Efficient and Flexible Deformation Representation for Data-Driven Surface Modeling

LIN GAO
Institute of Computing Technology, Chinese Academy of Sciences
YU-KUN LAI
School of Computer Science & Informatics, Cardiff University
DUN LIANG
TNList, Tsinghua University
and
SHU-YU CHEN and SHIHONG XIA
Institute of Computing Technology, Chinese Academy of Sciences

Effectively characterizing the behavior of deformable objects has wide applicability but remains challenging. We present a new rotation-invariant deformation representation and a novel reconstruction algorithm to accurately reconstruct the positions and local rotations simultaneously. Meshes can be very efficiently reconstructed from our representation by matrix predecomposition, while, at the same time, hard or soft constraints can be flexibly specified with only positions of handles needed. Our approach is thus particularly suitable for constrained deformations guided by examples, providing significant benefits over state-of-the-art methods. Based on this, we further propose novel data-driven approaches to mesh deformation and non-rigid registration of deformable objects. Both problems are formulated consistently as finding an optimized model in the shape space that satisfies boundary constraints, either specified by the user, or according to the scan. By effectively exploiting the knowledge in the shape space, our method produces realistic deformation results in real-time and produces high quality registrations from a template model to a single noisy scan captured using a low-quality depth camera, outperforming state-of-the-art methods.

Categories and Subject Descriptors: I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism—*Animation*; I.3.5 [**Computer Graphics**]: Computational Geometry and Object Modeling—*Surface representation*

General Terms: Algorithms

Additional Key Words and Phrases: Mesh representation, rotation difference, data-driven, deformation, registration

ACM Reference Format:

Lin Gao, Yu-Kun Lai, Dun Liang, Shu-Yu Chen, and Shihong Xia. 2016. Efficient and flexible deformation representation for data-driven surface modeling. ACM Trans. Graph. 35, 5, Article 158 (July 2016), 17 pages.
DOI: http://dx.doi.org/10.1145/2908736

1. INTRODUCTION

Effectively characterizing the behavior of deformable objects such as human bodies, animals, and so on, has wide applicability in computer graphics, ranging from mesh editing to animation. Previous research often considers this in the settings of mesh deformation. For articulated models such as human bodies, skeletons have been widely used to allow control by artists. Mesh-based deformation is becoming popular due to its generality. Methods either work directly on the mesh coordinates or utilize coordinates such as differential coordinates that are insensitive to certain rigid transformations, hence better represent fundamental non-rigid deformations. Rotation sensitive coordinates such as Laplacian coordinates or gradient domain representation [Sorkine et al. 2004; Yu et al. 2004; Au et al. 2006] do not propagate rotations directly and thus require sophisticated heuristics and optimizations to improve results. To address this, existing works [Lipman et al. 2005; Kircher and Garland 2008; Baran et al. 2009; Hasler et al. 2009] consider rotation-invariant representations. However, two-stage reconstructions are needed by these methods where the local frames are constructed in the first stage followed by recovery of vertex positions in the second stage. As a result, such methods require not only positions but also compatible *rotations* to be specified at handles for rotations to be properly reconstructed.

In this work, we present a new rotation-invariant mesh representation that also encodes local deformation differences, similarly

This work was supported by the National Natural Science Foundation of China (Grants No. 61502453 and No. 61173055), Beijing Municipal Natural Science Foundation (Grant No. 4132073), and CCF-Tencent Open Research Fund (Grant No. CCF-TencentIAGR20150116).
Authors' addresses: Emails: gaolin@ict.ac.cn, Yukun.Lai@cs.cardiff.ac.uk, randonlang@gmail.com, chenshuyu@ict.ac.cn, xsh@ict.ac.cn. L. Gao, Shihong Xia and S.-Y. Chen are with the Beijing Key Laboratory of Mobile Computing and Pervasive Device, ICT, CAS, China. Y.-K Lai is with the School of Computer Science & Informatics, Cardiff University, Cardiff, United Kingdom. D. Liang is with TNList, Tsinghua University, China.

This work is licensed under a Creative Commons Attribution International 4.0 License.

2016 Copyright is held by the owner/author(s).
0730-0301/2016/07-ART158
DOI: http://dx.doi.org/10.1145/2908736

to Baran et al. [2009]. Unlike Baran et al. [2009], we propose a novel surface reconstruction method based on our representation to solve for the vertex positions and local rotations at the same time. The shape can be efficiently and accurately recovered, by solving a non-linear yet efficient as-rigid-as-possible optimization [Sorkine and Alexa 2007]. Our representation and reconstruction approaches allow flexible *positional* constraints to be specified without the need to specify rotations at handles. Moreover, the matrix involved in our process is determined by the mesh connectivity, which can be pre-decomposed. As a result, given our representation, reconstructing the mesh is very efficient, with real-time performance achieved using multi-resolution optimization. These fundamental advantages allow novel applications for surface modeling, and we will in particular investigate mesh deformation and non-rigid registration.

Geometry-based surface modeling methods are not aware of the physical properties of the objects, thus the naturalness of manipulation is limited. Efforts have been made to use physically based approaches. However, such methods rely on careful modeling and analysis of objects, which are only possible to some extent for specific types of objects. Physically based modeling is also often too expensive for interactive applications. With the proliferation of models, data-driven methods have received a lot of attention. Natural manipulation results are learned from examples for both deformation (e.g., Sumner et al. [2005] and Fröhlich and Botsch [2011]) and morphing (e.g., Gao et al. [2013]). We propose a data-driven approach to mesh deformation. As we will show later, our method benefits from the unique characteristics of the representation and the reconstruction method and produces substantially better results over state-of-the-art methods.

Shape registration aims to find appropriate transformations to put multiple shapes (e.g., 3D scans) into alignment. Based purely on the shapes to be registered, registration techniques typically require a good initialization to converge to the desired solutions and may often get stuck at a poor local minimum for incomplete, noisy data. This is more challenging for non-rigid registration due to the substantially larger solution space. Previous work considers data-driven approaches by using a deformable model (e.g. Schneider and Eisert [2009]) and finding the optimal fitting of the deformable model to the target scan. However, such techniques only work for cases where deformation is relatively subtle and can be easily blended, for example, human faces or heads; these cases are also easier to establish correspondence using closest points. We propose a data-driven approach to more *general* non-rigid registration using the deformation space as a prior.

In this article, based on our rotation-invariant deformation representation and shape reconstruction method, we propose novel data-driven deformation and non-rigid registration algorithms. We treat these two problems in a uniform optimization framework as finding a suitable model in the shape space following constraints, either specified as handle positions by the user for deformation or as the target scan for registration. Example models in our representation are used to provide a useful prior to constrain deformation and registration. Some results are shown in Figure 1. By using a collection of models as examples, our method produces realistic deformation even with substantial movement of handles (Figures 1(a)–(c)). For non-rigid registration, we first obtain a complete template of the deforming object using KinectFusion (Figure 1(d)) and transfer the geometry to a model collection. Given a new incomplete, single-view, noisy, and distorted depth scan obtained using a Kinect v2 camera (Figure 1(e)), our method successfully registers the template to the scan with the help of example models (Figure 1(f)). We will demonstrate that such deformation and non-rigid registration are challenging for existing methods.

The main contributions of this article are summarized as follows:

—We propose a novel shape reconstruction algorithm based on a new rotation-invariant representation that solves for vertex positions and local rotations simultaneously. Plausible deformations in this representation often form a near linear subspace, which allows standard dimensionality reduction and linear combination to be applied. Given our representation, the mesh can be *efficiently* reconstructed, with *flexible* constraints.
—Based on this, we propose a novel *data-driven* mesh deformation method, which produces substantially improved results over state of the art.
—We further propose a novel *data-driven* non-rigid registration technique that produces high quality registrations from a template model to a single noisy scan captured using a low-quality depth camera, by exploiting knowledge in a model database.

We first review relevant work in Section 2. The representation and its properties are discussed in Section 3. We introduce two novel data-driven surface modeling techniques, namely mesh deformation and non-rigid registration in Section 4. Experimental results and discussions are provided in Section 5, and, finally, Section 6 concludes the article.

2. RELATED WORK

Surface representation and surface-based deformation. This is an active research direction in recent decades. A large volume of research work exists in the field. A complete survey is beyond the scope of this article. Please refer to Botsch and Sorkine [2008] and Gain and Bechmann [2008] for excellent surveys. We focus on the techniques most relevant to our work.

Many surface-based deformation techniques benefit from some suitable representations: Results that better preserve local details are obtained with coordinates that are invariant to certain rigid transformations. Local differential coordinates are used to encode local details and recover them after deformation. These methods include Laplacian coordinates [Sorkine et al. 2004], Poisson-based gradient field reconstruction [Yu et al. 2004], and the iterative dual Laplacian approach for improved results [Au et al. 2006]. While differential (Laplacian) coordinates are translation invariant, they are still sensitive to rotations. As a result, sophisticated heuristics and optimizations are needed to cope with rotational deformations. The volumetric graph Laplacian constructed in the adjacent space of the surface is proposed to better preserve the volume under large deformations [Zhou et al. 2005]. Huang et al. [2006] propose a non-linear gradient domain approach that incorporates various constraints such as volume, skeleton, and projection. A subspace technique is used for efficiently optimizing the non-linear energy first on the coarse mesh and then interpolated over the original mesh. Rotation-invariant coefficients are employed in this work. Sumner et al. [2005] use deformation gradients to represent shape deformations. As deformation gradients are related to global orientation, this approach cannot effectively blend multiple deformations when they have different global orientations and may lead to artifacts.

Rotation-invariant coordinates handle rotations effectively, which is a highly desirable property. Lipman et al. [2005] propose a linear rotation-invariant shape representation that defines connection maps between adjacent frames. The connection maps are not explicitly stored, so need to be recovered first. Although the method only requires solving two linear systems, to obtain good results an iterative approach is often needed. Kircher and Garland [2008] propose a second-order representation to represent and process free-form motions. Their representation defines connection maps between

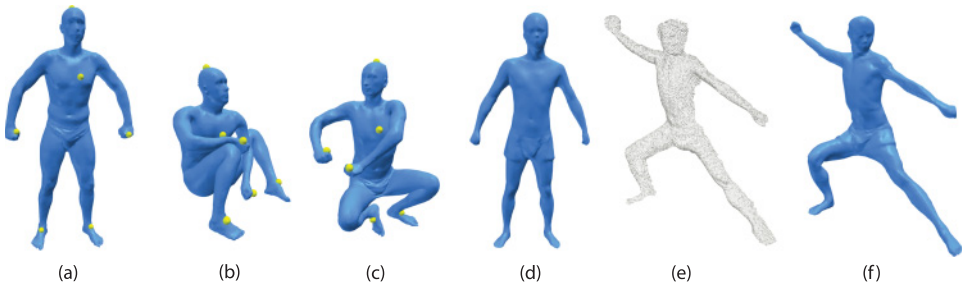


Fig. 1. Data-driven deformation and non-rigid registration using our method. (a) Input model with handles highlighted, ((b) and (c)) results using our data-driven deformation, (d) a template model of the target person obtained using KinectFusion, (e) a scanned point cloud of a single view captured using a Kinect v2, and (f) our non-rigid registration result.

adjacent frames explicitly, although the frames are not orthogonal, which may introduce global shear artifacts. To avoid this, Baran et al. [2009] define connection maps between adjacent orthonormal frames. A similar rotation-invariant representation is used in Hasler et al. [2009] to encode human shapes and poses for regression of semantic properties (e.g., height and weight). However, these methods have similar limitations that require rotations to be reconstructed before vertex positions. To get good deformation results, *compatible* rotations at handles need to be specified, in addition to positions, which increases user effort substantially.

Our representation and reconstruction methods are rotation invariant, but do not have such restrictions. Vertex positions and rotations are optimized together; there is no need to specify rotations at handles. Furthermore, our approach can cope with large-scale deformations in a consistent manner and is therefore more suitable for data-driven algorithms that may need to combine multiple shapes with potentially large deformations.

Geometry-based mesh deformation is also formulated as an optimization problem. Terzopoulos et al. [1987] formulate a shell energy to measure the distortions between the input and the deformed models. Sorkine and Alexa [2007] estimate the rigid transformations of local cells and collect the transformations to deform the whole model. The principle of as-rigid-as-possible (ARAP) deformation has also been applied to shape manipulation [Igarashi et al. 2005]. Such works based on the ARAP principle have a similar framework. They all estimate local rigid transformations of geometric elements (e.g., triangle faces) and then build a global energy formulation based on the L_2 norm. Zohar et al. [2015] propose a smooth rotation (SR) enhanced ARAP-style method for shape deformation and morphing. Freifield and Black [2012] define a low-dimensional non-linear manifold with the Lie group of deformations, mainly for human body shape representation. Their experiments were restricted to human bodies with similar poses. Chao et al. [2010] define a smooth deformation map minimizing the difference between the differential map and the rotation group. Its discretization leads to an ARAP energy. For surface cases with one-ring edge sets, it leads to a new continuous energy involving a parameter radius r. However, the optimal choice of r is still an open problem. The work also provides an interpolation method for linear blending of shapes. However, the energy cannot be applied for *extrapolation* because the minimization of the energy would become negative infinity. The method thus is not suitable for data-driven deformation as extrapolation is essential.

Data-driven shape deformation. Purely geometry-based approaches are limited in identifying suitable deformation given user constraints. This is because the physical properties of the objects cannot be fully captured by the geometry alone. Instead of using expensive physically based modeling, data-driven approaches exploit existing examples to improve the naturalness of deformation. Skeleton-based methods (e.g., Feng et al. [2008]) learn the relation between control points and different surface regions for improved mesh skinning. Shi et al. [2008] propose a data-driven skinning method for articulated models with volumetric effects learned from example sequences. Such methods are restricted to articulated deformation, mainly for human bodies. Mesh-based inverse kinematics derived from example models are used to produce stylized surface deformation [Sumner et al. 2005; Fröhlich and Botsch 2011]. Such methods, however, are generally restricted to a small number of examples due to the high computational costs and also produce suboptimal results with large deformations. As pointed out in previous work [Botsch and Sorkine 2008; Winkler et al. 2010], the gradient-based MeshIK method [Sumner et al. 2005] will lead to problems that cannot blend rotations larger than 180°. To address this problem, Fröhlich and Botsch [2011] use edge length, dihedral angles, and volumes that are rotation invariant to represent the mesh. However, it is not suitable for extrapolation (i.e., deformation beyond those in the examples), since this may need the edge length to be negative, which is not possible. Extrapolation is essential to address challenging cases when a data-driven approach is used. Our method works effectively in both interpolation and extrapolation.

A relevant active research area is example-based simulation. Such methods typically require volumetric tetrahedral meshes as input and use a small number of examples. Material properties also need to be specified. The pioneering work by Martin et al. [2011] proposes an example-based simulation method for objects of complex elastic material. The deformation manifold is defined by shape interpolation with a Finite Element Method (FEM) energy. The method is physically realistic, although expensive to optimize. To speed up the computation, Koyama et al. [2012] propose to define the manifold by simple linear interpolation, which runs in real time, although at the cost of losing some physical accuracy, especially when the behavior cannot be fully captured by the examples. Schuma et al. [2012] extend Martin et al. [2011] to improve efficiency and provide flexible artistic control by combining incompatible linearly interpolated shapes with a compatible configuration. An efficient physical solver is proposed [Bouaziz et al. 2014] for real-time simulation with local/global optimization. Zhang et al. [2015] propose a Green strain tensor-based potential energy for example-based elastic material with real-time efficiency. Compared with these works, our approach does not require tetrahedral meshes as input and is able to cope with a large number of example models as well as situations where physical properties are complex or unknown.

Example-based deformation has also been used for dynamic sprite animation [Jones et al. 2015], where a drawing and example poses are specified by artists, and the dynamics are achieved by navigating in the pose manifold following specified forces. Tycowicz et al. [2015] consider the problem of non-linear interpolation of shapes and propose a very efficient real-time solution by constructing the optimization problem in a low-dimensional subspace. Our approach focuses on data-driven deformation and non-rigid registration, which has different input and/or output, compared with these works.

Non-Rigid Registration. Registration is a technique of finding appropriate transforms to put two or more shapes into alignment. Please refer to Tam et al. [2013] for a recent survey. Rigid registration assumes a global rigid-body transform and is largely based on Iterative Closest Point (ICP) [Besl and McKay 1992] or its variants [Pottmann et al. 2006]. However, a good initialization is required as such methods only converge to local minima. Non-rigid registration is more flexible and better copes with deforming objects. Such techniques allow objects to be deformed as part of the alignment. Li et al. [2008, 2009] non-rigidly register dynamic depth scans using deforming templates. Such work depends on high-quality depth scans as input. Bouaziz and Pauly [2013] survey non-rigid registration work and provide a code implementation for low-quality RGBD data (e.g., captured using a Kinect). Zollhöfer et al. [2014] develop a combined software and hardware solution to real-time non-rigid registration of a template to live RGBD data. Such methods do not use additional data apart from the template (i.e., a constructed static surface) for registration of general shapes.

Some works use data priors to help reconstruct shapes. For facial reconstruction, works such as Weise et al. [2011] blend shape models to fit the scans. Schneider and Eisert [2009] use a deformable model to fit human heads. However, linear blending models used by such methods cannot handle shapes with substantial rotations. Anguelov et al. [2005] parameterize the space of human body and pose deformation and use marker-based motion-captured data to reconstruct human bodies. The pose deformation of this work is defined on the articulated skeleton, which is not suited for general non-rigid registration or reconstruction. With this parameterized model, Loper et al. [2014] capture the shape and motion of human bodies from sparse markers. Alternative work uses a data-driven approach to recover poses from a single depth camera [Wei et al. 2012]. Such techniques are only designed for human bodies and the purpose is tracking rather than registration. Non-rigid registration is the fundamental technique for both static [Li et al. 2013] and dynamic [Zhang et al. 2014] human body reconstruction. Our work focuses on non-rigid registration of *general* shapes, although the advances would also be beneficial to human body reconstruction.

In this article, we propose a new rotation-invariant mesh difference representation and a novel surface reconstruction method to effectively encode model deformations. Plausible deformations in this representation often form a near linear subspace, which allows standard dimensionality reduction and linear combination to be applied. Using this, we further propose novel data-driven approaches to mesh deformation and non-rigid registration of a template model to a single (often incomplete) scan of a deformed object captured using a low-quality depth camera. Various examples demonstrate that our method produces substantially improved results over state-of-the-art methods.

3. DEFORMATION REPRESENTATION

Fundamental to this work is a new shape representation to encode the rotation-invariant local mesh differences between deforming surfaces, and a novel reconstruction algorithm to efficiently obtain deformed shapes from the representation. In this section, we first introduce this representation, which involves two parts, namely rigid rotation differences and local scaling/shear. We then analyze the characteristics of this representation, including efficient mesh reconstruction by matrix pre-decomposition, incorporating constraints, and near linear subspaces formed by typical deformations. These provide the basis for novel algorithms for data-driven deformation and non-rigid registration.

3.1 Representation Formulation

We assume that we have m models ($m \geq 2$) consistently triangulated, each with n vertices that are in one-to-one correspondence. Without loss of generality we further assume that the first model is the reference model and other models are deformed models. Let us denote $\mathbf{p}_i$ as the position of the ith vertex (denoted as v_i) on the reference model and $\mathbf{p}'_i$ as the position of v_i on a deformed model. The deformation gradient $\mathbf{T}_i$ in the one-ring neighborhood of v_i from the reference model to the deformed model can be calculated by minimizing the following energy:

$$E(\mathbf{T}_i) = \sum_{j \in N_i} c_{ij} \|\mathbf{e}'_{ij} - \mathbf{T}_i \mathbf{e}_{ij}\|^2, \tag{1}$$

where N_i is the one-ring neighbors of vertex v_i, $\mathbf{e}'_{ij} = \mathbf{p}'_i - \mathbf{p}'_j$, and $\mathbf{e}_{ij} = \mathbf{p}_i - \mathbf{p}_j$. c_{ij} is the cotangent weight $c_{ij} = \cot\alpha_{ij} + \cot\beta_{ij}$, which helps prevent mesh discretization bias [Sorkine and Alexa 2007; Levi and Gotsman 2015], where α_{ij} and β_{ij} are angles opposite to the edge connecting v_i and v_j. As we will show later in Section 5.1, cotangent weights lead to reduced reconstruction errors, in particular for meshes with poor triangulation. The affine transformation matrix can be decomposed into a rotation part and a scaling/shear part using polar decomposition $\mathbf{T}_i = \mathbf{R}_i\mathbf{S}_i$.

We define the rotation difference dR_{ij} from v_i to adjacent vertex v_j as follows:

$$dR_{ij} = \mathbf{R}_i{}^T \mathbf{R}_j. \tag{2}$$

The energy $E(\mathbf{T}_i)$ can be rewritten using rotation differences as:

$$E(\mathbf{T}_i) = \sum_{j \in N_i} c_{ij} \sum_{t \in N_i} \tilde{c}_i \|\mathbf{e}'_{ij} - \mathbf{R}_t dR_{ti} \mathbf{S}_i \mathbf{e}_{ij}\|^2, \tag{3}$$

where $\tilde{c}_i = 1/|N_i|$, $|N_i|$ is the number of neighboring vertices of v_i. The scaling/shear transformations are rotation invariant by nature and can be interpolated directly. The rotation transformations in the rotation matrix space $SO(3)$ are usually interpolated by first mapping them to 3×3 skew-symmetric space $so(3)$ using the matrix logarithm, linearly interpolating them in this space, and, finally, mapping them back to $SO(3)$ using the matrix exponential [Murray et al. 1994; Alexa 2002]. Since $\|e^{\mathbf{X}+\mathbf{Y}} - e^{\mathbf{X}}\| \leq \|\mathbf{Y}\| \cdot e^{\|\mathbf{X}\|} \cdot e^{\|\mathbf{Y}\|} (\forall \mathbf{X}, \mathbf{Y})$, the exponential map is Lipschitz continuous [Horn and Johnson 1986]. This means that if two matrices are sufficiently close in *so(3)*, they are also close in *SO(3)*. To be rotation invariant and allow effective *linear* combination, we combine the *logarithm* of the rotation difference dR_{ij} of each edge (v_i, v_j) and scaling/shear matrix $\mathbf{S}_i$ of each vertex v_i to get the feature representation $\mathbf{f}$ of the deformed model as follows:

$$\mathbf{f} = \{\log dR_{ij}; \mathbf{S}_i\} (\forall i,\ j \in N_i). \tag{4}$$

As discussed, including log in this representation allows robust linear interpolation and the rotation difference cancels out global rotation and thus makes it rotation invariant. As we will show later in Section 5.1, the representation is more effective than alternative

representations, in particular for linear extrapolation. In this article, we call this rotation-invariant mesh difference (abbreviated as RIMD) representation.

3.2 Surface Reconstruction from RIMD Representation

Given the initial pose $\mathbf{p} = \{\mathbf{p}_j\}$, where $\mathbf{p}_j$ is the position of vertex v_j, and a RIMD representation $\mathbf{f} = \{\log dR_{ij}; \mathbf{S}_i\}$ of the deformation, the reconstructed geometry can be obtained by finding new positions $\mathbf{p}' = \{\mathbf{p}'_j\}$ that minimize the following energy function:

$$E(\mathbf{p}') = \sum_{i\in V} E(\mathbf{T}_i) = \sum_{i\in V}\sum_{j\in N_i} \tilde{c}_j \sum_{k\in N_j} c_{jk} \|\mathbf{e}'_{jk} - \mathbf{R}_i\, dR_{ij}\, \mathbf{S}_j\, \mathbf{e}_{jk}\|^2, \quad (5)$$

where $\mathbf{e}_{jk} = \mathbf{p}_j - \mathbf{p}_k$, $\mathbf{e}'_{jk} = \mathbf{p}'_j - \mathbf{p}'_k$, $dR_{ij} = \exp(\log dR_{ij})$ as well as $\mathbf{S}_j$ is part of the RIMD representation, $\mathbf{R}_i$ is the (unknown) rotation matrix at vertex v_i, V is the set of the vertices, and $\tilde{c}_j = 1/|N_j|$. This is equivalent to Equation (1) with the order of summation changed.

Given a reference shape and the RIMD feature, to obtain the new positions $\mathbf{p}'$ as well as the per-vertex rotation matrix $\mathbf{R}_i$, we alternate the following two steps:

Step 1 (global step): Given $\mathbf{R}_i$ for each vertex, find the optimal positions $\mathbf{p}'$. For each $\mathbf{p}'_j$, a linear equation is obtained:

$$\frac{\partial E(\mathbf{p}')}{\partial \mathbf{p}'_j} = \frac{\partial}{\partial \mathbf{p}'_j}\sum_{i\in N_j} \tilde{c}_j \sum_{k\in N_j} c_{jk} \|\mathbf{e}'_{jk} - \mathbf{R}_i\, dR_{ij}\, \mathbf{S}_j\, \mathbf{e}_{jk}\|^2 + \frac{\partial}{\partial \mathbf{p}'_j}\sum_{k\in N_j} c_{kj} \sum_{s\in N_k} \tilde{c}_k \|\mathbf{e}'_{kj} - \mathbf{R}_s\, dR_{sk}\, \mathbf{S}_k\, \mathbf{e}_{kj}\|^2 = 0.$$

Using $c_{kj} = c_{jk}$, this can be simplified to

$$\sum_{k\in N_j} c_{jk}\mathbf{e}'_{jk} = \frac{1}{2}\sum_{k\in N_j} c_{jk}\left(\tilde{c}_k \sum_{s\in N_k} \mathbf{R}_s\, dR_{sk}\, \mathbf{S}_k + \tilde{c}_j \sum_{i\in N_j} \mathbf{R}_i\, dR_{ij}\, \mathbf{S}_j\right)\mathbf{e}_{jk}.$$

Those terms involving two-ring neighbors can be efficiently calculated by accessing one-ring neighbors of each vertex twice. In the first pass, one-ring neighbors are accessed to calculate $\sum_{i\in N_j} \mathbf{R}_i\, dR_{ij}\, \mathbf{S}_j$ that are saved for use in the second pass. The resulting linear system $\mathbf{A}\mathbf{p}' = \mathbf{b}$ has the matrix $\mathbf{A}$ fixed, irrespective of varying $\mathbf{R}_i$, so by using Cholesky factorization, the linear system can be *efficiently* solved in each iteration.

Step 2 (local step): Given $\mathbf{p}'$, find the optimal $\mathbf{R}_i$. Let us denote $\mathbf{e}_{jk} = \mathbf{p}_j - \mathbf{p}_k$ and $\mathbf{e}'_{jk} = \mathbf{p}'_j - \mathbf{p}'_k$. $\mathbf{R}_i$ is separate, so it can be individually optimized. Expand Equation (5) and ignore terms irrelevant to $\mathbf{R}_i$, so the optimal $\mathbf{R}_i$ can be obtained as:

$$\arg\max_{\mathbf{R}_i} \sum_{j\in N_i} \tilde{c}_j \sum_{k\in N_j} c_{jk}\mathbf{e}'^T_{jk}\mathbf{R}_i\, dR_{ij}\, \mathbf{S}_j\, \mathbf{e}_{jk} = Tr\left(\sum_{j\in N_i} \tilde{c}_j \sum_{k\in N_j} c_{jk}\mathbf{R}_i\, dR_{ij}\, \mathbf{S}_j\, \mathbf{e}_{jk}\mathbf{e}'^T_{jk}\right) = Tr\left(\mathbf{R}_i \sum_{j\in N_i} \tilde{c}_j dR_{ij}\, \mathbf{S}_j \left(\sum_{k\in N_j} c_{jk}\mathbf{e}_{jk}\mathbf{e}'^T_{jk}\right)\right).$$

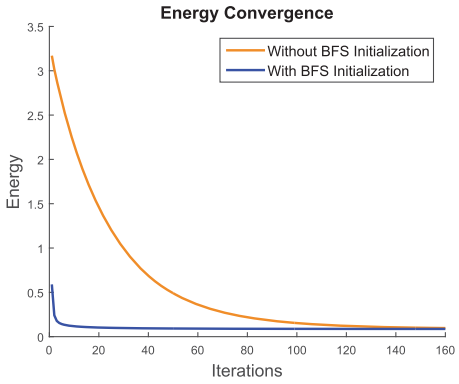


Fig. 2. Reconstruction energy over iterations for the bar example in Figure 14(top), with or without breadth-first search- (BFS) based initialization.

Let us denote $Q_i = \sum_{j\in N_i} \tilde{c}_j dR_{ij}\, \mathbf{S}_j \left(\sum_{k\in N_j} c_{jk}\mathbf{e}_{jk}\mathbf{e}'^T_{jk}\right)$. This step can also be done by one-ring neighbor transversal twice, with $\sum_{k\in N_j} c_{jk}\mathbf{e}_{jk}\mathbf{e}'^T_{jk}$ calculated in the first pass. Using Singular Value Decomposition (SVD), $Q_i = \mathbf{U}_i\Sigma_i\mathbf{V}_i^T$. Then $\mathbf{R}_i$ can be explicitly obtained as $\mathbf{V}_i{\mathbf{U}_i}^T$ (choosing appropriate signs to make $\det \mathbf{R}_i > 0$).

The iterative optimization terminates on convergence (i.e., when the energy change $|\Delta E| < \varepsilon$, $\varepsilon = 10^{-3}$ in our experiments). This stopping criterion for iterations works well for all the examples in our experiments. While more iterations can be applied, no visible improvements can be seen in all these examples.

Initial values are needed for the optimization. A trivial initialization would set all the $\mathbf{R}_i$ to the identity matrix (so with no rotation). Since we have rotation difference dR_{ij} between every pair of adjacent vertices v_i and v_j, we can very efficiently obtain a good initialization for $\mathbf{R}_i$. We choose an arbitrary vertex and set the initial rotation matrix to the identity matrix. We then propagate the rotation matrix from this vertex to neighboring vertices using a bread-first search (BFS) strategy. Assume vertex v_i is visited and its adjacent vertex v_j is about to be visited, then $\mathbf{R}_j$ is initialized as $\mathbf{R}_i dR_{ij}$. If the RIMD representation is derived directly from a deformed shape, then BFS initialization recovers the shape directly, so no iteration is needed and the reconstructed geometry recovers the geometry *exactly* (apart from numerical errors). In case the RIMD representation does not correspond to a deformed shape (e.g., by blending multiple RIMD representations), the BFS-based initialization helps to converge more quickly. For the example shapes with large difference in Figure 13 and reconstruction of their blending in Figure 14(top), as demonstrated in Figure 2, with the BFS initialization, the energy converges within four iterations. Algorithm 1 gives the pseudocode of major steps for surface reconstruction from the RIMD representation.

Compared with traditional manipulation methods that rely on differential coordinates over one-ring neighbors [Sorkine and Alexa 2007], our representation exploits second-order representations that rely on two-ring neighbors. It is *rotation invariant* and can be effectively combined by a *linear* combination. This not only provides a useful tool for shape space analysis but also allows example deformation information to be incorporated for data-driven deformation and non-rigid registration, as will be demonstrated later.

Reconstruction with constraints. As we will show later, reconstruction with constraints is essential for data-driven deformation and registration.

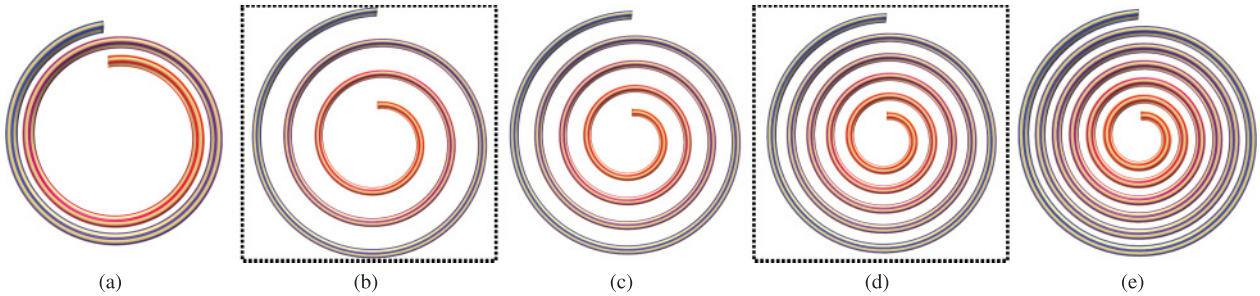


Fig. 3. Shape blending with interpolation and extrapolation. Panels (b) and (d) are the source ($t = 0$) and target ($t = 1$) models. Panels (a), (c), and (e) are interpolated/extrapolated models with $t = -0.5, 0.5, 1.5$, respectively.

For *hard* constraints, certain vertices (known as handles) $\mathbf{p}'_i$ are fixed to specified positions $\mathbf{t}_i$: $\mathbf{p}'_i = \mathbf{t}_i$, $i \in H$, where H is the handle set containing all the vertices with positional constraints. The elements in the ith row of matrix $\mathbf{A}$ are set to zeros except for the diagonal element which is 1, $\mathbf{b}_i = \mathbf{t}_i$. With these changes, the remaining optimization is the same as described.

Given a *soft* constraint that the position at handle h should be close to $\mathbf{v}_h$, we introduce a new energy term $\lambda\|\mathbf{n}_h^T(\mathbf{p}'_h - \mathbf{v}_h)\|^2$ to the energy formulation where λ is a weight for the soft constraint (the bigger it is the stronger the constraint will be), and $\mathbf{n}_h$ is the normal direction similar to point-to-plane distances such that only deviations off the surface are penalized. The soft energy term does not have a direct influence on the local rigid rotation optimization step (step 2 above). For the step of optimizing positions given rotations (step 1 above), $\frac{\partial}{\partial \mathbf{p}'_h}\lambda\|\mathbf{n}_h^T(\mathbf{p}'_h - \mathbf{v}_h)\|^2 = 2\lambda(\mathbf{n}_h\mathbf{n}_h^T\mathbf{p}'_h - \mathbf{n}_h\mathbf{n}_h^T\mathbf{v}_h)$, so $\lambda\mathbf{n}_h\mathbf{n}_h^T$ and $\lambda\mathbf{n}_h\mathbf{n}_h^T\mathbf{v}_h$ will be added to the corresponding entries of $\mathbf{A}$ and $\mathbf{b}$. The optimization can then proceed as before.

ALGORITHM 1: Surface Reconstruction from RIMD Representation.

Require: Initial pose vertex positions $\mathbf{p}$, RIMD feature $\mathbf{f}$
Ensure: Deformed mesh vertex positions $\mathbf{p}'$
  Construction of matrix $\mathbf{A}$ and Cholesky pre-decomposition
  Initialization of $\mathbf{R}_i$ using Breadth First Search
  repeat
    Global Step Optimization for $\mathbf{p}'$
    Local Step Optimization for $\mathbf{R}_i$
  until $|\Delta E| < \varepsilon$

3.3 Shape Blending

By using polar decomposition and the matrix logarithm/exponential, our RIMD representation allows intuitive shape blending by using linear weights:

$$\mathbf{f}(\mathbf{w}) = \sum_k w_k \cdot \mathbf{f}_k, \tag{6}$$

where $\mathbf{f}_k = \{\log dR_{k,ij}; \mathbf{S}_{k,i}\}$ is the RIMD feature of the kth model and w_k is an arbitrary weight (not necessarily in the range of $[0, 1]$). This is equivalent to blending rotation difference and scaling/shear matrices as follows:

$$dR_{ij}(\mathbf{w}) = \exp\left(\sum_k w_k \cdot \log(dR_{k,ij})\right), \tag{7}$$

Fig. 4. A failure case of directly blending large rotations: self intersection is caused by interpolation of substantially different poses. ((a) and (f)) Two shapes to be blended, ((b)–(e)) blended shapes.

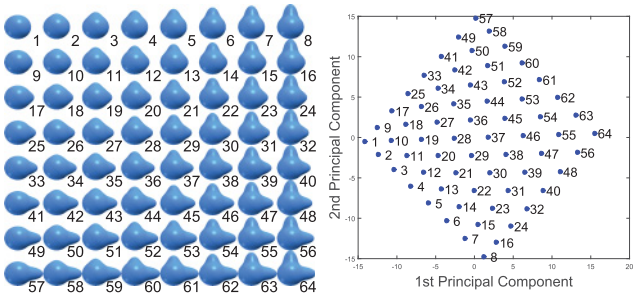
$$\mathbf{S}_j(\mathbf{w}) = \sum_k w_k \cdot \mathbf{S}_{k,j}. \tag{8}$$

An example is shown in Figure 3. Take the source (Figure 3(b)) and target (Figure 3(d)) models that are spirals with three and five cycles, respectively. Because of the extensive rotations, it is a challenging case for interpolation. Our approach can effectively *interpolate* (Figure 3(c)) as well as *extrapolate* (Figures 3(a) and (e)) them by changing the parameter t, and realistic results are obtained. In this example, the weights used for the source and target models are t and $1 - t$, respectively. Extrapolation is essential for effective data-driven surface modeling, as knowledge hidden in the given examples can be better utilized.

In theory, the linear blending of the logarithm used here may introduce errors when the rotations being blended are not coaxial [Bloom et al. 2004]. In practice, however, even for shapes with very large rotations, the rotation differences between adjacent faces are usually still small, and the error is often negligible, as shown in examples throughout the article. When very large rotations are to be blended directly, self-intersection in space may be produced (see an example in Figure 4).

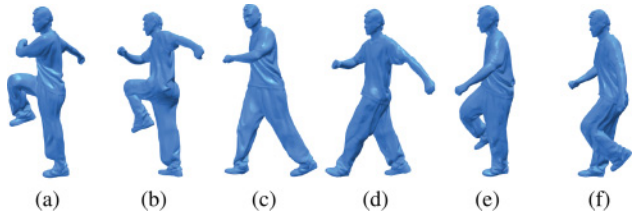
3.4 Deformation Space Analysis

Our feature vector gives an effective rotation-invariant representation of the deformations. To better understand the behavior of this representation, given a set of deforming models, we use standard Principal Component Analysis (PCA) of the feature vectors to reduce the dimensions to 2. An example is shown in Figure 5. Given a collection of spheres (Figure 5(a)) from Rustamov et al. [2013] with uniform deformation distribution, Figure 5(b) shows the distribution of coefficients on the two most significant principal axes. Results obtained are very similar to map-based exploration



(a) (b)

Fig. 5. PCA analysis of a collection of spheres with two modes of variation, using RIMD features. (a) A collection of spheres; (b) the value distribution over the two most significant principal coordinates.



(a) (b) (c) (d) (e) (f)

Fig. 6. First three axis models for the "march2" dataset from Vlasic et al. [2008]. ((a) and (b)) Models on first axis, ((c) and (d)) models on second axis, and ((e) and (f)) models on third axis.

[Rustamov et al. 2013], although with a much simpler approach. This example shows that, for typical deformations, the resulting feature vectors tend to form a low-dimensional near linear subspace.

Compact representation of deforming models. For datasets with a large number of models, the essential variations may form a much lower dimensional space. Applying PCA to such datasets allows us to use the mean shape as well as the first few principal components to represent all the major modes of variations compactly. A similar idea has been widely used but mainly for images or shapes with limited amount of deformation (e.g., human faces). Thanks to our RIMD representation, we show that this tool works effectively for general deformed shape datasets. An example is given in Figure 6, using the "march2" dataset from Vlasic et al. [2008] containing 250 captured human bodies of a marching sequence. The first four principal components capture over 70% of the energy (variances) and the first 18 principal components capture over 90% of the energy. The first three modes are shown, with models corresponding to maximum/minimum values. Instead of using the whole dataset for data-driven deformation/registration, we instead represent the unknown RIMD feature $\mathbf{f}(\mathbf{w})$ as a linear combination of basis vectors:

$$\mathbf{f}(\mathbf{w}) = \bar{\mathbf{f}}_0 + \sum_s w_s \cdot \bar{\mathbf{f}}_s, \tag{9}$$

where $\bar{\mathbf{f}}_0$ is the mean feature vector, $\bar{\mathbf{f}}_s$ corresponds to the sth principal component, and $\mathbf{w}$ is the weight vector that determines the reconstructed feature vector. Using this equation instead of Equation (6), due to the reduced dimension, the problem can be solved much more efficiently as the running time scales almost linearly with the number of unknown weights. As we will show later (see Figure 19), using dimensionality reduction, the data-driven deformation results with 4 and 18 basis vectors are visually very similar yet tens of times faster than directly using the original example model dataset. This shows that fewer basis vectors are often sufficient for mesh manipulation than for traditional reconstruction.

In summary, our RIMD representation supports flexible positional constraints, allows meaningful linear blending (including interpolation/extrapolation) and dimensionality reduction, and can be very efficiently reconstructed using matrix pre-decomposition. These unique characteristics make it particularly suitable for data-driven surface modeling as we will demonstrate in the next section.

4. DATA-DRIVEN DEFORMABLE SURFACE MODELING

In this section, we further exploit our representation for novel data-driven surface modeling techniques, namely data-driven deformation and non-rigid registration.

ALGORITHM 2: Data Driven Deformation.

Require: Initial pose vertex positions $\mathbf{p}$, RIMD features of example models $\mathbf{f}_k$, Handle vertex set H and their target positions $\mathbf{v}_h$
Ensure: Deformed mesh vertex positions $\mathbf{p}'$
  Combination weight $\mathbf{w}$ is initialized based on previous deformation or initial pose.
  repeat
    Optimize $\mathbf{w}$ using gradient descent and line search
    Optimize positions $\mathbf{p}'$ given $\mathbf{w}$ using Algorithm 1
  until line search step size $r < \tilde{\varepsilon}$

4.1 Data-Driven Deformation

Given a set of example models M_k, and their corresponding RIMD features $\mathbf{f}_k = \{\log dR_{k,ij}, \mathbf{S}_{k,j}\}$, we assume a RIMD feature constrained by the examples is defined as a linear combination of these feature vectors $\mathbf{f}_k$, using weights $\mathbf{w} = \{w_k\}$ satisfying $\sum w_k = 1$.

Our data-driven deformation is defined as finding the optimal weights $\mathbf{w}$ such that the derived RIMD feature leads to a reconstructed mesh with handles placed at the specified locations while at the same time the overall deformation energy is minimized. Treating handles as *hard* constraints guarantee that the deformed surfaces follow user constraints precisely. Let us denote H as the index set of handle vertices, $h \in H$ is a handle index, $\mathbf{p}'_h$ is the location of the handle vertex, and $\mathbf{v}_h$ is the user-specified location for the handle h. Once the weights are determined, the RIMD feature $\mathbf{f}(\mathbf{w})$ can be obtained by simple linear combination, and the deformed vertex positions can be obtained by minimizing Equation (5).

Data-driven deformation is now formulated as minimizing

$$E(\mathbf{w}, \mathbf{p}') = \sum_{i \in V} \sum_{j \in N_i} \tilde{c}_j \sum_{k \in N_j} c_{jk} \| \mathbf{e}'_{jk} - \mathbf{R}_i \, dR_{ij}(\mathbf{w}) \, \mathbf{S}_j(\mathbf{w}) \, \mathbf{e}_{jk} \|^2, \tag{10}$$

where $\mathbf{e}_{jk} = \mathbf{p}_j - \mathbf{p}_j$, $\mathbf{e}'_{jk} = \mathbf{p}'_j - \mathbf{p}'_j$, and for each $h \in H$, $\mathbf{p}'_h$ is set to the user specified $\mathbf{v}_h$ and not included in the optimization. Note that, as described before, $\mathbf{R}_i$ also needs to be solved in the energy above. We omit it from $E(\cdot)$ for simplicity since it is not the focus here. This is a non-linear problem. We use gradient descent with line search to find a suitable step size. The gradients are numerically calculated. Assuming $\boldsymbol{\delta}$ is the negative gradient direction, we find the step size r through repeated halving, such that $E(\mathbf{w} + r\boldsymbol{\delta}) < E(\mathbf{w})$. $\mathbf{w}$ is then updated to $\mathbf{w} + r\boldsymbol{\delta}$. The above optimization is iterated until convergence when r is less than $\tilde{\varepsilon} = 10^{-6}$ in our experiment. Algorithm 2 shows pseudocode of the data-driven deformation algorithm.

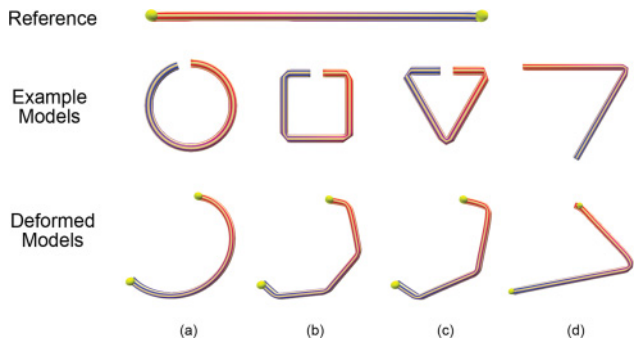


Fig. 7. Deforming a cylinder model (top row) using various styles specified by example models (middle row). Bottom row: The corresponding deformed models with the handles highlighted.

Our method is very efficient. Given the fixed set of handle vertices, the matrix pre-decomposition only needs to be performed once and can be reused for both updated weights (iterations used to find optimal weights) and updated handle positions (when the user drags handles). With a good initialization from the initial shape/previous deformation, the solution is efficient and real-time performance is obtained for a medium-sized collection of example models. For a large number of example models we first perform an analysis of the shape space to reduce redundancy (see Section 3.4). A simple example is shown in Figure 7. Given a reference model and one of the example models, our method effectively produces deformed models following the specified styles.

As a data-driven approach, our method effectively utilizes example shapes to help guide the deformation. In practice, however, the deformation specified by the handles may not be fully present in the examples. As we will show later, even in such cases our method still produces reasonable deformation with necessary distortions uniformly distributed, which is desirable.

Multi-Resolution Optimization. To efficiently process dense mesh models, inspired by Fröhlich and Botsch [2011], we apply the data-driven deformation in simplified mesh models first, which is then used to guide the deformation of the original mesh models. To simplify the shapes, we adapt the quadric error metric-based mesh decimation framework [Garland and Heckbert 1997] such that the cost of contracting an edge (v_i, v_j) is related to both the geometry and deformation properties. The cost now includes three terms. The first term is from the original definition $\bar{\mathbf{v}}^T(\mathbf{Q}_i + \mathbf{Q}_j)\bar{\mathbf{v}}$, where $\bar{\mathbf{v}}$ is the new vertex location, and $\mathbf{Q}_i$ and $\mathbf{Q}_j$ are quadric error matrices at v_i and v_j, respectively. The second and the third terms are new, which are the Frobenius norm of standard deviations of $\log(dR_{ij})$ and $\mathbf{S}_i/\mathbf{S}_j$ over all the shapes in the dataset. These new terms penalize contraction of edges with more variations across shapes. We normalize each term by the standard deviations δ_Q, δ_R, and δ_S over all the edges. $C_{ij} = (\bar{\mathbf{v}}^T(\mathbf{Q}_i + \mathbf{Q}_j)\bar{\mathbf{v}})/\delta_Q + \|std(\log(dR_{ij}))\|_F/\delta_R + (\|std(\mathbf{S}_i)\|_F + \|std(\mathbf{S}_j)\|_F)/\delta_S$, where $std(\cdot)$ is the matrix corresponding to the standard deviation of each entry, and $\|\cdot\|_F$ is the Frobenius norm of the matrix.

We simplify the original mesh M to obtain M' and keep the map from M to M'. Then data-driven deformation is applied to the simplified mesh. After the optimization we get the representation $\tilde{\mathbf{f}}$ of the deformed mesh $\tilde{M}'$. This is equivalent to getting the rigid rotation matrix $\tilde{\mathbf{R}}_i$ and scaling matrix $\tilde{\mathbf{S}}_i$ at each vertex on the simplified mesh. According to the mapping, we obtain the rigid rotation matrix $\mathbf{R}_i$ and scaling matrix $\mathbf{S}_i$ of the original mesh M, which can be converted to RIMD representation $\mathbf{f}$. Based on $\mathbf{f}$, we reconstruct the deformed shape by optimizing Equation (5) (typically within a few iterations). Fröhlich and Botsch [2011] use deformation transfer [Sumner and Popović 2004] to obtain deformation of the original mesh from the simplified mesh. When the mesh is significantly simplified, direct deformation gradient transfer of adjacent faces produces jagged results. Our multi-resolution optimization, on the other hand, uses the transferred representation as the guidance and the rigid rotation of each vertex on the original mesh is re-optimized and thus is free from such artifacts. The Hausdorff distance between the multi-resolution optimization and direct optimization on the original mesh is negligible—for the SCAPE (Shape Completion and Animation of People) [Anguelov et al. 2005] and elephant cases they are both less than 1% of the radius of the bounding sphere. As demonstrated in the supplementary video, the data-driven deformation for the SCAPE dataset with 10 PCA axis models is very efficient: by simplifying the original mesh with 25K triangles to 3K triangles, data-driven deformation takes less than 50ms. Real-time performance with over 20fps is achieved.

4.2 Data-Driven Non-Rigid Registration

Using our representation, we further propose a novel data-driven approach to non-rigid registration. For a dynamic object (such as a human body), we first capture a complete template model of the object which can be obtained using, for example, KinectFusion [Izadi et al. 2011]. Then given a new scan of the deformed object from a single view using a low-quality depth camera (Kinect v2 is used in our experiments), we register the template to the noisy, often incomplete, scan. This is challenging due to the missing information and potentially substantial deformation between the template and the scan. To address this, we use a data-driven approach to help improve the non-rigid registration.

We make a reasonable assumption that a collection of deformed objects of the same *class* is available. So, for instance, for human body registration, a collection of deformed human bodies (of an arbitrary person) is sufficient. This allows us to use existing public datasets for data-driven non-rigid registration. We assume models in the collection have the same connectivity (which many existing datasets satisfy). We establish correspondence between the template model and the models in the collection by specifying a few key correspondences. This is similar to Sumner and Popović [2004], where the user selects vertices in correspondence between the template model and the models in the collection. Deformation transfer [Sumner and Popović 2004] is then used to deform the models in the collection to produce models with geometry similar to the template. The resulting models are used as examples.

Similar to the data-driven deformation, we assume that a linear combination of feature vectors in the example space best represents the desired non-rigid deformation. $\mathbf{w}$ is the weight vector, and $\sum w_k = 1$. We formulate non-rigid registration as an optimization problem minimizing

$$E(\mathbf{w}, \mathbf{p}') = \sum_{i\in V}\sum_{j\in N_i}\tilde{c}_j \sum_{k\in N_j} c_{jk}\|\mathbf{e}'_{jk} - \mathbf{R}_i\, dR_{ij}(\mathbf{w})\,\mathbf{S}_j(\mathbf{w})\,\mathbf{e}_{jk}\|^2 + \lambda_{point}\sum_{h\in H}\|\mathbf{v}_h - \mathbf{p}'_h\|^2 + \lambda_{plane}\sum_{h\in H}\|\mathbf{n}_h(\mathbf{v}_h - \mathbf{p}'_h)\|^2, \tag{11}$$

where $\mathbf{p}$ contains the template vertex positions and $\mathbf{p}'$ is the solution that corresponds to the deformed template vertex positions that register well with the scan. H is the set of vertices whose current positions are sufficiently close to the scan. $\mathbf{v}_h$ is the foot point position when projecting vertex h to the scan, and $\mathbf{n}_h$ is the normal direction

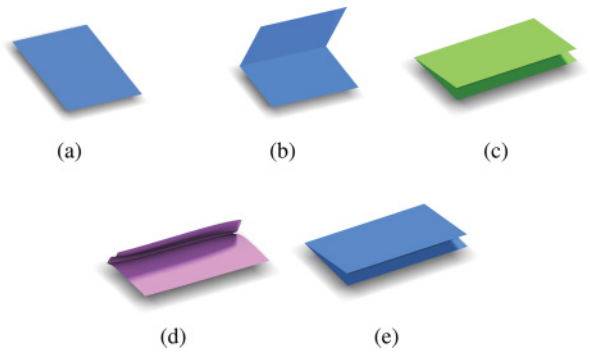


Fig. 8. Non-rigid registration of a synthetic card paper model with two examples. ((a) and (b)) Example models (flat paper with 0° dihedral angle and folded paper with 90° dihedral angle), (c) the target paper folded with 170° dihedral angle, (d) result of standard non-rigid registration, and (e) result of our data-driven method.

at the foot point. The first term constraints the deformation to be as rigid as possible, from some examples in the deformation space; this gives a knowledge-based prior. The second and the third terms ensure the deformation fits with the scan, where the second term is point to point distance, and the third term is the point-to-plane distance. We choose λ_{point} and λ_{plane} as 0.2 and 0.8 in our experiments. Note that in principle we should also optimize the rigid transform between $\mathbf{p}'_h$ and $\mathbf{v}_h$, in addition to non-rigid deformation. We can prove that our optimization handles this automatically without explicitly introducing the rigid transform; see the proof in the appendix that shows that the energies with or without explicit optimization of rigid transforms have the same global minimum, although, due to the non-linear nature, they may lead to different local minima.

The energy formulation involves the weights $\mathbf{w}$ as well as the closest point correspondence, in addition to the deformed positions $\mathbf{p}'$. We initialize the optimization by choosing one example model that best fits the scan. Assuming this is the kth model, we initialize $\mathbf{w}$ such that $w_k = 1$ and $w_j = 0, \forall j \neq k$. We then solve the problem by using an algorithm similar to non-rigid ICP: We alternate between two steps, namely finding the correspondence H and finding the improved positions $\mathbf{p}'$ and weights $\mathbf{w}$ minimizing the energy. The former step is similar to ICP as described above. Given fixed correspondence, we solve the latter step by using a gradient descent algorithm to optimize $\mathbf{w}$ with line search for a suitable step size, similar to the data-driven deformation. For a given $\mathbf{w}$, we use constrained reconstruction with soft constraints to recover vertex positions $\mathbf{p}'$ and local rotations (see Section 3.2). Algorithm 3 shows pseudocode of the data-driven non-rigid registration algorithm. For a large number of example models, we similarly perform an analysis of the shape space first to reduce redundancy (see Section 3.4). A simple synthetic example is shown in Figure 8. Given only two examples with card papers forming dihedral angles of 0° (Figure 8(a)) and 90° (Figure 8(b)), the aim is to find a non-rigid registration from Figure 8(b) to Figure 8(c) with a dihedral angle of 170°. The existing non-rigid registration method [Bouaziz and Pauly 2013] does not converge to the correct position (Figure 8(d)), whereas the additional examples help to show the potential deformation which leads to the correct result (Figure 8(e)).

5. RESULTS

Our experiments were carried out on a computer with an Intel i7-4790K CPU and 16GB RAM. Depending on the size of the mesh and the number of example models (or the number of reserved basis vectors if PCA is used), our data-driven deformation takes from a few milliseconds up to about 50ms, which gives real-time feedback. The detailed running times of our non-rigid registration algorithms are shown in Table I. Our data-driven non-rigid registration takes under 15s for these examples. We used various datasets from the existing research, including those of Anguelov et al. [2005], Zhang et al. [2004], Sumner and Popović [2004], and Vlasic et al. [2008]. When compared with existing non-rigid registration methods, we use the code from Bouaziz and Pauly [2013] and adapt it to register 3D meshes to scans. Throughout the article we have shown some examples to demonstrate the ideas of our method. In this section, we will show more results and compare them with state-of-the-art methods.

Table I. Statistics of the Data-Driven Non-Rigid Registration Running Times

Dataset	# Vertices	# Models	# iteration. (s)	total (s)
card	2500	2	0.02	0.11
SCAPE	12500	71	1.37	14.3
hand	10825	56	1.24	12.9

ALGORITHM 3: Data-Driven Non-Rigid Registration.

Require: Initial pose vertex positions $\mathbf{p}$, RIMD features of example models $\mathbf{f}_k$, scanned point cloud data
Ensure: Deformed mesh vertex positions $\mathbf{p}'$
  Combination weight $\mathbf{w}$ is initialized based on the nearest shape in the example set.
  repeat
    Find nearest points in the point cloud for each vertex
    Optimize $\mathbf{w}$ using gradient descent and line search
    Optimize positions $\mathbf{p}'$ given $\mathbf{w}$ (with position and normal soft constraints) using Algorithm 1
  until $|\Delta E| < \varepsilon$

5.1 Deformation Representation

Comparison of rotation difference representations. Our technique benefits greatly from linear analysis of features for both shape space construction and PCA dimensionality reduction. We compare the log-exp-based rotation difference representation we used with alternative rotation difference representations, namely rotation matrix and quaternion. As the slerp interpolation of quaternions is non-linear, it cannot be applied to our problem. We thus use an alternative approach of linear interpolation of rotation matrices and quaternions, followed by normalization. For the blended rotation matrices, the Procrustes projection is used to map them back to optimal orthonormal matrices, and for quaternions, the blended quaternions are simply normalized to be of unit length.

We compare the results by blending two shapes shown in Figures 8(a) and (b). By using a synthetic example, the ground truth can be easily obtained for fair comparison. As shown in Figure 9, the log-exp representation we used produces very similar results to the ground truth in both *interpolation* (first row) and *extrapolation* (second and third rows), whereas both the linear interpolations of the rotation difference matrices and quaternions lead to incorrect extrapolation results. Extrapolation appears frequently and is essential to effectively exploit the hidden knowledge in data-driven shape deformation and PCA analysis. The log-exp representation allows linear blending and obtains robust results in both interpolation and extrapolation and thus is more suitable for our technique.

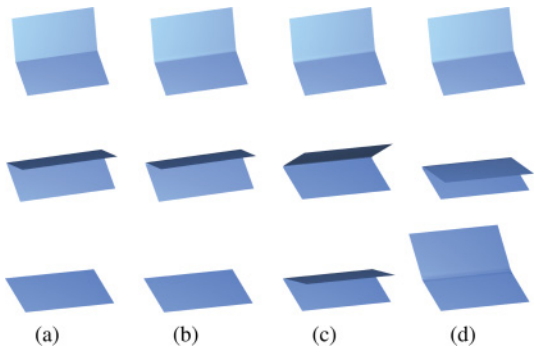


Fig. 9. Comparison of rotation difference representations for shape blending using the synthetic example in Figure 8. The first row is the interpolation results when $t = 0.5$, and the second and third rows are the extrapolation results with $t = 1.5$ and $t = 2$, respectively. (a) The ground truth with 45 degrees, 135 degrees and 180 degrees dihedral angles, (b) the log-exp results, (c) the Procrustes projection of linear blending of rotation difference matrices, and (d) the linear interpolation of quaternions.

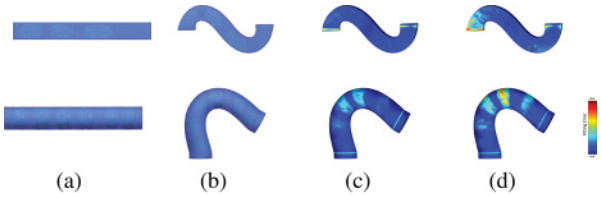


Fig. 10. Shape reconstruction for the deformed *bar* (top row) and *cylinder* (bottom row) shapes. (a) Source mesh, (b) deformed mesh, (c) reconstructed mesh using cotangent weights (with error color coding), and (d) reconstructed mesh using uniform weights (with error color coding).

Table II. Mean and Maximum Reconstruction Errors (cf. Figure 10) with Different Edge Weighting (Cotangent vs. Uniform)

Case	mean errors		maximum errors	
	cotangent	uniform	cotangent	uniform
Bar	1.41×10^{-4}	1.95×10^{-4}	1.97×10^{-3}	2.95×10^{-3}
Cylinder	1.05×10^{-4}	1.35×10^{-4}	5.97×10^{-4}	1.15×10^{-3}

Comparison of different edge weighting c_{ij}. We compare the cotangent weights c_{ij} used in Equation (1) with alternative uniform weights for shape reconstruction (see Figure 10). We use shapes from Levi and Gotsman [2015] and simplify certain regions to demonstrate the behavior of poor triangulation. The reconstruction errors measured as Euclidean distances from the ground truth are illustrated using color coding. The mean and maximum errors are summarized in Table II. While errors are fairly small for both cases with little visual difference, the reconstructed errors (especially maximum errors) are substantially smaller with cotangent weights.

Comparison with ARAP. Similar to Sorkine and Alexa [2007], our optimization approach involves both global and local steps. For the global step, both our method and ARAP [Sorkine and Alexa 2007] solve a linear system involving a Laplacian matrix with cotangent weights. For the local step, both methods use SVD decomposition. These dominant steps take identical time. The only difference in running times is that ARAP needs to access one-ring neighbors of each vertex, whereas our method needs to access two-ring neighbors when the matrices are built. By using precomputation, our optimization can be implemented by accessing one-ring neighbors of each vertex twice. Although our method needs to access more neighboring vertices, the runtime difference is very little. For each iteration of the bar example in Figure 13 with 12K triangles, our method takes 44ms, whereas ARAP takes 40ms.

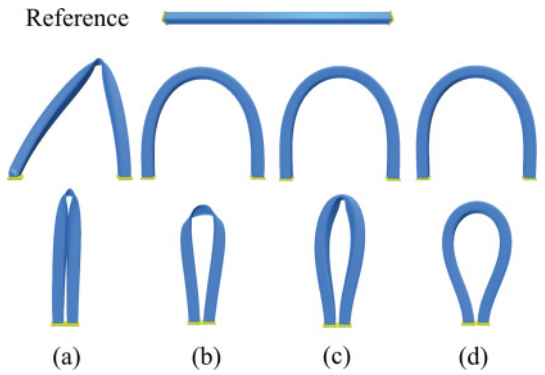


Fig. 11. Deforming of a thin bar with large scale rotations. (a) Fröhlich and Botsch [2011], (b) Sumner et al. [2007], (c) Sorkine and Alexa [2007], and (d) our reconstruction results with constraints.

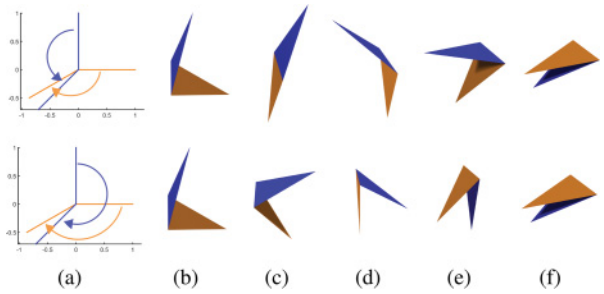


Fig. 12. Shape blending results of MeshIK and our method using a synthetic example. The first row is the result of MeshIK [Sumner et al. 2005], The second row is the result of our method. ((b) and (f)) Two simple surfaces indicating the start and end shapes for blending, ((c)–(e)) intermediate surfaces, and (a) illustration of the rotations of both triangles.

In Figure 11 we show non-data-driven deformation results (without examples). The handle is rotated by 180°, and our method distributes the deformation distortions much more uniformly than state-of-the-art methods [Fröhlich and Botsch 2011; Sumner et al. 2007; Sorkine and Alexa 2007]. Note that although the focus of our method is data-driven deformation, high-quality non-data-driven deformation is also essential and is particularly useful when the user moves handles beyond the scope of example deformations.

Comparison with MeshIK. Data-driven approaches rely on examples, and the capability of blending shapes is essential. Figure 12 shows a comparison with MeshIK [Sumner et al. 2005] on a simple synthetic dataset containing only two triangles. MeshIK finds paths with minimal rotation of triangles *individually*, which causes self intersections from Figures 12(e) to (f). This shows that MeshIK cannot blend large-scale rotations well. Our method produces artifact-free blending.

Shape blending and constrained reconstruction. Existing research most relevant to our representation involves data-driven approaches [Sumner et al. 2005; Fröhlich and Botsch 2011] and rotation-invariant coordinates [Lipman et al. 2005; Kircher and Garland 2008; Baran et al. 2009]. The latter are not designed for data-driven deformation so do not exploit examples. To make a

Fig. 13. The original bar model and with one end rotated by 360°.

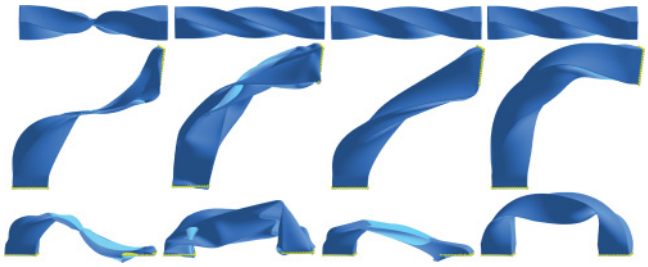


Fig. 14. Blended shapes and deformation. First column: MeshIK [Sumner et al. 2005], second column: Fröhlich and Botsch [2011], third column: Baran et al. [2009], fourth column: our results. First row: 1:1 blending of two shapes in Figure 13, second and third rows: blended shapes deformed with constraints.

fair comparison, we apply these methods to a blended shape (1:1 blending from examples in Figure 13).

Rotation-invariant coordinates are not designed for data-driven deformation. For example, Baran et al. [2009] use a technique for semantic deformation transfer. To use such representations for deformation, the user needs to specify both the *positions* and *rotations* at handles. The handle rotations are used as constraints for the first linear system and the handle positions for the second linear system. As demonstrated in Lipman et al. [2005], both the rotation and position handles need to be assigned compatibly to avoid unnatural deformation. This increases the workload and difficulties for the user. Our approach, as well as existing data-driven deformation techniques [Sumner et al. 2005; Fröhlich and Botsch 2011], only requires handle positions to be specified.

As shown in Figure 14, the original bar and the bar rotated by 360° are blended (see top row). MeshIK does not handle large rotations well. Fröhlich and Botsch [2011], Baran et al. [2009], and our approach all produce reasonable results. When significant deformations are applied, existing methods exhibit significant artifacts (with Baran et al. [2009] producing overall better results) and our method handles all these cases effectively.

Reconstruction running time comparison. We compare the running times with state-of-the-art deformation methods using the example in Figure 14 with 12K triangles, with the following implementation setup: solving linear systems using MATLAB, SVD decomposition using Eigen, and parallelization using OpenMP.

Existing rotation-invariant coordinates [Lipman et al. 2005; Kircher and Garland 2008; Baran et al. 2009] have a similar reconstruction framework that involves solving two linear systems. The first linear system solves local frames and the second global positions. Given a new shape representation, the coefficients of the first linear system will differ, so it is necessary to decompose the matrix *every* time. When [Baran et al. 2009] is used without segmenting the shapes into patches (i.e., treating each face as a patch), it needs 520ms to solve the first linear system for frames and a further 20ms for SVD decomposition to calculate the local rigid rotations. For comparison, without using multi-resolution optimization, after a one-off matrix pre-decomposition, each iteration of our algorithm involves two steps: 14ms for the global step and 30ms for the local step. A total of 44ms is needed for each iteration, and four iterations are needed to converge in this case. Our method is thus about 3 times faster than that of Baran et al. [2009]. Fröhlich and Botsch [2011] use the Gauss-Newton method to solve the formulated optimization. In each iteration, the Jacobian matrix is calculated and used for solving the linear equations. However, the Jacobian matrix is changed in each iteration, leading to a total running time of 24, 600ms. While being slow, this method applies to data-driven deformation, unlike existing linear rotation-invariant coordinates.

To reduce the size of the linear system for frames, Baran et al. [2009] partition the shapes consistently into patches. While being effective, this technique needs a large dataset with various poses such that face clusters well capture rigid components of the deforming shape, and the approximation error is minimized. This is not the case for this example where only two examples with uniform distortions are provided, and thus the patch-based technique is not appropriate. As we have shown, our method can also use multi-resolution optimization for speedup, which achieves real-time performance for data-driven deformation. Note also that it is not obvious how existing rotation-invariant coordinates [Lipman et al. 2005; Kircher and Garland 2008; Baran et al. 2009] can be generalized to data-driven deformation.

5.2 Data-Driven Mesh Deformation

We compare our deformation method with various state-of-the-art deformation methods, with or without using examples.

Figure 15 is an example of deforming an elephant model with a small number (12) of examples. Since the handles move substantially it is challenging for non-data-driven methods, which produce results with visible distortion artifacts. State-of-the-art non-data-driven methods [Sorkine and Alexa 2007; Sumner et al. 2007] (Figures 15(b) and (c)) produce overall deformations that look quite rigid, as the deformations are purely driven by the movement of handles, and, hence, do not look realistic. Data-driven methods (Figures 15(d)–(f)) produce much more vivid results, since the dataset provides the elephant galloping sequences which give the information of the leg and head movements. The data-driven methods deform the front and rear legs much more realistically, giving a running effect. The non-data-driven methods simply rotate the front legs, which is too rigid. Existing data-driven methods [Sumner et al. 2005; Fröhlich and Botsch 2011], however, are not able to cope with such large-scale deformation well and thus have visible artifacts. Our method produces realistic deformation without artifacts. For example, the trunk is deformed with no examples in the dataset having similar trunk deformation. In such cases, our approach turns to as-rigid-as-possible reconstruction with constraints defined on two-ring neighborhoods, which produces smoother and more realistic results than existing data-driven methods.

Figure 16 shows an example of using the SCAPE dataset [Anguelov et al. 2005]. Due to the large movement of handles (hence significant change of pose), existing methods, including data-driven methods [Sumner et al. 2005; Fröhlich and Botsch 2011], produce results with significant distortions and/or self-intersections. This is because existing non-data-driven methods do not distribute distortions well due to lack of information, and existing data-driven methods do not handle large deformations well. A natural result is obtained using our data-driven method.

Figure 17 shows a challenging case where the example models in the dataset differ substantially. The head is rotated more than 180°, and the tip of the tail is rotated nearly 360° between Figure 17(a) and Figures 17(b) and (c). These three models give the information of the rotation of the tail and the neck as well as the joint movement of the legs. As shown in Figures 12 and 14, MeshIK [Sumner et al.

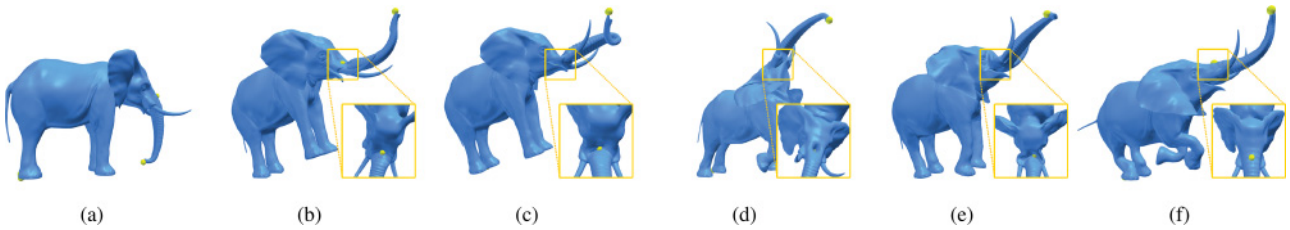


Fig. 15. Deformation of an elephant using the dataset from Sumner and Popović [2004]. (a) Input model with handles highlighted, (b) ARAP [Sorkine and Alexa 2007], (c) [Sumner et al. 2007], (d) MeshIK [Sumner et al. 2005], (e) [Fröhlich and Botsch 2011], and (f) our data-driven deformation method.

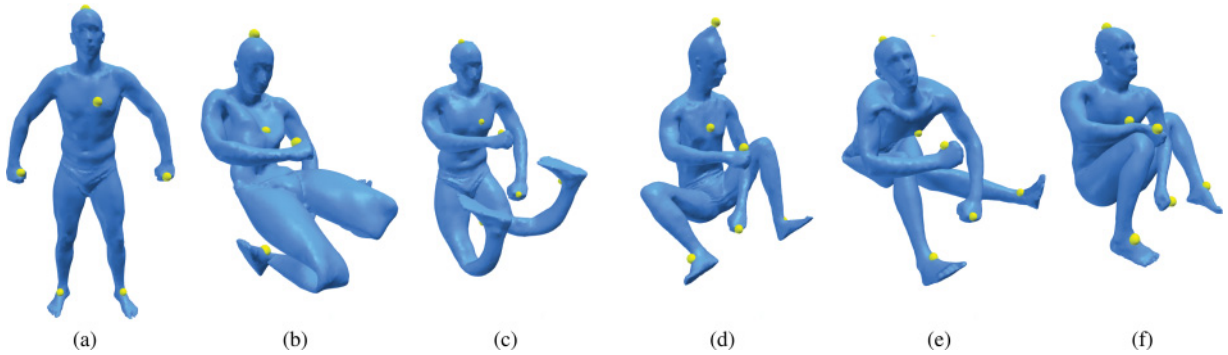


Fig. 16. Comparison of deformation results using the SCAPE dataset [Anguelov et al. 2005]. (a) Input model, (b) ARAP [Sorkine and Alexa 2007], (c) [Sumner et al. 2007], (d) MeshIK [Sumner et al. 2005], (e) [Fröhlich and Botsch 2011], and (f) our data-driven deformation method.

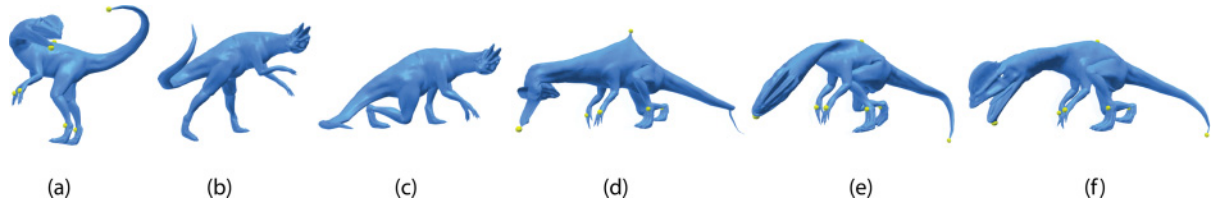


Fig. 17. Data-driven deformation results with two example models. (a) Input model with handles highlighted, ((b) and (c)) two additional example models, (d) result of MeshIK [Sumner et al. 2005], (e) result of Fröhlich and Botsch [2011], and (f) result of our data-driven deformation.

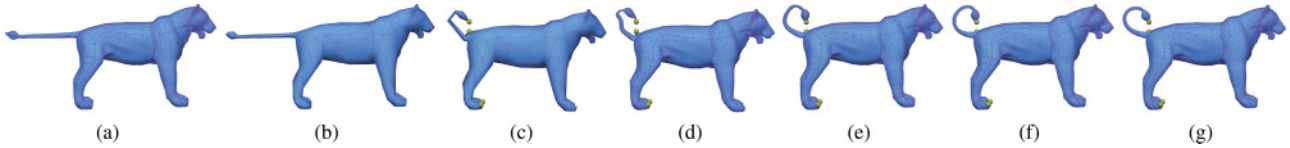


Fig. 18. Comparison of our multi-resolution optimization and direct deformation transfer. (a) The original lion mesh model with 5,000 vertices, (b) simplified model with 10% of the original size, (c) deformed model of the simplified model, (d) direct deformation gradient transfer to the original resolution with handle constraints, (e) our multi-resolution optimization result using (b), (f) our multi-resolution optimization result using a simplified model with 3K triangles, and (g) deformation result on the original mesh.

2005] fails to blend example models with more than 180° rotation, so it is not surprising that MeshIK also fails in this example. The three example models give the information about the head turning from left to right. However in this example, the user drags the head down, which is out of the knowledge from the dataset. As shown in Figure 11, the approach of Fröhlich and Botsch [2011] does not handle such cases well and over-blends the neck. Our method works well by delivering realistic deformation learned from the examples (both interpolation and extrapolation) or smooth deformation when such information is not available.

Figure 18 compares our multi-resolution optimization with direct deformation transfer. We substantially simplify the lion model to only 10% of the original size. Our multi-resolution optimization produces a visually reasonable deformation result, whereas direct deformation transfer produces a deformed result that is not smooth. In practice, we simplify the original shapes to 3K triangles for better

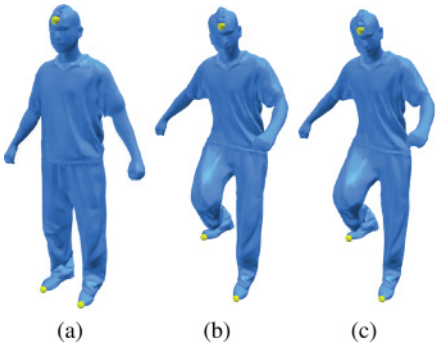


Fig. 19. Data-driven deformation using the "march2" dataset from (a) an input model with handles highlighted, (b) data-driven deformation with 4 basis vectors, and (c) data-driven deformation with 18 basis vectors.

balance of quality and speed, and the obtained result is visually very close to deformation on the original mesh. For the examples in the article, the Hausdorff distance between the multi-resolution optimization result and the direct deformation result of the original mesh is below 1% of the radius of the bounding sphere.

Figure 19 is an example of data-driven deformation using PCA for dimensionality reduction. This dataset contains shapes of a man walking. The differences between deformation results with 4 and 18 basis vectors are almost unnoticeable. This shows that for deformation, a fewer number of basis vectors than for reconstruction is often sufficient.

5.3 Data-Driven Non-Rigid Registration

We compare our method with general surface-based non-rigid registration. Such methods [Bouaziz and Pauly 2013] are generally not data driven. To make a fairer comparison, we choose the example model that is closest to the scan as the initialization for both our and alternative methods. The method used in Bouaziz and Pauly [2013] has a variation that is data driven and uses PCA. However, it is only used for faces where the deformation is relatively subtle. We will demonstrate that such a technique performs worse than not using data for large-scale deformations. Thus by default we compare our method with the non-data-driven version of their method.

Figure 20 shows two examples of non-rigid registration, using the SCAPE dataset [Anguelov et al. 2005] as the model collection. A complete reconstruction of the person treated as a template is obtained using KinectFusion (b) and transferred to all the models in the SCAPE database (see supplementary material for the transferred shapes). A single depth scan is then taken using a Kinect v2 camera (a) and our aim is to find a non-rigid registration that aligns the template to the scan. As the template can substantially differ from the scan, we choose the transferred model closest to the scan as initialization (c). This model, however, still has a significant difference from the scan. Standard non-rigid registration using an ICP-type optimization tends to find wrong correspondences. As a result, with 30 iterations, the result is distorted and differs considerably from the scan (d). With more iterations (40), however, the result becomes worse, with even more distortions (e). Our result nicely registers the template to the scan, thanks to the prior provided by the example models (f). Kinect scans are noisy, which can be better seen when our result is overlaid with the scan (g). Figure 21 shows the registration results of our data-driven method and that of Bouaziz and Pauly [2013] without and with PCA (for the latter the same number of principal axes are used as our method for a fair comparison). It can be clearly seen that both the non-rigid registration and its PCA variant introduce visible artifacts. In particular, the registration results using PCA have shrinking artifacts and look worse. This is because simple PCA linear analysis based on vertex coordinates cannot handle datasets with large rotations.

Figure 22 gives two examples of non-rigid registration to deformed hands (see supplementary material for the transferred example shapes). The captured scans are noisy and incomplete due to occlusions. The standard non-rigid registration converges to some local minima that are still quite far from the target scans. Our data-driven approach properly constrains the deformation space and allows us to find suitable transformations to align the scans. For the example in the bottom row, we also show the registration error distribution as a histogram in Figure 23(a). Our method has more points with low errors than standard non-rigid registration. Figure 23(b) shows how the registration error reduces and converges using our method. Note that since the energy reduces monotonically, our method always converges to some local minimum. As demonstrated by these examples, the use of example models helps to produce well-registered results.

We show an example to demonstrate how our method performs with increasing numbers of examples. We perform experiments using the "march2" dataset from Vlasic et al. [2008]. We select one shape (shown in Figure 24(b)) from the dataset as the *target* shape and remove it from the dataset. The standing shape (as shown in Figure 24(a)) is chosen as the source shape and registered to the target. To avoid bias due to the order of models in the dataset, we randomly order the dataset and run the experiments 20 times and report the average results. For each run, we start from an empty example dataset and incrementally add models in a random order into the dataset one at a time and run our data-driven non-rigid registration algorithm. As discussed in the article, 18 PCA modes capture over 90% of variance in the whole dataset. When the example dataset for data-driven registration contains 18 or fewer models, we do not apply PCA analysis and use all the shapes. When the example dataset contains more than 18 shapes, we obtain PCA-based non-rigid registration results by applying PCA analysis and only keeping the 18 dominant modes. This is then compared with the results obtained without PCA by using all the shapes in the example dataset. We calculate the average Euclidean distance between the registered shape and the ground-truth shape to measure the accuracy. As shown in Figure 24(c), when the number of models for data-driven registration increases, for registrations with and without PCA, the mean Euclidean distances decrease. This is understandable as with more example shapes involved, the information of the deformation space is better covered. The data-driven registration with PCA produces slightly more accurate results than without using PCA, as it is less sensitive to inclusion of dissimilar examples to the example dataset (see Figure 24(c)) while reducing the running times dramatically (see Figure 24(d)).

We perform further quantitative analysis of non-rigid registration. We take each model from the SCAPE dataset [Anguelov et al. 2005] in turn as the target model and use the remaining models as example models. Figure 25 shows such an example. Figure 25(a) is the chosen target model, and Figure 25(b) is the closest model in the remaining example set. Distortions of standard non-rigid registration (Figure 25(c)) and our data-driven registration (Figure 25(d)) results are shown using color coding. The registration error distribution of all the vertices is summarized in Figure 26, which indicates the percentage of vertices (Y-axis) within different fitting errors (X-axis). This shows that for our method substantially more vertices have smaller errors.

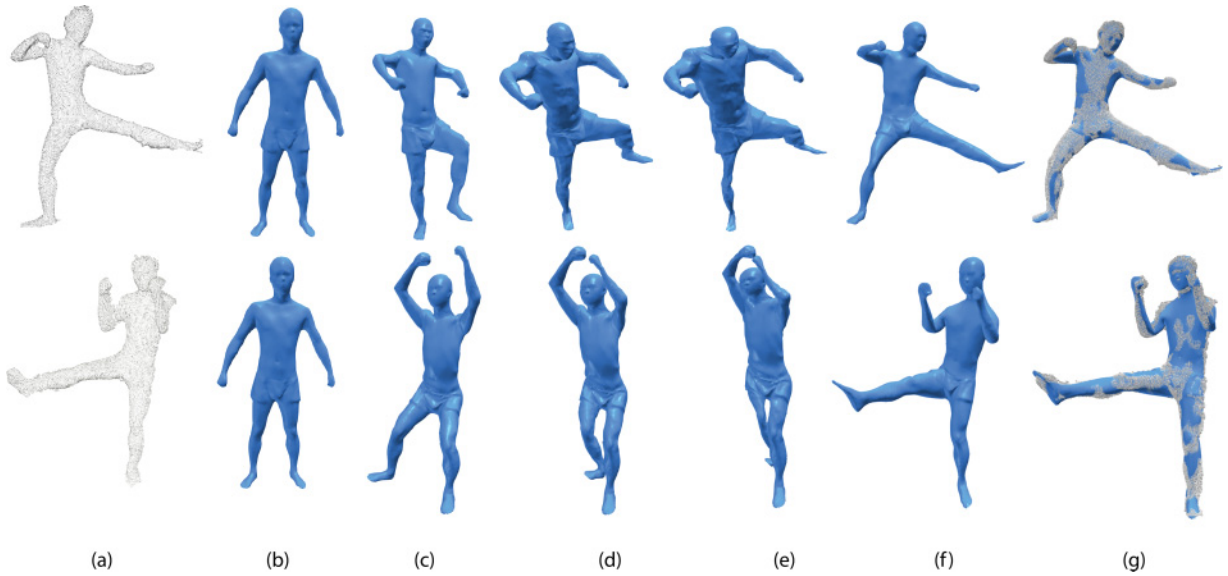


Fig. 20. Data-driven registration of human bodies. (a) A Kinect scan, (b) the template model obtained using KinectFusion, (c) the closest model in the transferred dataset, (d) standard non-rigid registration with 30 iterations, (e) standard non-rigid registration with 40 iterations, (f) our data-driven registration results, and (g) our results with the scans overlaid.

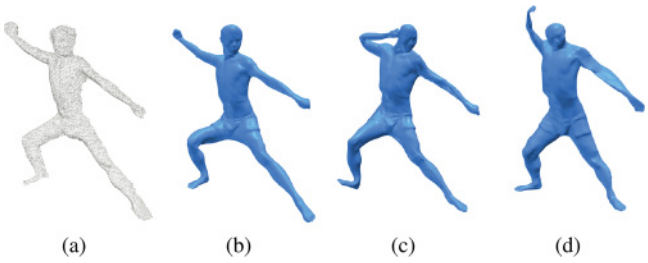


Fig. 21. Data-driven registration comparison. (a) Point cloud captured by a Kinect V2, (b) the registration result of our data-driven registration method, (c) non-rigid registration result of Bouaziz and Pauly [2013] without PCA, and (d) non-rigid registration result of Bouaziz and Pauly [2013] with PCA.

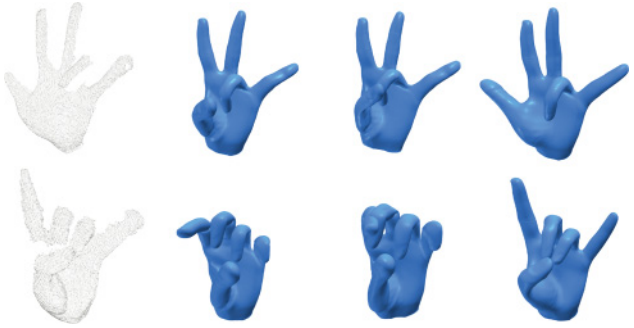


Fig. 22. Data-driven registration of hand. (a) Kinect scans, (b) closest models in the transferred dataset, (c) standard non-rigid registration results, and (d) our data-driven registration results.

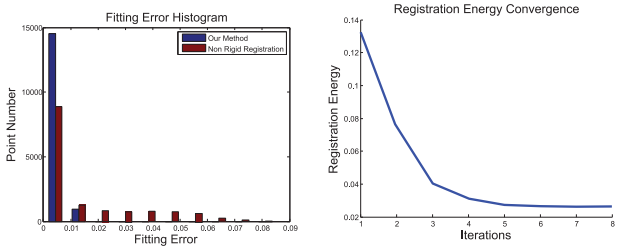


Fig. 23. Quantitative analysis of Figure 22(bottom). (a) Histogram showing vertex registration error distribution and (b) registration energy over iterations.

Please refer to the accompanying video for real-time screen recording and a dynamic presentation of deformation and registration results.

6. CONCLUSION

In this article, we introduce a new rotation-invariant mesh difference representation to encode mesh deformations and a novel reconstruction algorithm that efficiently solves for the vertex positions and local rotations simultaneously. The representation allows us to combine multiple deformations by a linear combination. We propose a data-driven approach by exploiting knowledge in the example models. Significantly better results than state-of-the-art methods are obtained for shape deformation as well as non-rigid registration. The representation also allows analysis of a set of deforming models and extraction of a compact set of bases to represent essential deformation modes in the dataset. Using this approach, mesh manipulation becomes more efficient, especially when a large number of examples are provided.

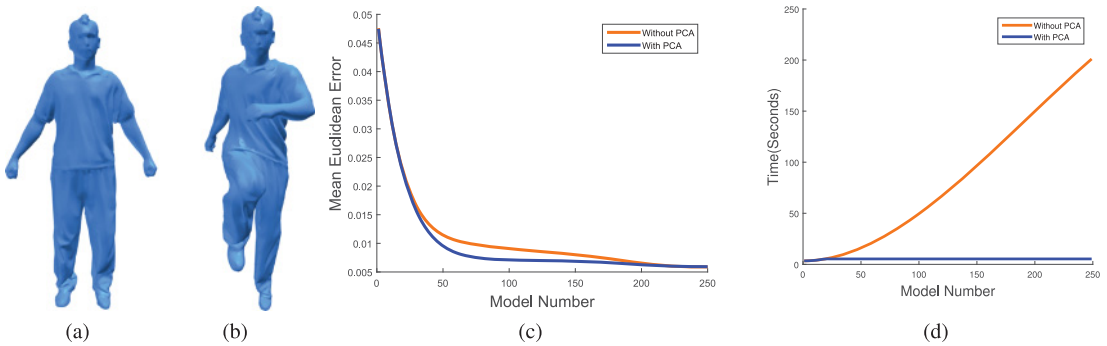


Fig. 24. Registration accuracy and running time comparison without and with PCA. (a) Initial shape for data-driven registration, (b) target shape to be registered that is removed from the example dataset, (c) mean Euclidean error, and (d) running time. Results are averaged over 20 runs.

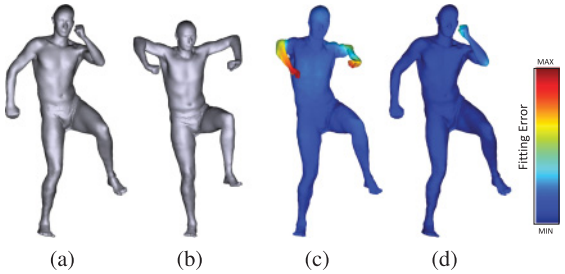


Fig. 25. Leave-one-out non-rigid registration test using the SCAPE dataset [Anguelov et al. 2005]. (a) Chosen target model, (b) the closest example model, (c) standard non-rigid registration result with error visualization, and (d) our result with error visualization.

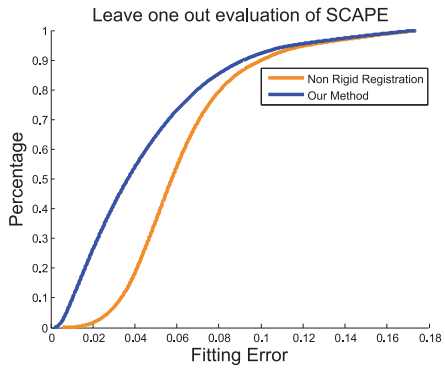


Fig. 26. Leave-one-out evaluation of non-rigid registrations using SCAPE. Vertex fitting error distribution using standard non-rigid registration and our data-driven method.

As demonstrated, our data-driven algorithms effectively exploit knowledge in the example model collection. Our representation allows both interpolation as well as extrapolation and effectively blends multiple deformations. A limitation of our data-driven method is that it may not perform realistically if the dataset is not large enough to cover the target deformation or scan. However, we have demonstrated that in such challenging cases, our method still produces better results than existing data-driven methods. Another limitation is that we currently use gradient descent to solve our problem, which has scope to improve the efficiency. Although this is sufficiently efficient for many interactive applications, and real-time data-driven deformation is achieved with various optimization (matrix pre-decomposition, and optionally multi-resolution optimization and PCA dimensionality reduction), the performance may still be further improved for registration. We will investigate using a GPU-based Gauss-Newton optimizer [Zollhöfer et al. 2014].

ACKNOWLEDGMENTS

This work was performed while D. Liang was an intern at ICT, CAS. The authors thank the reviewers for their constructive comments and suggestions. We also thank Mario Botsch for providing the core code of Fröhlich and Botsch [2011] and Zohar Levi for providing the mesh models in Figure 10.

REFERENCES

Marc Alexa. 2002. Linear combination of transformations. *ACM Trans. Graph.* 21, 3 (2002), 380–387.

Dragomir Anguelov, Praveen Srinivasan, Daphne Koller, Sebastian Thrun, Jim Rodgers, and James Davis. 2005. SCAPE: Shape completion and animation of people. *ACM Trans. Graph.* 24, 3 (2005), 408–416.

O. K.-C. Au, C. L. Tai, L. Liu, and H. Fu. 2006. Dual Laplacian editing for meshes. *IEEE Trans. Vis. Comp. Graph.* 12, 3 (2006), 386–395.

Ilya Baran, Daniel Vlasic, Eitan Grinspun, and Jovan Popović. 2009. Semantic deformation transfer. *ACM Trans. Graph.* 28, 3 (2009), 36:1–36:6.

P. J. Besl and Neil D. McKay. 1992. A method for registration of 3-D shapes. *IEEE Trans. Pattern Anal. Mach. Intell.* 14, 2 (1992), 239–256.

Charles Bloom, Jonathan Blow, and Casey Muratori. 2004. Errors and Omissions in Marc Alexa's "Linear Combination of Transformations." http://www.cbloom.com/3d/techdocs/lcot_errors.pdf. (2004).

Mario Botsch and Olga Sorkine. 2008. On linear variational surface deformation methods. *IEEE Trans. Vis. Comp. Graph.* 14, 1 (2008), 213–230.

Sofien Bouaziz, Sebastian Martin, Tiantian Liu, Ladislav Kavan, and Mark Pauly. 2014. Projective dynamics: Fusing constraint projections for fast simulation. *ACM Trans. Graph.* 33, 4, Article 154 (2014), 11 pages.

Sofien Bouaziz and Mark Pauly. 2013. Dynamic 2d/3d registration for the kinect. In *ACM SIGGRAPH 2013 Courses*. 21:1–21:14.

Isaac Chao, Ulrich Pinkall, Patrick Sanan, and Peter Schröder. 2010. A simple geometric model for elastic deformations. *ACM Trans. Graph.* 29, 4 (2010), 38:1–38:6.

Wei-Wen Feng, Byung-Uck Kim, and Yizhou Yu. 2008. Real-time data driven deformation using kernel canonical correlation analysis. *ACM Trans. Graph.* 27, 3 (2008), 91:1–91:9.

Oren Freifeld and Michael J. Black. 2012. Lie bodies: A manifold representation of 3d human shape. In *ECCV (1)*. 1–14.

Stefan Fröhlich and Mario Botsch. 2011. Example-driven deformations based on discrete shells. *Comput. Graph. Forum* 30, 8 (2011), 2246–2257.

James Gain and Dominique Bechmann. 2008. A survey of spatial deformation from a user-centered perspective. *ACM Trans. Graph.* 27, 4 (2008), 107:1–107:21.

Lin Gao, Yu-Kun Lai, Qi-Xing Huang, and Shi-Min Hu. 2013. A data-driven approach to realistic shape morphing. *Comput. Graph. Forum* 32, 2 (2013), 449–457.

Michael Garland and Paul S. Heckbert. 1997. Surface simplification using quadric error metrics. In *Proc. ACM SIGGRAPH*. 209–216.

Nils Hasler, Carsten Stoll, Martin Sunkel, Bodo Rosenhahn, and Hans-Peter Seidel. 2009. A statistical model of human pose and body shape. *Comput. Graph. Forum* 28, 2 (2009), 337–346.

Roger A. Horn and Charles R. Johnson (Eds.). 1986. *Matrix Analysis*. Cambridge University Press, New York, NY.

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. *ACM Trans. Graph.* 25, 3 (2006), 1126–1134.

Takeo Igarashi, Tomer Moscovich, and John F. Hughes. 2005. As-rigid-as-possible shape manipulation. In *Proc. ACM SIGGRAPH*. 1134–1141.

Shahram Izadi, David Kim, Otmar Hilliges, David Molyneaux, Richard A. Newcombe, Pushmeet Kohli, Jamie Shotton, Steve Hodges, Dustin Freeman, Andrew J. Davison, and Andrew W. Fitzgibbon. 2011. KinectFusion: Real-time 3d reconstruction and interaction using a moving depth camera. In *Proc. ACM Symposium on User Interface Software and Technology*. 559–568.

Ben Jones, Jovan Popovic, James McCann, Wilmot Li, and Adam Bargteil. 2015. Dynamic sprites: Artistic authoring of interactive animations. *Comput. Anim. Virt. Worlds* 26 (2015), 97–108.

Scott Kircher and Michael Garland. 2008. Free-form motion processing. *ACM Trans. Graph.* 27, 2 (2008), 12:1–12:13.

Yuki Koyama, Kenshi Takayama, Nobuyuki Umetani, and Takeo Igarashi. 2012. Real-time example-based elastic deformation. In *Symposium on Computer Animation*. 19–24.

Zohar Levi and Craig Gotsman. 2015. Smooth rotation enhanced as-rigid-as-possible mesh animation. *IEEE Trans. Vis. Comp. Graph.* 21, 2 (2015), 264–277.

Hao Li, Bart Adams, Leonidas J. Guibas, and Mark Pauly. 2009. Robust single-view geometry and motion reconstruction. *ACM Trans. Graph.* 28, 5 (2009), 175:1–175:10.

Hao Li, Robert W. Sumner, and Mark Pauly. 2008. Global correspondence optimization for non-rigid registration of depth scans. *Comput. Graph. Forum* 27, 5 (2008), 1421–1430.

Hao Li, Etienne Vouga, Anton Gudym, Linjie Luo, Jonathan T. Barron, and Gleb Gusev. 2013. 3D self-portraits. *ACM Trans. Graph.* 32, 6 (2013), 187:1–187:9.

Yaron Lipman, Olga Sorkine, David Levin, and Daniel Cohen-Or. 2005. Linear rotation-invariant coordinates for meshes. *ACM Trans. Graph.* 24, 3 (2005), 479–487.

Matthew Loper, Naureen Mahmood, and Michael J. Black. 2014. MoSh: Motion and shape capture from sparse markers. *ACM Trans. Graph.* 33, 6 (2014), 220:1–220:13.

Sebastian Martin, Bernhard Thomaszewski, Eitan Grinspun, and Markus H. Gross. 2011. Example-based elastic materials. *ACM Trans. Graph.* 30, 4 (2011), 72:1–72:8.

Richard M. Murray, S. Shankar Sastry, and Li Zexiang. 1994. *A Mathematical Introduction to Robotic Manipulation* (1st ed.). CRC Press, Boca Raton, FL.

Helmut Pottmann, Qi-Xing Huang, Yong-Liang Yang, and Shi-Min Hu. 2006. Geometry and convergence analysis of algorithms for registration of 3d shapes. *Int. J. Comput. Vision* 67, 3 (2006), 277–296.

Raif M. Rustamov, Maks Ovsjanikov, Omri Azencot, Mirela Ben-Chen, Fr'ed'eric Chazal, and Leonidas Guibas. 2013. Map-based exploration of intrinsic shape differences and variability. *ACM Trans. Graph.* 32, 4 (2013), 72:1–72:12.

D. C. Schneider and P. Eisert. 2009. Fast nonrigid mesh registration with a data-driven deformation prior. In *Proc. ICCV Workshops*. 304–311.

Christian Schumacher, Bernhard Thomaszewski, Stelian Coros, Sebastian Martin, Robert W. Sumner, and Markus H. Gross. 2012. Efficient simulation of example-based materials. In *Symposium on Computer Animation*. 1–8.

Xiaohan Shi, Kun Zhou, Yiying Tong, Mathieu Desbrun, Hujun Bao, and Baining Guo. 2008. Example-based dynamic skinning in real time. *ACM Trans. Graph.* 27, 3 (2008), 29:1–29:8.

Olga Sorkine and Marc Alexa. 2007. As-rigid-as-possible surface modeling. In *Proc. Symposium on Geometry Processing*. 109–116.

O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rossl, and H.-P. Seidel. 2004. Laplacian surface editing. In *Proc. Symposium on Geometry Processing*. 175–184.

Robert W. Sumner and Jovan Popović. 2004. Deformation transfer for triangle meshes. *ACM Trans. Graph.* 23, 3 (2004), 399–405.

Robert W. Sumner, Johannes Schmid, and Mark Pauly. 2007. Embedded deformation for shape manipulation. *ACM Trans. Graph.* 26, 3 (2007), 80:1–80:7.

Robert W. Sumner, Matthias Zwicker, Craig Gotsman, and Jovan Popović. 2005. Mesh-based inverse kinematics. *ACM Trans. Graph.* 24, 3 (2005), 488–495.

G. K. L. Tam, Zhi-Quan Cheng, Yu-Kun Lai, F. C. Langbein, Yonghuai Liu, D. Marshall, R. R. Martin, Xian-Fang Sun, and P. L. Rosin. 2013. Registration of 3d point clouds and meshes: A survey from rigid to nonrigid. *IEEE Trans. Vis. Comp. Graph.* 19, 7 (2013), 1199–1217.

Demetri Terzopoulos, John Platt, Alan Barr, and Kurt Fleischer. 1987. Elastically deformable models. In *Proc. ACM SIGGRAPH*. 205–214.

Daniel Vlasic, Ilya Baran, Wojciech Matusik, and Jovan Popović. 2008. Articulated mesh animation from multi-view silhouettes. *ACM Trans. Graph.* 27, 3 (2008), 97:1–9.

Christoph Von-Tycowicz, Christian Schulz, Hans-Peter Seidel, and Klaus Hildebrandt. 2015. Real-time nonlinear shape interpolation. *ACM Trans. Graph.* 34, 3 (2015), 34:1–34:10.

Xiaolin Wei, Peizhao Zhang, and Jinxiang Chai. 2012. Accurate realtime full-body motion capture using a single depth camera. *ACM Trans. Graph.* 31, 6 (2012), 188:1–188:12.

Thibaut Weise, Sofien Bouaziz, Hao Li, and Mark Pauly. 2011. Realtime performance-based facial animation. *ACM Trans. Graph.* 30, 4 (2011), 77:1–77:10.

Tim Winkler, Jens Drieseberg, Marc Alexa, and Kai Hormann. 2010. Multi-scale geometry interpolation. *Comput. Graph. Forum* 29, 2 (2010), 309–318.

Yizhou Yu, Kun Zhou, Dong Xu, Xiaohan Shi, Hujun Bao, Baining Guo, and Heung-Yeung Shum. 2004. Mesh editing with poisson-based gradient field manipulation. *ACM Trans. Graph.* 23, 3 (2004), 644–651.

Li Zhang, Noah Snavely, Brian Curless, and Steven M. Seitz. 2004. Spacetime faces: High resolution capture for modeling and animation. *ACM Trans. Graph.* 23, 3 (2004), 548–558.

Qing Zhang, Bo Fu, Mao Ye, and Ruigang Yang. 2014. Quality dynamic human body modeling using a single low-cost depth camera. In *CVPR*. 676–683.

Wenjing Zhang, Jianmin Zheng, and Nadia Magnenat Thalmann. 2015. Real-time subspace integration for example-based elastic material. *Comput. Graph. Forum* 34, 2 (2015), 395–404.

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. *ACM Trans. Graph.* 24, 3 (2005), 496–503.

Michael Zollhöfer, Matthias Niessner, Shahram Izadi, Christoph Rehmann, Christopher Zach, Matthew Fisher, Chenglei Wu, Andrew Fitzgibbon, Charles Loop, Christian Theobalt, and Marc Stamminger. 2014. Real-time non-rigid reconstruction using an RGB-D camera. *ACM Trans. Graph.* 33, 4 (2014), 156:1–156:12.

Received February 2016; accepted March 2016