Detail-Preserving Controllable Deformation from Sparse Examples

Haoda Huang, KangKang Yin, Ling Zhao, Yue Qi, *Member*, *IEEE*,
Yizhou Yu, *Senior Member*, *IEEE*, and Xin Tong, *Member*, *IEEE*

Abstract—Recent advances in laser scanning technology have made it possible to faithfully scan a real object with tiny geometric details, such as pores and wrinkles. However, a faithful digital model should not only capture static details of the real counterpart but also be able to reproduce the deformed versions of such details. In this paper, we develop a data-driven model that has two components; the first accommodates smooth large-scale deformations and the second captures high-resolution details. Large-scale deformations are based on a nonlinear mapping between sparse control points and bone transformations. A global mapping, however, would fail to synthesize realistic geometries from sparse examples, for highly deformable models with a large range of motion. The key is to train a collection of mappings defined over regions locally in both the geometry and the pose space. Deformable fine-scale details are generated from a second nonlinear mapping between the control points and per-vertex displacements. We apply our modeling scheme to scanned human hand models, scanned face models, face models reconstructed from multiview video sequences, and manually constructed dinosaur models. Experiments show that our deformation models, learned from extremely sparse training data, are effective and robust in synthesizing highly deformable models with rich fine features, for keyframe animation as well as performance-driven animation. We also compare our results with those obtained by alternative techniques.

Index Terms—Detail-preserving deformation, controllable skinning, learning from sparse examples, CCA regression.

1 Introduction

Technology for laser range scanning has been significantly improved over the last decade in terms of both precision and speed. It has become possible to faithfully scan a real object with tiny geometric details, such as pores and wrinkles. However, many real objects including most natural organisms deform. A faithful digital model should not only capture static details of the real counterpart but also reproduce the deformed versions of such details. Data-driven methods are well suited for this purpose for two reasons. First, it would be extremely expensive to physically simulate deformations of such high-resolution details. Second, fine-scale deformations of different objects follow different styles. A data-driven method incorporates the unique characteristics of different types of deformation.

- H. Huang is with the Microsoft Research Asia, 507 Central Avenue, Apt Q, Mountain View, CA 94043. E-mail: haoda.huang@gmail.com.
- K. Yin is with the Department of Computer Science, School of Computing, National University of Singapore, Computing 1, 13 Computing Drive, Singapore 117417. E-mail: kkyin@comp.nus.edu.sg.
- L. Zhao and Y. Qi are with the State Key Laboratory of Virtual Reality Technology and Systems, School of Computer Science and Engineering, Beihang University, New Main Building, No. 37 Xueyuan Road, Haidian District, Beijing 100191, China. E-mail: {zhaoling, qy}@vrlab.buaa.edu.cn.
- Y. Yu is with the University of Hong Kong and Department of Computer Science, University of Illinois at Urbana-Champaign, 201 North Goodwin Ave., Urbana, IL 61801. E-mail: yyz@illinois.edu, yizhouy@acm.org.
- X. Tong is with the Microsoft Research Asia, B2-13171, No. 5. Dan Ling Street, ZhongGuanCun, Beijing 100080, China. E-mail: xtong@microsoft.com.

Manuscript received 7 Nov. 2011; revised 8 Feb. 2012; accepted 14 Feb. 2012; published online 9 Mar. 2012.
Recommended for acceptance by A.W. Bargteil and M. van de Panne.
For information on obtaining reprints of this article, please send e-mail to: tvcg@computer.org, and reference IEEECS Log Number TVCGSI-2011-11-0273.
Digital Object Identifier no. 10.1109/TVCG.2012.88.

There exist two major challenges in building high-resolution data-driven deformation models. First, only a limited amount of training data is typically available due to the amount of time and effort required to scan high-resolution details. Training data-driven models with sparse examples can easily result in inaccurate models that produce poor predictions. Second, from an animation perspective, we would like to generate realistic deformations from a sparse set of markers. This calls for a data-driven model that can correlate low-dimensional control signals with high-dimensional deformation details.

We propose a robust deformation framework, as shown in Fig. 1, to tackle the above challenges. Our data-driven model has two components; the first accommodates smooth large-scale deformations and the second captures high-resolution details. Large-scale deformations are based on linear blend skinning (LBS) and nonlinear mappings between sparse control points and bone transformations, as shown in Fig. 1b. To alleviate poor fitting caused by training with sparse nonlinear data, we train a collection of local mappings defined over the manifold of seen example poses. A *pose* in this paper refers to a specific shape of a surface model or a specific configuration of its control points. Each of the local mappings takes one of the training poses as its reference pose, and its nearby poses as training inputs. The deformation associated with a novel pose is then predicted using a weighted mixture of local mappings defined for the pose closest to the target pose. This training process may, however, mistakenly learn false coupling among distant control points from sparse training examples even when constrained within local pose subspaces. We address this problem by learning correlations only between control points and deformation regions geometrically located nearby.

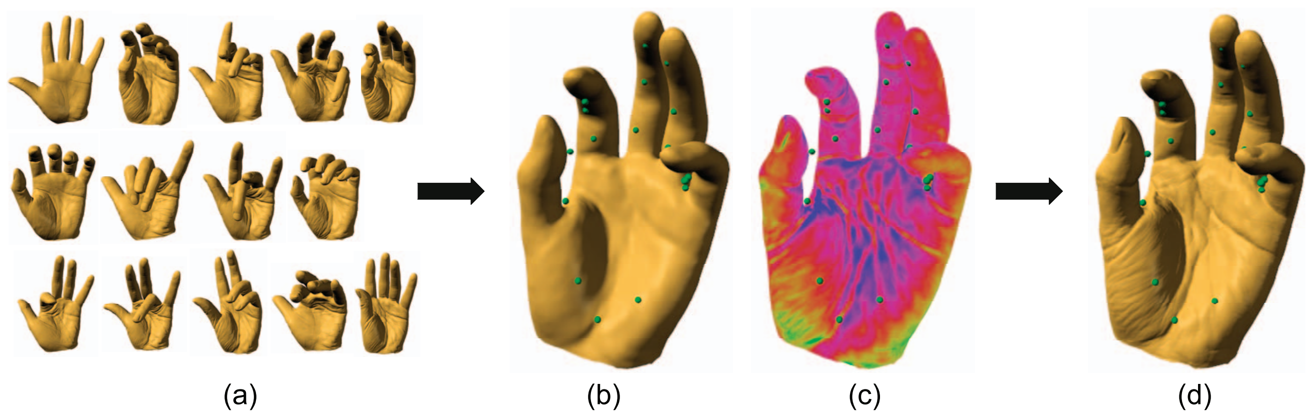


Fig. 1. Given sparse training examples (a), we train a collection of deformation models at two layers. Given a new configuration of the control points, these models can generate smooth large-scale deformations (b) and high-resolution displacements (c), which are then combined to produce deformed models with rich details (d).

High-resolution deformable details are modeled in a separate training pass as shown in Fig. 1c, which learns a per-vertex nonlinear mapping between control points and per-vertex displacements. Several choices exist in terms of input signals for displacement modeling. We have confirmed that a direct mapping between the control points and the displacements is more effective than a cascaded mapping where large-scale deformation predictions are used to drive the deformation of details. We prefer 1D displacements along vertex normals to 3D displacement vectors to increase the robustness to noise and reduce the memory requirement.

Our deformation modeling scheme is first proposed in [1]. However, only scanned human hand models are tested there. In this paper, we further test the framework with scanned face models, face models reconstructed from video, and artificial data sets. To the best of our knowledge, few previous works test on both human hands and human faces. Human hands have large degrees of freedom (DoFs), large ranges of motion, and highly deformable wrinkles. Human faces pose a stringent requirement for the synthesized results to be deemed realistic, because the human perceptual system is extremely familiar and sensitive to human facial expressions. Our framework is capable of synthesizing high-quality mesh animations for both hands and faces with rich and varying details, from sparse training examples. The additional test with artificial data illustrates how animators can use our framework to reduce their workload in animating models with existing animation tools. They now only need to construct a small number of key examples, and all other frames in an animation sequence can be automatically generated by inferring both the large-scale deformation and the small-scale details.

Our experiments use between $8 \sim 26$ control points to control hand, face, and full-body deformation and animation. If the model to be controlled has an inherent skeleton structure, we can further reduce the number of control points with the help of an Inverse Kinematics (IK) module. Our choice of low-dimensional and easy-to-manipulate control signals results in intuitive keyframe animation tools readily adoptable into traditional animation pipelines, and provides a promising way for performance-driven mesh animation as well.

2 Related Work

Data-driven mesh skinning. For real-time applications, Linear Blend Skinning is widely used by artists because of its simplicity and efficiency. However, the original LBS suffers from "candy-wrapper" artifacts. Pose Space Deformation (PSD) improves skinning quality by integrating LBS and RBF-based interpolation [2]. More advanced example-based techniques [3], [4], [5], [6] have been effectively integrated with mesh deformation algorithms to further improve the quality of skinning. EigenSkin models the residual errors of LBS using principal component analysis [7]. Kurihara and Miyata [8] use a per-vertex weighting scheme for PSD to animate hand meshes from sparse examples. DrivenShape [9] exploits known correspondences between two sets of deformation examples. These methods, however, do not support direct manipulations or handle fine-scale features at the wrinkle level. Most of them require dense example data as well.

Data-driven methods that support direct manipulation with low-dimensional control signals [10], [11], [12] are the closest in spirit to our own. Mesh-based Inverse Kinematics (MESHIK) adopts a *global* weighting scheme where all vertices from the same example mesh are given the same weight [10], [11]. Such global scheme produces artifacts when modeling from sparse examples, for both large-scale and fine-scale deformations. In the absence of a skeleton, Feng et al. [12] build a *global* data-driven mapping between sparse control points and proxy bone transformations for predicting novel surface deformations in real time. There are two limitations with this method. First, prediction errors may increase significantly when a novel control point configuration deviates far away from the reference configuration. Second, given sparse training examples, false dependencies between distant object parts may be mistakenly enforced by the global mapping. Our method addresses these challenges by learning local deformation models in both the geometry and the pose space. We compare our results with respect to those obtained from MESHIK and [12] in Section 8.

Lau et al. have pointed out as a future direction to learn region-based models to allow fine-grained control over local geometry and to improve the generalization ability of

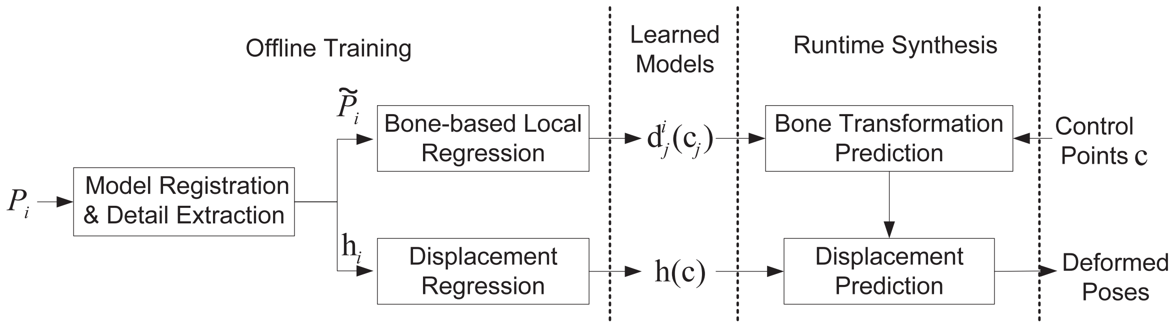


Fig. 2. The training phase and the runtime phase of our deformation framework. P_i: high-resolution training examples; $\widetilde{P}_i$: low-resolution registered meshes; $\mathbf{h}_i$: displacement maps; $\mathbf{c}$: control points; $\mathbf{d}^i_j(\mathbf{c}_j)$: bone-based large-scale deformation models; $\mathbf{h}(\mathbf{c})$: vertex-level models for fine details.

their models [13]. More recently, Huang et al. [14] combine high-resolution 3D face scans and high-speed motion captured markers to produce high-fidelity 3D facial performances in a blendshape interpolation framework. They report that a region-based fine-scale mesh registration process produces much better results than a global registration approach. It is also reported in [15] that region-based PCA models generalize better than its holistic counterpart, and give the user intuitive localized control.

Detail modeling for mesh animation. Detail modeling for mesh animation has been attracting more and more research effort in recent years, mainly due to the advances in acquisition techniques. Several multiscale deformation schemes have been proposed for face modeling [16], [17], [18], and they all represent large-scale deformations with thin shell models. Fine-scale geometric details such as wrinkles are modeled using 2D splines [16], pose-space interpolations [17], or polynomial displacement maps [18]. We target highly deformable models with larger ranges of motion and deformation, such as human hands, for which techniques developed for 2.5D surfaces such as faces cannot be applied directly. Furthermore, these methods require dense markers and training data for motion tracking and deformation modeling, while our method only needs a sparse set of control points and a sparse set of training examples. More recently, body parts or full-body geometries can be reconstructed from single-view or multiple-view dense video sequences [19], [20], [21], [22]. The reconstructed geometries usually lack fine details, and do not generalize beyond seen examples. Another line of research, such as [23], simulates the motion and deformation of muscles and tendons for human hands. It is not clear how to extend this physics and biomechanics-based approach to incorporate wrinkle-level details.

Data acquisition. Generally speaking, acquiring high-resolution 3D models with fine features is difficult, expensive, and time consuming. Structured light and photometric stereo are commonly used to capture 3D geometries, especially for facial expressions. The quality of the models depends on the equipment and reconstruction algorithms used. A template-based method is employed in [24] to produce point correspondences across an entire video sequence without using any markers, while [18] uses 178 markers for registration. Golovinskiy et al. capture static faces with a commercial face-scanning system to model aging effects [25]. Deformations, large or small, are not considered there. Park and Hodgins use a commercial motion capture system and 350 markers to capture medium-scale muscle deformations for full-body motions [26], but fine-scale skin movements are hard to capture using motion capture systems alone.

We use a high-precision commercial 3D scanner to capture hand models. We would like to emphasize that our deformation technique is independent of the underlying geometry acquisition method, and can animate models obtained by various means. For example, one set of the face models we use is reconstructed from multiview video sequences [24]; the dinosaur models we test are manually constructed by an artist.

3 Overview

Our system consists of an offline training stage, and an online synthesis stage, as shown in Fig. 2. The training examples, P_i, are high-resolution meshes with rich details. We will describe several methods for obtaining such examples in Section 4. These examples P_i are first registered with respect to each other, in terms of both large-scale and fine-scale features. This model registration process will be described in Section 5. We denote the output of model registration as $\widetilde{P}_i$, which are a collection of low-resolution smooth meshes of the same topology. The deformation learning process consists of two layers: bone-based transformation modeling for large-scale deformations, and displacement modeling for fine-scale details. The low-resolution meshes $\widetilde{P}_i$ are used to train the large-scale deformation models. To train the fine-scale detail models, we extract the differences between P_i and $\widetilde{P}_i$ as displacement maps $\mathbf{h}_i$, which capture high-frequency deformation details.

Both deformation layers learn regression models with the same set of control knobs, i.e., the control points $\mathbf{c}$. A global regression models correlations between the full control point vector $\mathbf{c}$ and transformations $\mathbf{d}_j$ of bone b_j from all example poses, and generates a single prediction model $\mathbf{d}_j(\mathbf{c})$ for each bone. In contrast, we train a collection of models $\mathbf{d}^i_j(\mathbf{c}_j)$, where $i = 1, \dots, \#poses,\ j = 1, \dots, \#bones$. These models are local in both the geometry space and the pose space. The learning methods will be given in Section 6.1, together with necessary implementation details. Building local deformation models in both the geometry space and the pose space effectively eliminates false coupling of independent object parts and severe model mismatches for nonlinear sparse training data.

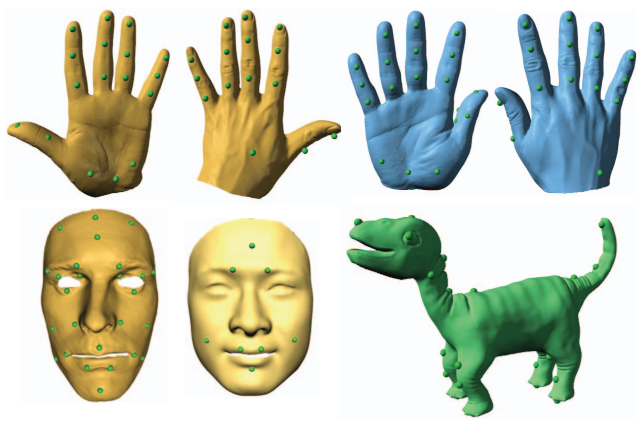


Fig. 3. Our testing models and their control points. From top to bottom and left to right: Hand-I front and back; Hand-II front and back; Face-I; Face-II; Dinosaur.

Displacement maps $\mathbf{h}_i$ are modeled in another pass of regression as $\mathbf{h}(\mathbf{c})$. We will detail this process in Section 6.2. This pass of vertex-level displacement regression is to model the myriad of variations of fine details, such as wrinkles and palm lines, which are beyond the modeling capability of bone-based linear blend skinning.

At runtime, new control point configurations drive the learned models $\mathbf{d}^i_j(\mathbf{c}_j)$ and $\mathbf{h}(\mathbf{c})$ to produce new poses with plausible large-scale deformations as well as realistic fine features. Section 7 describes the necessary formulas for deformation synthesis.

4 Training Data Acquisition

We use five sets of training data to demonstrate the capability of our deformation algorithm: scanned hand models, scanned face models, reconstructed face models from multiview videos, and artist-designed dinosaur models. Fig. 3 shows our models at their reference pose. The control points, represented by the green dots, are defined on the low-resolution reference pose. The same set of control points are used to learn and drive both large-scale deformations and high-resolution displacements.

Capturing hand models with fine wrinkles is difficult due to severe self-occlusions between fingers and the need of accurate registration for data captured from a not-entirely-static hand. We thus use the traditional art of body casting. Negative silicone rubber molds were first created from various hand poses. Then, plaster models were casted from the silicone molds. These models contain fine surface details such as finger prints. We then used a Konica-Minolta Range 7 laser scanner to scan the plaster hand models. The scanner can scan one region of a 3D object in about 2 seconds with high accuracy ($\pm 40\ \mu$m). The scanning software then processes and merges the point clouds and generates surface models. Despite the claimed high precision of our scanner, the finest details such as finger prints were lost in the scanned models, most likely due to noise introduced by our handheld scanning process. Luckily, there were still enough interesting details present in our final scans. These scans are at extremely high resolution of around 900K vertices. Experimentally, we found that downsampling these meshes to about 200K vertices [27] did not lead to any visually noticeable difference from the original scans. So, we simply used the 200K meshes as our training examples P_i.

We captured two hands from two male subjects, both graduate students in their twenties. The upper row of Fig. 3 shows the hand models at their reference pose. Hereafter, we denote the left hand of the first subject as hand-I, and the right hand of the second subject as hand-II. For each hand, 14 highly detailed mesh models were prepared as training data for deformation learning. We manually specified fewer than 20 mesh vertices as control points as shown in Fig. 3. There are two sets of control points defined for each hand: one set all on the palm, and the other set all on the back. Our deformation synthesis system works equally well with both sets of control points. The existence of two sets was basically the result of moving markers from the palm side to the back of the hand to drive our deformation system with motion captured control points. Finger movements caused severe self-occlusion in motion capturing the markers on the palm side.

We use face models from two subjects to test our framework. The first set of face examples, called face-I models hereafter, were generously shared with us by the authors of [14]. This data set is called sequence Matt in [14] and used as their teaser example. It contains 21 high-resolution face scans of about 70K vertices, and 40 seconds of motion captured facial expressions. The high-resolution face scans contain enough wrinkles and details for us to test our layered deformation modeling. The motion capture sequence contains 111 marker trajectories, from which we manually selected 26 markers as control points as shown in Fig. 3.

The second set of face examples, called face-II models hereafter, were shared with us by the authors of [24]. Their capture system employs synchronized video cameras and structured light projectors to record videos of a moving face from multiple viewpoints. A spacetime stereo technique first derives high-quality depth maps from the structured light video sequences. A surface fitting and tracking procedure then combines the depth maps with optical flow to create face models with vertex correspondence. Thus, model registration is not needed for this set of face models. We manually selected six extreme expressions from the original reconstructed sequence as the training examples. Eight vertices around the head and the mouth regions of the neutral face were manually specified as the control points, as shown in Fig. 3. These reconstructed faces from video are about 24K in resolution, however, and do not contain fine-scale details such as wrinkles. Thus, we only use this data set to validate the large-scale deformation modeling component of our framework.

The dinosaur models were constructed by an artist in two steps. First, the artist created a smooth dinosaur mesh of about 34K vertices, and then deformed it into 12 different poses. Next, these meshes were subdivided to about 136K vertices, and sculpted with geometric details. The model registration step was not needed for this data set either. The original 13 low-resolution meshes serve as $\widetilde{P}_i$, and the subdivided high resolution ones with details are used as P_i. We manually specified 24 vertices from smooth regions of the reference mesh as control points, as shown in Fig. 3.

5 Model Registration and Detail Extraction

Training models that are independently scanned initially reside in different coordinate systems. We therefore rotate and translate the training examples $P_i, i = 0...n-1$ into the coordinate frame of a chosen reference mesh P_0. More specifically, we interactively identified a small number of corresponding points between each training mesh and P_0 to resolve the rigid transformations between them, so that the differences among the rigidly transformed meshes are the deformations we wish to model.

We further employ deformation transfer [28] to build per-vertex correspondences among the training models. Note that deformation transfer enforces a smoothness constraint on nearby vertices, and is thus not suitable for meshes with high-frequency details. Therefore, we apply mesh retiling and Laplacian smoothing techniques [27] to obtain a collection of smooth meshes, $\overline{P_i}$, at a lower resolution. The smooth reference model $\overline{P_0}$ is then deformed toward each training model $\overline{P_i}$, and we denote the deformed reference models as $\widetilde{P}_i$. $\widetilde{P}_i$ will be used to model large-scale deformations, each of which possesses the shape of $\overline{P_i}$, but the topology of $\overline{P_0}$. A homogeneous mesh topology also facilitates fine-scale feature extraction for detail modeling.

To extract the differences between the original high-resolution meshes P_i and their smoothed low-resolution version $\widetilde{P}_i$, we subdivide $\widetilde{P}_i$ to retrieve the resolution of the original mesh P_i. At each vertex of the subdivided $\widetilde{P}_i$, a per-vertex displacement with respect to P_i is calculated along the vertex normals. We denote these displacement maps as $\mathbf{h}_i$, which will be used to train deformation models for fine surface features.

6 Deformation Modeling

The modeling component of our deformation framework consists of two layers: large-scale bone-based deformation modeling, and fine-scale vertex-based displacement modeling.

6.1 Large-Scale Deformation Modeling

Large-scale low-frequency deformations are usually generated from bone and muscle motions. We therefore follow the conventional bone-based linear blend skinning to generate large-scale deformation. From the registered input models $\widetilde{P}_i$, we obtain transformations of all the individual triangles. We then cluster triangles of similar rigid transformations to form abstract bones [12], [29], as shown in Fig. 4a. Note that the abstract bones do not conform to the biological bones anatomically. For instance, more than a thousand bones were generated for the hand models. Each bone acts as an abstract representation for rigid transformations, and its influence weights for a vertex are obtained by minimizing the total fitting error of vertex positions using all the examples. The large number of abstract bones is to guarantee the accuracy of the large-scale deformation models. Because they are automatically generated, no extra work is required from the user.

Denote the fitted influence bone set for vertex v as $B(v)$, and its skinning weight from bone b_j in $B(v)$ as w_j. The skinned vertex position v is computed using a weighted average of rigid transformations from its influence bones

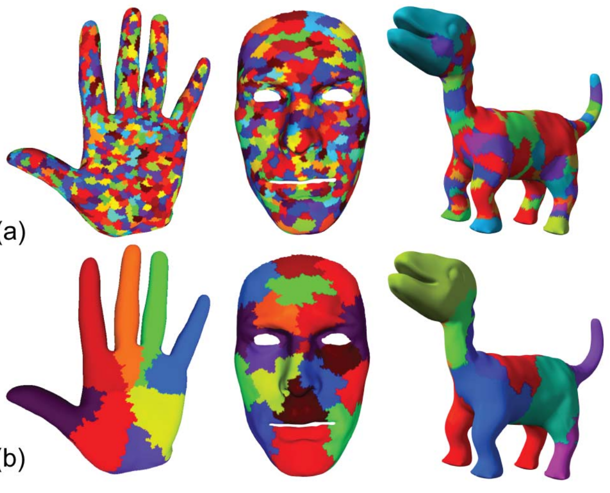


Fig. 4. Color-coded abstract bones (a) and regions sharing the same set of local control points (b).

$$v = \sum_{j \in B(v)} w_j \mathbf{T}_j v^r, \tag{1}$$

where $\mathbf{T}_j$ is the transformation matrix of bone b_j, and v^r represents the vertex position in the reference pose.

The task of the large-scale deformation modeling is to learn models of the form $\mathbf{T}_j(\mathbf{c})$ to predict bone transformations from control points $\mathbf{c}$. We choose to use a quaternion representation for bone rotations $\mathbf{d}_j$ only, and solve for bone translations using a Poisson solver. Predicted bone rotations and solved bone translations together can then be converted to $\mathbf{T}_j$ appropriately for skinning.

As mentioned in Sections 1 and 3, learning large-scale deformation models globally from nonlinear sparse training data suffers from false correlation and poor fitting. Fig. 6a previews some artifacts resulted from such poor global models. In the pursuit of a robust system for deformation modeling and synthesis, we found that local learning in both the geometry space and the pose space is crucial.

6.1.1 *Local Regression in the Geometry Space*

The degrees of freedom of human hands are extremely large. There are at least 21 DoFs associated with the skeletal structure of a hand, five for the thumb and four for each of the fingers. There are correlations between nearby structures, but faraway bones such as the thumb and the pinky can move relatively independently. When the training data are extremely sparse, basic correlation analysis may, however, enforce unnecessary constraints on the movement of a bone with respect to that of a distant control point, and result in severe model mismatches. We therefore only train prediction models for bone rotations from spatially close control points, to decouple the accidental correlations between distant bones and control points seen from a few examples. To locate the local control points for a particular bone b_j, the center location of all the vertices controlled by this bone is calculated, and its nearest vertex on the mesh is denoted as v_j. The k_c nearest control points to v_j, measured by the geodesic distance, are collected as the influence control point set $\mathbf{c}_j$ for b_j. Instead of learning deformation

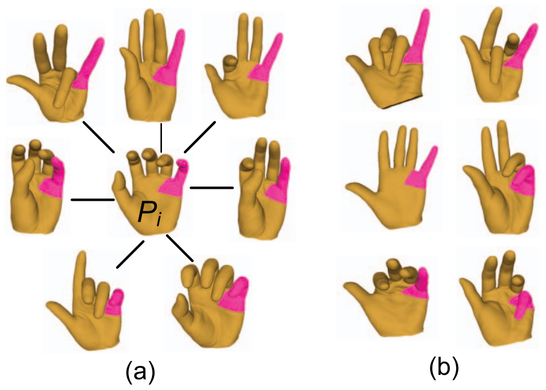


Fig. 5. The similarity graph for the pinky region (a). Distant example poses in (b) are not used in training the local deformation models $\mathbf{d}_j^i(\mathbf{c}_j)$ for bone b_j near pose $\widetilde{P}_i$.

models from the full control point vector $\mathbf{c}$, we now use $\mathbf{c}_j$ to learn a geometrically local deformation model of the form $\mathbf{d}_j(\mathbf{c}_j)$. We set $k_c = 7$ in all our experiments. Fig. 4b visualizes the bones sharing a same set of influence control points in the same color. These maps conform well with anatomical regions such as fingers.

6.1.2 *Local Regression in the Pose Space*

The range of motion of human hands, the most dexterous part of the human body, is large and highly nonlinear. Yet, we only have 14 training examples in total. These examples appear extremely distant with respect to each other inside the huge configuration and deformation space of human hands. A global fit using all the sparse examples as so far described results in models with poor prediction results. Inspired by the success of local model learning and manifold learning methods, such as k-nearest neighbors (kNN), locally weighted regression (LWR) [30], and dimensionality reduction techniques where local proximities are preserved rather than global proximities [31], we advocate building local regression models for each pose from their neighboring poses.

The assumption is that the natural deformation poses we model reside on a low-dimensional nonlinear manifold embedded in the original high-dimensional configuration space. It is from this manifold that our sparse input examples are sampled, and it is within this manifold that we would like our synthesized new poses to reside. We propose to use a regression method Canonical Correlation Analysis (CCA) locally. The local CCA regression relates to the global CCA regression in a way similar to how local linear regressions relate to conventional linear regressions, where local fitting of data points in the vicinity of the input query can greatly improve the prediction accuracy [30].

To build local prediction models in the pose space, we first construct a weighted graph based on the *local similarity* [32]. For a particular bone b_j, each example pose is connected with its k_p nearest neighbors ($k_p = 7$ in all our experiments), measured by the euclidean distance of their influence control point vectors. Each edge is weighted by a heat kernel

$$w(\mathbf{c}_j^i, \mathbf{c}_j^l) = e^{-|\mathbf{c}_j^i - \mathbf{c}_j^l|^2/2\sigma^2}, \tag{2}$$

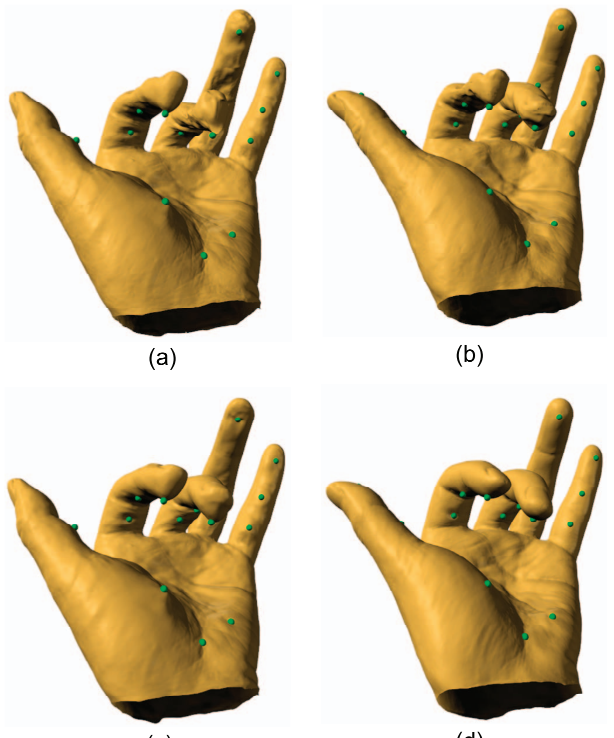


Fig. 6. Large-scale deformations generated from (a) globally trained models as in [12]; (b) locally trained models in the geometry space; (c) locally trained models in the pose space; (d) locally trained models in both spaces.

where $\mathbf{c}_j^i$ represents the control point vector for bone b_j in example pose $\widetilde{P}_i$. Fig. 5 shows such a graph for the bones in the pinky region. To train the local model at pose $\widetilde{P}_i$ for bone b_j, we compute the relative rotation $\mathbf{d}_j^i$ of this bone between $\widetilde{P}_i$ and each of its neighboring pose in the similarity graph. $\mathbf{d}_j^i$ and $\mathbf{c}_j$ are then used to train a deformation predictor $\mathbf{d}_j^i(\mathbf{c}_j)$. These models, though unable to predict deformations for poses far away from $\widetilde{P}_i$, are much more accurate locally than a global predictor. Again, eager readers can preview a comparison between global models and local models in Fig. 6.

6.1.3 *Implementation Details*

We perform linear CCA regressions to learn $\mathbf{d}_j^i(\mathbf{c}_j)$, and use the kernel trick to establish nonlinear dependencies between input variables, similar to [12]. That is, $\mathbf{d}_j^i(\mathbf{c}_j) = \mathbf{M}_j^i(\xi(\mathbf{c}_j))$, where $\mathbf{M}_j^i$ is a linear operator composed of several linear mappings, and $\xi(\mathbf{c}_j)$ is the kernelized vector of the input $\mathbf{c}_j$. We use Gaussian kernels for all our experiments.

Because of the extremely low number of training examples in our case, we only train bone rotation predictors, in the format of quaternions rather than 8D dual-quaternions [33], to reduce overfitting. Bone translations are solved afterwards by the Poisson translation solver [12], which minimizes a weighted sum of the edge prediction differences E_e and the control point positional errors E_c of the form $(\sum E_e + \beta \sum E_c)$. β is a weighting factor to control how exactly the surface should follow the control points. We use different weighting schemes for keyframe animation and performance-driven animation, which will be further explained in the results section. The Poisson

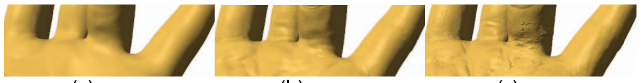


Fig. 7. Different strategies to model details. (a) The predicted smooth base mesh. (b) The base mesh plus details trained from control points. (c) The base mesh plus details trained from predicted bone transformations. Note the artifacts caused by error propagation.

minimization equates to a linear least-squares problem whose solution can be written as $\mathbf{t} = \mathbf{P}\mathbf{f}$, where $\mathbf{t}$ is the vector of bone translations, $\mathbf{P}$ is a precomputed pseudoinverse matrix, and $\mathbf{f}$ contains both the predicted bone rotations $\mathbf{d}$ and the control point positions $\mathbf{c}$. From $\mathbf{t}$ and $\mathbf{d}$, we can then easily compute the bone transformation matrices $\mathbf{T}$. We refer the interested readers to [12] for the remaining details.

6.2 Fine-Scale Displacement Modeling

What we have done in the previous section is essentially linear blend skinning, even though the bone transformations are predicted from control point configurations that are local in both the geometry and the pose space. It is well known that linear blend skinning is ineffective in modeling high-frequency deformation details. Using more abstract bones would improve the data fitting quality to some extent, but will eventually run into the problem of overfitting.

Therefore, we train another layer of CCA-based regression models to account for the differences between P_i and $\widetilde{P}_i$. We use the high-resolution displacement maps $\mathbf{h}_i$ and their corresponding control points to train a displacement prediction model $\mathbf{h}(\mathbf{c})$ for every vertex. We have also tried to use the predicted bone transformations $\mathbf{T}$ instead of the original control points $\mathbf{c}$ as input for the regression process. Our experiments show an inferior synthesis quality using such a cascaded scheme, as indicated by Fig. 7c, because the bone prediction errors are transferred to and amplified by the displacement predictor, leading to noticeable visual artifacts. Both Figs. 7b and 7c show results after the Poisson reconstruction.

We use 1D displacements along vertex normals rather than 3D displacement vectors for detail modeling. Because the number of our model parameters relates to the product of the input and output dimensionality, learning regression models from sparse examples that predict 3D displacement vectors suffers from overfitting. Predicting a scalar value per vertex alleviates this problem, and is also more robust with respect to inaccuracies and noise in vertex correspondences and displacement maps. In addition, storage requirement is greatly reduced because one displacement predictor is trained for every vertex of the high-resolution model.

When learning per-vertex displacement prediction models, we simply run CCA-based regression using all the control points and training poses. The resulting prediction models are capable of producing satisfactory visual results without incorrect interferences between distant regions. This demonstrates that in the context of learning high-frequency displacement models, CCA can recognize and extract correct correlations and dependencies from the sparse training data without the assistance of any scheme that confines the model learning to local regions in the geometry space.

7 Deformation Synthesis

At runtime, the control points can either be manipulated by a user or driven by motion captured markers, and a new deformed model can be synthesized as follows: given a new control point vector $\mathbf{c}$, for each bone b_j, we select its precomputed influence control points $\mathbf{c}_j$ and look for its k_p nearest neighbors among all the example poses. The chosen poses each have a deformation model trained locally for bone b_j that can independently predict a bone rotation $\mathbf{d}_j^l(\mathbf{c}_j), l = 1, \dots, k_p$. The final bone rotation is computed as a weighted average of the individual predictions

$$\mathbf{d}_j(\mathbf{c}_j) = \sum_{l=1}^{k_p} w(\mathbf{c}_j, \mathbf{c}_j^l)\mathbf{d}_j^l(\mathbf{c}_j) \Bigg/ \sum_{l=1}^{k_p} w(\mathbf{c}_j, \mathbf{c}_j^l). \tag{3}$$

The weights $w(\mathbf{c}_j, \mathbf{c}_j^l)$ are calculated by the same heat kernel as in (2). Note that bone rotations predicted by a local model $\mathbf{d}_j^l$ are defined with respect to the specific example pose $\widetilde{P}_l$. It is necessary to first transform them to the reference pose $\widetilde{P}_0$ before blending them. For the sake of notation simplicity, we omit the coordinate transformation here and directly write $\mathbf{d}_j^l(\mathbf{c}_j)$ in the above formula. Now, from the predicted composite bone rotations, bone translations $\mathbf{t}_j(\mathbf{c}_j)$ can be computed via the Poisson solver as described in Section 6.1.3. Bone transformations $\mathbf{T}_j(\mathbf{c}_j)$ are then computed from $\mathbf{d}_j(\mathbf{c}_j)$ and $\mathbf{t}_j(\mathbf{c}_j)$. Finally, a vertex position is calculated according to (1).

The resulting mesh above is a low-resolution smooth mesh predicted from $\widetilde{P}_i$ and thus only reproduces large-scale deformations. To add details, we first subdivide the low-resolution mesh in the same way as in the subdivision procedure for detail extraction, and we also recompute the normal $\mathbf{n}(v)$ for each vertex v. Then, we use the trained displacement model $\mathbf{h}(\mathbf{c})$ to generate a high-resolution displacement map that can be added to the subdivided mesh along $\mathbf{n}(v)$. To put it into one single mathematical form, a vertex position v in a synthesized model with details can be estimated as follows:

$$v(\mathbf{c}) = \sum_{j \in B(v)} w_j \mathbf{T}_j(\mathbf{c}_j) v^r + \mathbf{h}(\mathbf{c})\mathbf{n}(v), \tag{4}$$

where $B(v)$ stands for the influence bone set for vertex v.

8 Experimental Results

We have implemented our deformation system in C++ on a 2.83 GHz Intel Quad core machine with 8 GB of RAM. The deformation results shown in the paper and the accompanying video, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TVCG.2012.88, are rendered by an OpenGL renderer with Phong shading.

Performance. Table 1 lists the performance statistics for each testing data set. For the hand models, we also tested two sets of control point configurations, one on the palm side, and another on the back side of the hands. The timing of each stage of the deformation system is given. The model registration and training steps are done offline, but still within a reasonable time frame of tens of minutes.

TABLE 1
Performance Statistics

Model	#Cpt	#Bone	DT(min)	Training(min) bone/disp.	Synthesis(sec) bone/disp.
Hand-I-K	17	1355	15	3/19	0.6/3.4
Hand-I-M	15	1355	15	3/19	0.6/3.4
Hand-II-K	19	1239	15	3/19	0.6/3.5
Hand-II-M	15	1239	15	3/19	0.6/3.5
Face-I	26	505	20	1/6	0.2/1.7
Face-II	8	247	NA	0.5/NA	0.1/NA
Dinosaur	24	463	NA	1/11	0.2/2.5

"#Cpt": number of control points; "#Bone": number of abstract bones; "DT": time spent on deformation transfer; "Training": training time for the bone deformation models and the displacement models, respectively; "Synthesis": synthesis time for bone transformations and vertex displacements. Hand-*-K: deformation modeling with the palm-side control points for keyframe animation; Hand-*-M: deformation modeling with the back-side motion captured markers for performance-driven animation.

Currently, the large-scale deformation synthesis is interactive, and the detail synthesis is several-fold slower.

Validation and comparison. Fig. 8 compares a mesh with details predicted from our pose-driven displacement models to another mesh with a static displacement map extracted from the rest pose. The same bone transformation models are used for generating the deformed base mesh. This comparison shows that a displacement map extracted from one pose cannot reproduce the deformed mesh details in other poses. It also demonstrates that our displacement predictor captures well the variation of geometric details from the input examples.

Fig. 6 demonstrates the effectiveness of our local modeling method as compared to the global regression approach of [12]. Fig. 6a exhibits large distortions, while locally learned models in both the geometry space and the pose space effectively eliminate such artifacts as shown in Fig. 6d. We also compared against learning local models in either the geometry or the pose space alone. Figs. 6b and 6c show that local models in just one of the spaces are helpful in reducing the prediction errors, yet neither alone can completely eliminate all deformation artifacts.

Fig. 9 compares our results with those of MESHIK [10] using the same control point input. Our results in Fig. 9a predict natural novel poses with clean fine-scale details, while the overall pose and details synthesized by MESHIK contain visible artifacts. The closeup in Fig. 9b further shows that details from multiple training poses incorrectly mix together, giving rise to unnecessarily cluttered wrinkles.

We also performed leave-one-out cross validations for each example pose of each model, and report the average

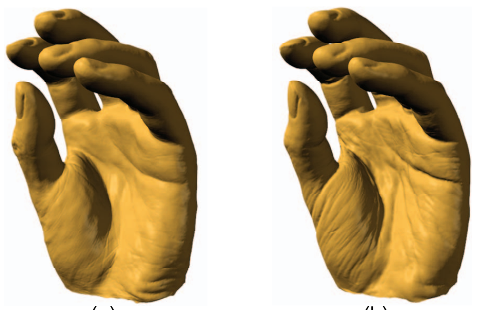


Fig. 8. Deformation results with a static displacement map (a) and our pose-dependent displacements (b).

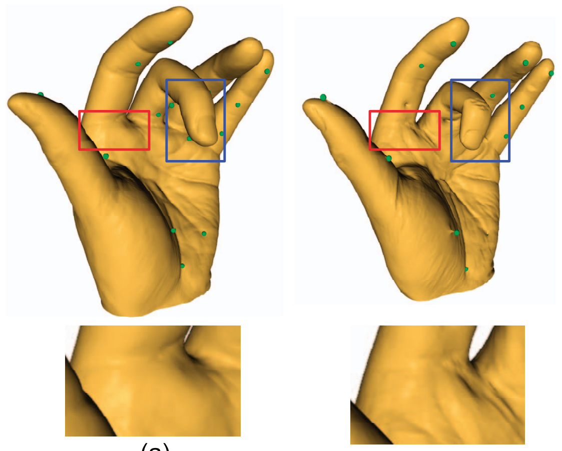


Fig. 9. Comparison with MESHIK. (a) Results from our method. (b) Results from MESHIK under the same control point configuration. The blue boxes point out one area that is problematic for MESHIK to synthesize correct large-scale deformations. The bottom row shows the closeup views of the area enclosed by the red boxes in the upper row. MESHIK produces false wrinkles in this case.

RMS errors in Table 2. The largest dimension of the bounding box of each model is scaled to unit length. In Fig. 10, we show visual comparison of two cross validations using the hand-I model. Our method not only generates consistently lower prediction errors than the method of [12], but also produces significantly better visual results, in terms of both the large-scale deformations and the fine-scale details. However, note that some of the veins appear smoother in the bottom of Fig. 10c. Our deformation framework is fundamentally data-driven, and cannot synthesize features not present in the remaining training examples.

Keyframe animation. We developed a Graphical User Interface (GUI) for editing the position of control points. To provide fast visual feedback, we only compute and render the bone-based large-scale deformations during interactive editing. Once the user is satisfied with the large-scale deformations, displacement predictions are added to refine the results. Figs. 13a and 13c show some representative frames of keyframe animation sequences of Hand-I and Hand-II, respectively. Despite the sparse examples used in the training phase, our deformation models produce smooth large-scale deformations of the whole hand as well as plausible deformations of detailed wrinkles for various gestures.

For the dinosaur model, the artist generated a motion sequence from the 13 low-resolution reference meshes using Maya skeletal animation. We then extracted the trajectories of control points and fed them into our system to generate

TABLE 2
Comparison of Cross Validations between Our Method and [12]

Model	#Example	#VertLow	#VertHigh	RMS Error	
				[12]	Ours
Hand-I	14	49K	195K	0.0435	0.0315
Hand-II	14	49K	196K	0.0332	0.0240
Face-I	21	17K	70K	0.0294	0.0281
Face-II	6	24K	NA	0.0291	0.0279
Dinosaur	13	34K	136K	0.0338	0.0270

The RMS errors are the averaged results of the leave-one-out validation for each example pose. "#VertLow": vertex number of the low-resolution meshes $\widetilde{P}_i$;"#VertHigh": vertex number of the high-resolution meshes P_i.

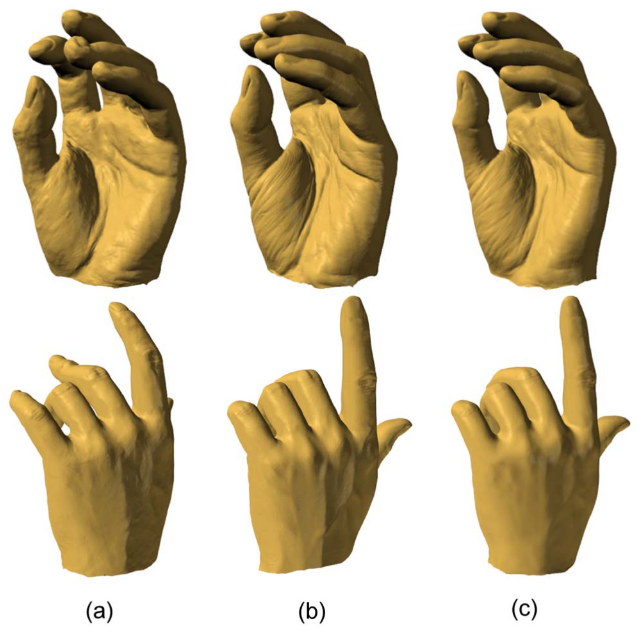


Fig. 10. Two leave-one-out cross validations with (a) the method of [12] and (c) our method. The ground truth models, i.e., the models removed from the training examples, are shown in column (b).

animated dinosaurs with deformation details, of which some representative frames are shown in Fig. 14e. Note that geometric details designed by the artist on the 13 example meshes have been well learned and then synthesized for new poses of the animation sequence. The hand-animated low-resolution mesh sequence without details is only used to extract control point trajectories, and is not used in the synthesis of Figs. 14d or 14e.

Performance-driven animation. We also tested our system with motion capture data. We placed 18 motion capture markers on the back side of the hands, of which 15 were used to train deformation models. The remaining three markers plus one of the 15 markers used for deformation training were used for control point alignment. The 3D marker positions were then captured at 120 Hz using a Vicon optical motion capture system, and then downsampled to 30 Hz to drive our deformation models. Some selected frames are shown in Figs. 13b and 13d, although the

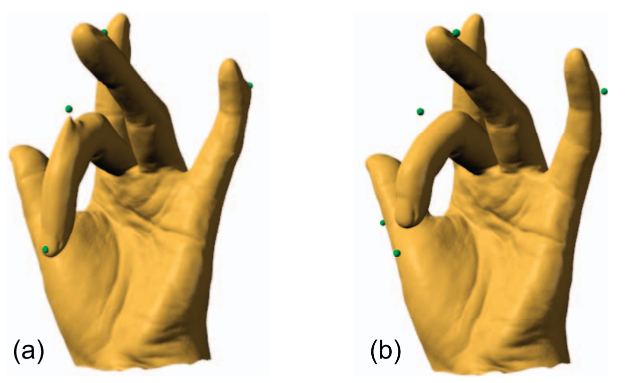


Fig. 11. Comparison of different weighting scheme for the control point constraint in the Poisson solver for noisy control points: (a) $\beta = 3.0$, (b) $\beta = 0.3$.

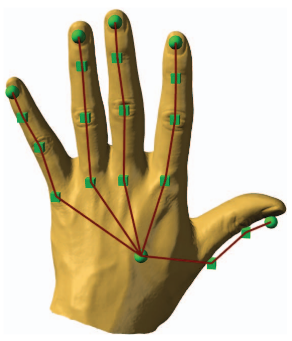


Fig. 12. The manually extracted skeleton on the reference pose of the hand-I model. End-effectors and internal joints are represented by spheres and boxes, respectively. For deformation control, only the end-effectors are exposed as control points.

results are best seen in the accompanying video, available online. Our framework essentially provides a performance-driven animation system that can produce high-quality mesh animations with fine-scale details, using just a handful of captured marker trajectories. We showed our results to two artists, and some of their comments include: “The dynamic wrinkles look very realistic and consistent. Such quality is very hard to achieve manually”; “It would take us days to make one of such animation sequences using currently available commercial software packages.”

For the face-I model, we used a subset of 26 markers among the full set of 111 markers available from the Matt face motion capture sequence as the control points, and fed their trajectories into our system to synthesize the mesh deformation sequence, of which some representative frames are shown in Fig. 14b. The synthesized quality is comparable to that of [14], but we use fewer markers, which can greatly reduce the workload in deformation editing and motion capture. For illustrating the effect of our two-layer deformation models, we also show the results of large-scale deformation synthesis without details in Fig. 14a.

For the face-II model, we extracted eight control point trajectories from the original face mesh sequence reconstructed from multiview videos, and fed them into our system as virtual markers to regenerate the deformation sequence, of which some representative frames are shown in Fig. 14c. Note that geometric details were not present in the original examples, so we only ran the large-scale deformation modeling and synthesis component of our framework for this experiment. The control point trajectories were extracted from the original full sequence but we did not use any mesh other than the six chosen examples, although we have access to the full mesh sequence. The sequence shown in Fig. 14c and the accompanying video, available online, is resynthesized from our framework.

There is one key difference in working with motion captured marker trajectories instead of manually edited control point positions. In an interactive editing setup, users generally want the pose to exactly follow their control points, and would otherwise feel frustrated. Yet, the motion captured marker positions, if taken literally, can produce large deformation artifacts, such as shown in Fig. 11a. There are many error sources degrading animations driven by captured markers. First, we cannot guarantee that the markers on the live hand and the control points on the hand

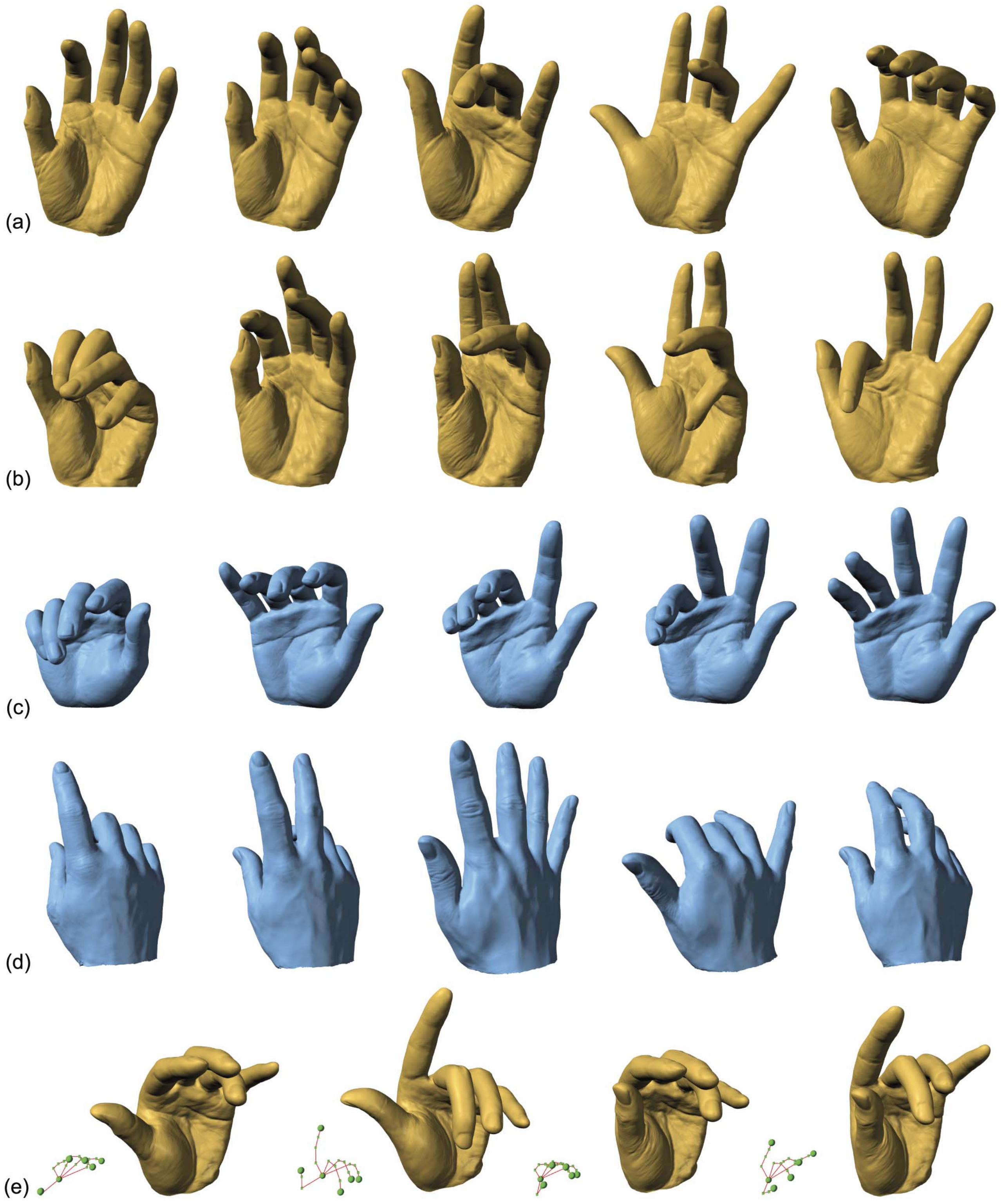


Fig. 13. Results of keyframe animation (a) and performance-driven animation (b) for the hand-I model; keyframe animation (c) and performance-driven animation (d) for the hand-II model; and end-effector-driven deformation (e) of the hand-I model.

mesh are located at exactly the same spots. Second, the markers should be aligned with the reference model to eliminate rigid transformations between them. Currently, we use four markers placed at the back of the hand to estimate these transformations, but anatomically the back of the hand consists of many small bones and is not literally a rigid body. Third, there are noise and even missing markers caused by occlusions in the captured data. Lastly, to make our performance-driven animation system practical, we cannot assume that we can always motion capture the same hand from which we scanned and learned our deformation models. Indeed, both our human hand models were unable to participate in our motion capture sessions, and we had to capture the marker motions from a third subject to drive the

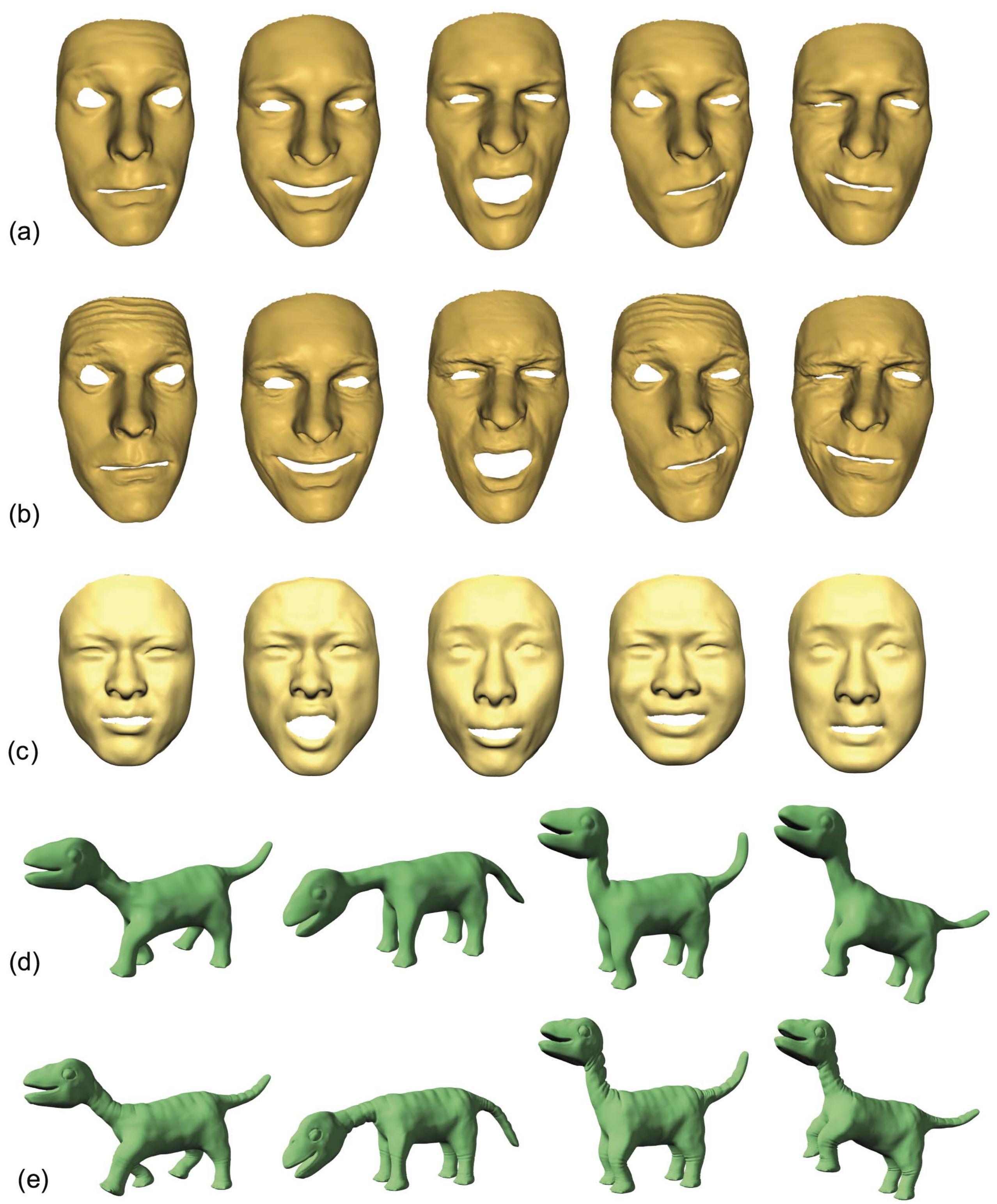


Fig. 14. Results of performance-driven animation for the face-I model with only large-scale deformations (a) and with details (b); performance-driven animation for the face-II model (c); and keyframe animation for the dinosaur model with only large-scale deformations (d) and with details (e).

virtual hand-I and hand-II models. Due to all the error factors above, for performance-driven animation we use a lower weight for the control point positional constraint term in the Poisson translation solver. Fig. 11 illustrates this effect.

End-effector-driven animation. Keyframe editing or motion capturing $15 \sim 20$ control points may still be overwhelming for novice users. We can further bring down the required number of control points by utilizing Inverse Kinematics for models that have a clear skeletal structure. We first manually label a set of points on the surface of the reference mesh to define a skeleton, as shown in Fig. 12. Note that two joints colocate at the root position, to allow the base of the thumb and the palm to move independently. If only one joint is used, the base of the thumb and the palm will be treated as a single rigid body by the IK solver, which reduces the range of motion of the thumb. The six green spheres at the finger tips and the center of the palm are the control points exposed to the user. The 14 green boxes roughly corresponding to joint locations for each finger are control points hidden from the user. At runtime, users only need to edit or capture the six end-effector points. We then use a numerical IK method called Damped Least Squares [34] to compute the positions of the other 14 internal control points. Then, the 20 control points together will drive the deformation models prelearned from all the control points and the example meshes.

Note that our manually extracted skeleton does not necessarily align with the anatomical skeleton of human hands. In addition, we use only ball-and-socket 3DoF joints to model the finger joints in our IK solver. Despite these approximations and simplifications, the results generated are comparable to those synthesized with a full set of explicitly specified control points, as shown in Fig. 13e for the hand-I model. Building regression models directly from an anatomically correct skeleton and the example meshes remains a future work, but should be straightforward.

9 Conclusions and Discussion

Generating user-controllable mesh animation with rich details from sparse examples is a challenging open problem. Our main contribution is a robust framework that can produce fine-scale details as well as large-scale deformations by learning from extremely sparse training data. CCA-based regressions are used to model deformations of both layers, which not only makes the framework simple and clean, but also allows end-users to directly manipulate control points for interactive mesh animation. We demonstrate the effectiveness and robustness of our method using both scanned and manually constructed example models, with either hand-edited or motion-captured control point trajectories.

Our method by far outperforms some of the global methods we have tested. Local fitting in both the geometry space and the pose space is the key to our success, which constrains models within the manifold of natural poses and effectively decouples independent object parts. Note, however, when there are more example poses available, a global method can usually generate better results than what is shown in Fig. 6a. For example, Feng et al. [12] report that using 80 example poses, a pair of pants, which can be thought as two fingers, can be animated with acceptable quality.

The input training examples should be carefully designed to span the configuration space as much as possible, and extrapolation should generally be avoided, similar to all data-driven methods. In extreme cases where the user insists on dragging a control point outside of the spanned subspace, such as to make a closed fist which our current hand acquisition method cannot cast, the synthesized deformation will stop following the control points, unless we make the control point error term in Section 6.1.3 a hard constraint.

There are several limitations of the proposed method that deserve future investigation. First, we need to manually specify feature correspondences to initialize the deformation transfer algorithm for model registration. To ensure good registration for both large-scale and fine-scale features, we used 120 feature correspondences for the hand models. We have experimented with adapting an image-space optical flow algorithm to 3D surfaces to register example models automatically. Our initial results indicate that registering features for highly deformable models with rich details is a challenging problem. On a side note, 178 feature points are used for face registration in [18]. The registration method of [21] requires dense scans and would fail for cases where only sparse scans are available.

Our method is general in the sense that it does not require rigged example meshes. This enables direct utilization of scanned data. However, to better integrate with traditional skeletal animation tools and utilize legacy animations, it would be useful to drive deformable models using anatomically based bone transformations for models that have inherent skeleton structures such as hands. Our end-effector-driven animation illustrates the feasibility of skeleton-driven deformation to a certain extent.

Another topic for future research concerns dynamic deformations. In this paper, we primarily focus on pose-driven static deformations, which means there is a unique deformation associated with each pose. It would be interesting to investigate mechanisms to carry over deformation dynamics from previous instants of time.

Acknowledgments

This work was partially supported by National University of Singapore Academic Research Fund (R-252-000-429-133), National Natural Science Foundation of China (61073078), and US National Science Foundation (NSF) (IIS 09-14631). The authors would like to thank Hoang Thai Duong for correcting and rendering some of the results in this paper. They also wish to thank the anonymous reviewers for their detailed comments and suggestions.

References

[1] H. Huang, L. Zhao, K. Yin, Y. Qi, Y. Yu, and X. Tong, "Controllable Hand Deformation from Sparse Examples with Rich Details," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA '11),* pp. 73-82, 2011.

[2] J.P. Lewis, M. Cordner, and N. Fong, "Pose Space Deformation: A Unified Approach to Shape Interpolation and Skeleton-Driven Deformation," *Proc. ACM SIGGRAPH '00,* pp. 165-172, 2000.

[3] P.-P.J. Sloan, C.F. Rose, and M.F. Cohen, "Shape by Example," *Proc. Symp. Interactive 3D Graphics (I3D '01),* pp. 135-143, 2001.

[4] A. Mohr and M. Gleicher, "Building Efficient, Accurate Character Skins from Examples," *ACM Trans. Graphics,* vol. 22, no. 3, pp. 562-568, 2003.

[5] O. Weber, O. Sorkine, Y. Lipman, and C. Gotsman, "Context-Aware Skeletal Shape Deformation," *Computer Graphics Forum,* vol. 26, no. 3, pp. 265-274, 2007.

[6] R.Y. Wang, K. Pulli, and J. Popović, "Real-Time Enveloping with Rotational Regression," *ACM Trans. Graphics,* vol. 26, no. 3, article 73, 2007.

[7] P.G. Kry, D.L. James, and D.K. Pai, "EigenSkin: Real Time Large Deformation Character Skinning in Hardware," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA '02),* pp. 153-159, 2002.

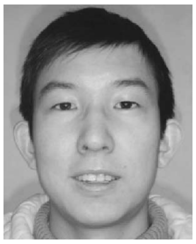
[8] T. Kurihara and N. Miyata, "Modeling Deformable Human Hands from Medical Images," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA '04),* pp. 355-363, 2004.

[9] T.-Y. Kim and E. Vendrovsky, "DrivenShape: A Data-Driven Approach for Shape Deformation," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA '08),* pp. 49-55, 2008.

[10] R.W. Sumner, M. Zwicker, C. Gotsman, and J. Popović, "Mesh-Based Inverse Kinematics," *ACM Trans. Graphics,* vol. 24, no. 3, pp. 488-495, 2005.
[11] K.G. Der, R.W. Sumner, and J. Popović, "Inverse Kinematics for Reduced Deformable Models," *ACM Trans. Graphics,* vol. 25, no. 3, pp. 1174-1179, 2006.
[12] W.-W. Feng, B.-U. Kim, and Y. Yu, "Real-Time Data Driven Deformation Using Kernel Canonical Correlation Analysis," *ACM Trans. Graphics,* vol. 27, no. 3, article 91, 2008.
[13] M. Lau, J. Chai, Y.-Q. Xu, and H. Shum, "Face Poser: Interactive Modeling of 3D Facial Expressions Using Model Priors," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA '07),* pp. 161-170, 2007.
[14] H. Huang, J. Chai, X. Tong, and H.-T. Wu, "Leveraging Motion Capture and 3D Scanning for High-Fidelity Facial Performance Acquisition," *ACM Trans. Graphics,* vol. 30, no. 4, article 74, 2011.
[15] J.R. Tena, F.D. la Torre, and I. Matthews, "Interactive Region-Based Linear 3D Face Models," *ACM Trans. Graphics,* vol. 30, no. 4, article 76, 2011.
[16] B. Bickel, M. Botsch, R. Angst, W. Matusik, M. Otaduy, H. Pfister, and M. Gross, "Multi-Scale Capture of Facial Geometry and Motion," *ACM Trans. Graphics,* vol. 26, no. 3, article 33, 2007.
[17] B. Bickel, M. Lang, M. Botsch, M.A. Otaduy, and M. Gross, "Pose-Space Animation and Transfer of Facial Details," *Proc. ACM SIGGRAPH/Eurographics Symp. Computer Animation (SCA '08),* pp. 57-66, 2008.
[18] W.-C. Ma, A. Jones, J.-Y. Chiang, T. Hawkins, S. Frederiksen, P. Peers, M. Vukovic, M. Ouhyoung, and P. Debevec, "Facial Performance Synthesis Using Deformation-Driven Polynomial Displacement Maps," *ACM Trans. Graphics,* vol. 27, no. 5, article 121, 2008.
[19] E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun, "Performance Capture from Sparse Multi-View Video," *ACM Trans. Graphics,* vol. 27, no. 3, article 98, 2008.
[20] D. Vlasic, I. Baran, W. Matusik, and J. Popović, "Articulated Mesh Animation from Multi-View Silhouettes," *ACM Trans. Graphics,* vol. 27, no. 3, article 97, 2008.
[21] H. Li, B. Adams, L.J. Guibas, and M. Pauly, "Robust Single-View Geometry and Motion Reconstruction," *ACM Trans. Graphics,* vol. 28, no. 5, article 175, 2009.
[22] D. Bradley, W. Heidrich, T. Popa, and A. Sheffer, "High Resolution Passive Facial Performance Capture," *ACM Trans. Graphics,* vol. 29, no. 4, article 41, 2010.
[23] S. Sueda, A. Kaufman, and D.K. Pai, "Musculotendon Simulation for Hand Animation," *ACM Trans. Graphics,* vol. 27, no. 3, article 83, 2008.
[24] L. Zhang, N. Snavely, B. Curless, and S. Seitz, "Spacetime Faces: High Resolution Capture for Modeling and Animation," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 546-556, 2004.
[25] A. Golovinskiy, W. Matusik, H. Pfister, S. Rusinkiewicz, and T. Funkhouser, "A Statistical Model for Synthesis of Detailed Facial Geometry," *Proc. ACM SIGGRAPH '06,* pp. 1025-1034, 2006.
[26] S. Park and J. Hodgins, "Capturing and Animating Skin Deformation in Human Motion," *ACM Trans. Graphics,* vol. 25, no. 3, pp. 881-889, 2006.
[27] G. Turk, "Re-Tiling Polygonal Surfaces," *Proc. ACM SIGGRAPH '92,* pp. 55-64, 1992.
[28] R.W. Sumner and J. Popović, "Deformation Transfer for Triangle Meshes," *ACM Trans. Graphics,* vol. 23, no. 3, pp. 399-405, 2004.
[29] D.L. James and C.D. Twigg, "Skinning Mesh Animations," *Proc. ACM SIGGRAPH '05,* pp. 399-407, 2005.
[30] W.S. Cleveland and S.J. Devlin, "Locally Weighted Regression: An Approach to Regression Analysis by Local Fitting," *J. Am. Statistical Assoc.,* vol. 83, no. 403, pp. 596-610, 1988.
[31] L.K. Saul and S.T. Roweis, "Think Globally, Fit Locally: Unsupervised Learning of Low Dimensional Manifolds," *J. Machine Learning Research,* vol. 4, pp. 119-155, 2003.
[32] M. Belkin and P. Niyogi, "Laplacian Eigenmaps and Spectral Techniques for Embedding and Clustering," *Proc. Advances in Neural Information Processing Systems (NIPS),* pp. 585-591, 2001.
[33] L. Kavan, S. Collins, J. Žára, and C. O'Sullivan, "Skinning with Dual Quaternions," *Proc. Symp. Interactive 3D Graphics and Games (I3D '07),* pp. 39-46, 2007.
[34] S.R. Buss and J.-S. Kim, "Selectively Damped Least Squares for Inverse Kinematics," *J. Graphics Tools,* vol. 10, no. 3, pp. 37-49, 2005.

Haoda Huang received the MS degree from the Institute of Software of the Chinese Academy of Sciences in 2007. He has worked as an associate researcher at Microsoft Research Asia from 2007 to 2011. His research interests include data-driven computer animation, facial performance capturing, image editing and image retrieval.

KangKang Yin received the PhD degree from the University of British Columbia in 2007. She has worked as an associate researcher at Microsoft Research Asia from 2008 to 2010. She is currently an assistant professor in the Department of Computer Science at the National University of Singapore. Her research interests include computer animation and geometry processing. She is a member of the ACM.



Ling Zhao is currently working toward the PhD degree in the State Key Laboratory of Virtual Reality Technology and Systems at Beihang University, China. His research interests include geometry modeling, animation and virtual reality.

Yue Qi received the BS and PhD degrees in system engineering from National University of Defense Technology in 1991 and 2001, respectively, and the MS degree in computer science from the University of Science and Technology of China in 1995. He is currently a professor in the State Key Laboratory of Virtual Reality Technology and Systems at Beihang University, China. His research interests include virtual reality and computer graphics, with an emphasis on geometry and appearance modeling. He is a member of the IEEE.

Yizhou Yu is an associate professor in the Department of Computer Science at University of Illinois at Urbana-Champaign and University of Hong Kong. He is the recipient of 2002 US National Science Foundation (NSF) CAREER Award and the Best Paper Award at 2005 and 2011 ACM SIGGRAPH/Eurographics Symposium on Computer Animation. He is on the editorial board of *Computer Graphics Forum, the Visual Compute*r, and the *International Journal of Software and Informatics*. He is a senior member of the IEEE.

Xin Tong received the PhD degree from Tsinghua University and then joined Microsoft Research Asia in 1999. He is a senior researcher in Microsoft Research Asia. His research interests include appearance modeling and rendering, texture synthesis, and performance capturing. He is a member of the IEEE and the ACM.