilton. 2013. Interactive Animation of 4D Performance Capture. *IEEE Transactions on Visualization and Computer Graphics* 19, 5 (May 2013), 762–773. https://doi.org/10.1109/TVCG.2012.314

Graeme Fairweather and Andreas Karageorghis. 1998. The method of fundamental solutions for elliptic boundary value problems. *Advances in Computational Mathematics* 9, 1-2 (1998), 69.

Pablo A Ferrari, Rafael M Grisi, and Pablo Groisman. 2012. Harmonic deformation of Delaunay triangulations. *Stochastic Processes and their Applications* 122, 5 (2012), 2185–2210.

Michael S Floater, Géza Kós, and Martin Reimers. 2005. Mean value coordinates in 3D. *Computer Aided Geometric Design* 22, 7 (2005), 623–631.

Michael Gleicher. 1998. Retargetting Motion to New Characters. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98)*. ACM, New York, NY, USA, 33–42. https://doi.org/10.1145/280814.280820

Edmond S. L. Ho, Taku Komura, and Chiew-Lan Tai. 2010. Spatial Relationship Preserving Character Motion Adaptation. In *ACM SIGGRAPH 2010 Papers (SIGGRAPH '10)*. ACM, New York, NY, USA, Article 33, 8 pages. https://doi.org/10.1145/1833349.1778770

Jin Huang, Xiaohan Shi, Xinguo Liu, Kun Zhou, Li-Yi Wei, Shang-Hua Teng, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2006. Subspace gradient domain mesh deformation. In *ACM Transactions on Graphics (TOG)*, Vol. 25. ACM, 1126–1134.

Alec Jacobson, Ilya Baran, Ladislav Kavan, Jovan Popović, and Olga Sorkine. 2012. Fast automatic skinning transformations. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 77.

Taeil Jin, Meekyoung Kim, and Sung-Hee Lee. 2018. Aura Mesh: Motion Retargeting to Preserve the Spatial Relationships between Skinned Characters. In *Computer Graphics Forum*, Vol. 37. Wiley Online Library, 311–320.

Pushkar Joshi, Mark Meyer, Tony DeRose, Brian Green, and Tom Sanocki. 2007. Harmonic coordinates for character articulation. In *ACM Transactions on Graphics (TOG)*, Vol. 26. ACM, 71.

Yeonjoon Kim, Hangil Park, Seungbae Bang, and Sung-Hee Lee. 2016. Retargeting human-object interaction to virtual avatars. *IEEE transactions on visualization and computer graphics* 22, 11 (2016), 2405–2412.

Richard Kulpa, Franck Multon, and Bruno Arnaldi. 2005. Morphology-independent representation of motions for interactive human-like animation. In *Computer Graphics Forum*, Vol. 24. Wiley Online Library, 343–351.

Hao Li, Robert W. Sumner, and Mark Pauly. 2008. Global Correspondence Optimization for Non-Rigid Registration of Depth Scans. *Computer Graphics Forum (Proc. SGP'08)* 27, 5 (July 2008).

X. Li, X. Guo, H. Wang, Y. He, X. Gu, and H. Qin. 2009. Meshless Harmonic Volumetric Mapping Using Fundamental Solution Methods. *IEEE Transactions on Automation Science and Engineering* 6, 3 (2009), 409–422.

Yaron Lipman, David Levin, and Daniel Cohen-Or. 2008. Green Coordinates. *ACM Trans. Graph.* 27, 3, Article 78 (Aug. 2008), 78:1–78:10 pages. https://doi.org/10.1145/1360612.1360677

Yaron Lipman, Olga Sorkine, Daniel Cohen-Or, David Levin, Christian Rossi, and Hans-Peter Seidel. 2004. Differential coordinates for interactive mesh editing. In *Shape Modeling Applications, 2004. Proceedings.* IEEE, 181–190.

Matthew Loper, Naureen Mahmood, Javier Romero, Gerard Pons-Moll, and Michael J. Black. 2015. SMPL: A Skinned Multi-person Linear Model. *ACM Trans. Graph.* 34, 6, Article 248 (Oct. 2015), 16 pages. https://doi.org/10.1145/2816795.2818013

Matthew M. Loper, Naureen Mahmood, and Michael J. Black. 2014. MoSh: Motion and Shape Capture from Sparse Markers. *ACM Transactions on Graphics, (Proc. SIGGRAPH Asia)* 33, 6 (Nov. 2014), 220:1–220:13.

William E. Lorensen and Harvey E. Cline. 1987. Marching Cubes: A High Resolution 3D Surface Construction Algorithm. In *Proceedings of the 14th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '87)*. ACM, New York, NY, USA, 163–169.

E. Molla, H. G. Debarba, and R. Boulic. 2018. Egocentric Mapping of Body Surface Constraints. *IEEE Transactions on Visualization and Computer Graphics* 24, 7 (2018), 2089–2102.

A Poullikkas, A Karageorghis, and G Georgiou. 1998. Methods of fundamental solutions for harmonic and biharmonic boundary value problems. *Computational Mechanics* 21, 4-5 (1998), 416–423.

Palghat A Ramachandran. 2002. Method of fundamental solutions: singular value decomposition analysis. *International Journal for Numerical Methods in Biomedical Engineering* 18, 11 (2002), 789–801.

Helge Rhodin, James Tompkin, Kwang In Kim, Kiran Varanasi, Hans-Peter Seidel, and Christian Theobalt. 2014. Interactive motion mapping for real-time character control. *Computer Graphics Forum* 33, 2 (2014), 273–282. https://doi.org/10.1111/cgf.12325 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12325

Pedro V Sander, Xianfeng Gu, Steven J Gortler, Hugues Hoppe, and John Snyder. 2000. Silhouette clipping. In *Proceedings of the 27th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 327–334.

Stefan A Sauter and Christoph Schwab. 2010. Boundary element methods. In *Boundary Element Methods*. Springer, 183–287.

Olga Sorkine, Daniel Cohen-Or, Yaron Lipman, Marc Alexa, Christian Rössl, and Hans-Peter Seidel. 2004. Laplacian Surface Editing. In *Proceedings of the EUROGRAPHICS/ACM SIGGRAPH Symposium on Geometry Processing*. ACM Press, 179–188.

Robert W Sumner and Jovan Popović. 2004. Deformation transfer for triangle meshes. In *ACM Transactions on Graphics (TOG)*, Vol. 23. ACM, 399–405.

Steve Tonneau, Rami Ali Al-Ashqar, Julien Pettré, Taku Komura, and Nicolas Mansard. 2016. Character contact re-positioning under large environment deformation. In *Computer Graphics Forum*, Vol. 35. Wiley Online Library, 127–138.

Yu Wang, Alec Jacobson, Jernej Barbič, and Ladislav Kavan. 2015. Linear subspace design for real-time shape deformation. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 57.

Rhaleb Zayer, Christian Rössl, Zachi Karni, and Hans-Peter Seidel. 2005. Harmonic guidance for surface deformation. In *Computer Graphics Forum*, Vol. 24. Wiley Online Library, 601–609.

Kun Zhou, Jin Huang, John Snyder, Xinguo Liu, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2005. Large mesh deformation using the volumetric graph laplacian. In *ACM transactions on graphics (TOG)*, Vol. 24. ACM, 496–503.

Kun Zhou, Weiwei Xu, Yiying Tong, and Mathieu Desbrun. 2010. Deformation Transfer to Multi-Component Objects. In *Computer Graphics Forum*, Vol. 29. Wiley Online Library, 319–325.