Data-Driven Weight Optimization for Real-Time Mesh Deformation

Yu-Jie Yuan[a,b], Yu-Kun Lai[c], Tong Wu[a], Shihong Xia[a], Lin Gao[a,*]

[a] *Beijing Key Laboratory of Mobile Computing and Pervasive Device, Institute of Computing Technology, Chinese Academy of Sciences*
[b] *University of Chinese Academy of Sciences*
[c] *School of Computer Science and Informatics, Cardiff University, UK*

Abstract

3D model deformation has been an active research topic in geometric processing. Due to its efficiency, linear blend skinning (LBS) and its follow-up methods are widely used in practical applications as an efficient method for deforming vector images, geometric models and animated characters. LBS needs to determine the control handles and specify their influence weights, which requires expertise and is time-consuming. Further studies have proposed a method for efficiently calculating bounded biharmonic weights of given control handles which reduces user effort and produces smooth deformation results. The algorithm defines a high-order shape-aware smoothness function which tends to produce smooth deformation results, but fails to generate locally rigid deformations. To address this, we propose a novel data-driven approach to producing improved weights for handles that makes full use of available 3D model data by optimizing an energy consisting of data-driven, rigidity and sparsity terms, while maintaining its advantage of allowing handles of various forms. We further devise an efficient iterative optimization scheme. Through contrast experiments, it clearly shows that linear blend skinning based on our optimized weights better reflects the deformation characteristics of the model, leading to more accurate deformation results, outperforming existing methods. The method also retains real-time performance even with a large number of deformation examples. Our ablation experiments also show that each energy term is essential.

*Corresponding Author: gaolin@ict.ac.cn (Lin Gao)

Keywords: Real-time mesh deformation; Biharmonic weights; Data-driven; As-rigid-as-possible

1. Introduction

With the rapid development of virtual reality and 3D printing technologies, the demand for 3D models in industry is ever increasing. Traditional methods for geometric modeling require users to have the expertise in using specialized software such as Autodesk Maya. However, this increases the cost of acquiring 3D models and limits the users who can effectively manipulate 3D models to suit their needs. This is particularly critical for virtual reality systems where real-time performance is essential. Meanwhile, 3D models are getting increasingly available as professional artists continue to create models or obtain models by 3D scanning. It is therefore useful to generate new models by reusing existing ones. Because of this, 3D model deformation plays a vital role in games, animations, films and virtual reality. Rapid deformation methods which produce realistic results are highly demanded.

Quite recently, Jacobson et al. [1] proposed bounded biharmonic weights, which when coupled with the linear blend skinning (LBS) algorithm [2], are able to deform 3D models without the need of manual weight specification. The method derives weights by optimizing a high-order shape-aware energy without exploiting example shapes, and so cannot capture full deformation behavior of the object. The weights sometimes result in objects that appear too soft, especially for those objects that are locally rigid.

In this paper, we propose a novel weight optimization method that automatically produces suitable weights for LBS handles by exploiting example deformations. To better preserve local shapes, our method utilizes an as-rigid-as-possible (ARAP) energy term [3] to optimize the weights. We further propose to incorporate a data-driven energy term that optimizes the weights according to the example deformed shapes in the dataset. Finally, to improve deformation results for near piecewise rigid deformation, a sparse regularization term is introduced to limit the control range of the weights. The final energy involves the data-driven, sparse regularization and ARAP energy terms.

We further develop an efficient optimization strategy with the weights initialized using biharmonic weights [1]. As we will show later by extensive experiments, our method produces weights that can better deform the target objects than state-of-the-art methods, while maintaining the benefits of LBS-based deformation, i.e. allowing handles of various forms (e.g. points, bones) and real-time performance even with a large number of example shapes.

2. Related Work

2.1. Geometry-based Mesh Deformation

Mesh deformation has been an active topic in computer graphics research. Ideally, visually reasonable deformation is performed on the mesh with the guidance of user input, often in the form of transformation or rotation of the handle points. For a 3D model in hinge structure like a human body, skeleton-based methods can be used for deformation. The deformed skeletons are used to drive mesh deformation. However, this does not apply to general 3D models. For a general 3D model, a typical method is based on local differential coordinates, which can be used to maintain the geometrical details of the model and reconstruct the deformed 3D model under the boundary conditions specified by the user. Typical work includes deformation based on Laplacian coordinates [4], the gradient domain method based on Poisson equation [5] and method based on dual Laplacian coordinates [6]. In order to maintain the consistency of the model volume during the deformation, methods [7, 8] introduce optimization terms that keep the volume constant during the reconstruction process using Laplacian coordinates. This type of differential coordinate based methods requires the user to specify the spatial coordinates and rotation transformation of the control vertex regions. To simplify user input, Sorkine et al. [3] estimate the differential coordinates of the rigid transformation iteratively in a framework that optimizes an as-rigid-as-possible (ARAP) energy. With this method, the user only needs to specify the deformed coordinates of the control points. The work [9] extends the original ARAP formula by introducing the anisotropy directly into the deformation energy. However, given the same input which has a large deformation space, it is difficult for non-data-driven mesh deformation methods with

no prior knowledge to reasonably distribute the distortion during model deformation, and the deformation results may not follow the behavior of the target objects.

Another approach to driving mesh deformation is through assigning control weights for mesh regions w.r.t. each handle, and using methods such as linear blend skinning (LBS) [2] to propagate handle movement to mesh vertices. This method is very efficient, and therefore suitable for real-time deformation. It supports a variety of different handle forms, such as control points, bones, etc. However, it requires not only specifying handles, but also their weights on vertices, which is tedious. To address this, Jacobson et al. [1] propose an automatic approach to determining weights by optimizing a high-order shape-aware energy, namely bounded biharmonic. The method is essentially heuristic and may produce unnatural deformation results especially for piecewise rigid objects as they appear too soft. Later work [10] allows the user to specify some constraints with the remaining constraints automatically derived by minimizing an ARAP energy. To cope with shapes that have complex deformation behavior, either a large number of constraints are required, or the method may not produce the desired results. Recent work [11] also combines the ARAP energy with the LBS algorithm to find suitable skinning weights. However, their method is only suited for skeleton handles. Moreover, LBS suffers from joint collapse. Recently, Bai et al. [12] proposed a volumetric skinning method using a set of meta-balls to solve this problem, while our method uses data-driven optimization to avoid this kind of artifacts.

2.2. Data-driven Mesh Deformation

For the deformation of 3D models, the aim is to obtain realistic deformation results. As the deformation behavior of objects can be complicated, learning this from the model dataset can help obtain desired results more effectively. Sumner et al. [13] first proposed a method based on global principal component analysis (PCA), which can analyze the model dataset and extract the main components of the model deformation. However, the user manipulation is often local when editing the model, but the use of the global principal components causes other unedited places to also be deformed, which violates the user's editing intention. To solve this problem, Neumann et al. [14] proposed a method that uses local PCA to exact principal components of local defor-

mations for models in the dataset, but the work is based on the Euclidean coordinates so cannot handle large rotations. Huang et al. [15] replaced the coordinate representation with the deformation gradient representation which better handles large rotations. Frohlich et al. [16] use rotation-invariant quantities for data-driven mesh editing, but they use rotation invariants such as edge lengths and dihedral angles of the mesh, so their work cannot handle extrapolation and requires solving a large-scale linear system of equations with varying coefficients. Gao et al. [17] proposed a data-driven model editing method based on rotational invariants, but this method still uses the global principal components, so it is difficult to perform local editing. Moreover, the method uses numerical methods to calculate derivatives required for optimizing the deformation energy, which is inefficient, especially when there is a large number of example shapes. Tan et al. [18] proposed a graph convolutional neural network to extract localized deformation components. By utilizing the as-consistent-as-possible representation [19], the method handles large rotations robustly. However, such methods still cannot cope with different types of handles. In skin deformation, Murai et al. [20] combine simulation-based and data-driven approaches, where simulation helps obtain realistic results for a wide variety of motions, and the new data can be used to adapt the model to different body types. Our method also exploits example shapes, and use them to help derive optimized weights for LBS. Le et al. [21] proposed a data-driven method to automatically generate LBS weights based on the skeleton. Their method is restricted to skeleton handles, whereas our method copes with general handles, including control points, skeleton bones and their combination.

2.3. Mesh Deformation using Sparsity

Sparsity has been widely used in various mesh deformation methods. Xu et al. [22] present a review about a few representative examples of how the interaction between sparsity-based methods and geometric processing can enrich both fields. Gao et al. [23] introduce general l_p norms to shape deformation, and show that different p values influence the distribution of unavoidable distortions. Deng et al. [24] introduce an $l_{1,2}$ sparse regularization penalty into their framework to explore local deformation. Le et al. [25] introduce Smooth Skinning Decomposition with Rigid Bones (SSDR) in order

to solve the inverse problem of the LBS framework. With the sparseness constraint on the weight map, SSDR can be used for traditional skinning decomposition tasks.

3. Energy Formulation

3.1. Preliminary

Our goal is to define smooth deformations for 2D or 3D shapes by blending affine transformations at arbitrary handles. Let $\Omega \subset \mathbb{R}^2$ or $\mathbb{R}^3$ be the volumetric domain enclosed by the union of the given shape $\mathcal{S}$ and cage controls (if any). We denote the handles by $H_j \subset \Omega, j = 1, ..., n_h$, where n_h is the total number of handles. A handle can be a single point, a region, a skeleton bone or a vertex of a cage. The user defines an affine transformation $\mathbf{T}_j$ for each handle H_j, and all points $\mathbf{p} \in \Omega$ are deformed by their weighted combinations:

$$\mathbf{p}_i^{'} = \sum_{j=1}^{n_h} w_{ij} \mathbf{T}_j \begin{bmatrix} \mathbf{p}_i \\ 1 \end{bmatrix}, \tag{1}$$

where $\mathbf{p}_i$ and $\mathbf{p}_i^{'}$ are the vertex coordinates before and after deformation, $\mathbf{T}_j$ is the affine matrix of handle H_j, and w_{ij} is the weight of handle H_j on vertex i.

To calculate the weights in the LBS algorithm, Jacobson et al. [1] proposed bounded biharmonic weights. They define the weight vector $\mathbf{w}_j$ of the j^{th} handle (consisting of its weights on all vertices) as the minimizer of a higher-order shape-aware smoothness functional, namely, the Laplacian energy, with some constraints.

$$\underset{\mathbf{w}_j, j=1,...,n_h}{\arg\min} \frac{1}{2} \int_{\Omega} \|\Delta \mathbf{w}_j\|^2 dV \tag{2}$$

subject to:

$$\mathbf{w}_j|_{H_k} = \delta_{jk} \tag{3}$$

$$\sum_{j=1}^{n_h} \mathbf{w}_j(\mathbf{p}) = 1, \forall p \in \Omega \tag{4}$$

$$\mathbf{0} \leq \mathbf{w}_j(\mathbf{p}) \leq \mathbf{1}, j = 1, ..., n_h, \forall p \in \Omega \tag{5}$$

δ_{jk} is the Kronecker function. The bounded biharmonic weights have some competitive properties, including smoothness, non-negativity, shape-awareness, partition of unity, locality and sparsity, and no local maximum. Please refer to [1] for details.

The Laplacian energy Eq. 2 is discretized using the standard linear FEM Laplacian $\mathbf{M}^{-1}\mathbf{L}$ where $\mathbf{M}$ is the lumped mass matrix (with Voronoi area/volume $\mathbf{M}_i$ of vertex v_i on each diagonal entry i) and $\mathbf{L}$ is the symmetric stiffness matrix. After discretizing the continuous integral term, we have

$$\begin{aligned}\sum_{j=1}^{n_h}\frac{1}{2}\int_{\Omega}||\Delta\mathbf{w}_j||^2dV &\approx \sum_{j=1}^{n_h}\frac{1}{2}(\mathbf{M}^{-1}\mathbf{L}\mathbf{w}_j)^T\mathbf{M}(\mathbf{M}^{-1}\mathbf{L}\mathbf{w}_j)\\ &= \frac{1}{2}\sum_{j=1}^{n_h}{\mathbf{w}_j}^T(\mathbf{L}\mathbf{M}^{-1}\mathbf{L})\mathbf{w}_j\end{aligned} \tag{6}$$

Through discretization, it is possible to convert an integral form which is difficult to solve into a quadratic form which is easy to compute. The above constraints Eqs. 3-5 are all linear equations or inequalities w.r.t. $\mathbf{w}_j$. Once we have the matrices $\mathbf{M}$ and $\mathbf{L}$ from the mesh representation of the 3D model, we can transform the problem into solving quadratic minima under linear constraints.

In this paper, we develop a novel data-driven approach to weight optimization. We therefore assume a set of deformed shapes with the same topology is given, and optimize weights for LBS according to the given handles.

3.2. ARAP Energy

To better evaluate the local shape preservation of the deformation, following [10], we minimize an as-rigid-as-possible (ARAP) energy [3] E_{arap} with the deformed vertex coordinates obtained from Eq. 1. To better capture piecewise rigid deformation, reduce computation and avoid overfitting, we also partition the mesh into a set of regions $\{\mathcal{G}_g\}$, $g = 1, 2, \ldots, |\mathcal{G}|$ and $|\mathcal{G}|$ is the number of regions (viewed as edge groups). For each region $\mathcal{G}_g$, we assign a local rotation matrix $\mathbf{R}_g$. Then, the ARAP energy can be written as:

$$E_{arap} = \sum_{g}\sum_{(i,j)\in G_g}\tilde{w}_{ij}\|(\mathbf{p}'_i - \mathbf{p}'_j) - \mathbf{R}_g(\mathbf{p}_i - \mathbf{p}_j)\|^2 \tag{7}$$

where, $\tilde{w}_{ij}$ is the cotangent weight [26] defined as

$$\tilde{w}_{ij} = \frac{1}{2}(\cot \alpha_{ij} + \cot \beta_{ij}) \tag{8}$$

where α_{ij} and β_{ij} are angles opposite to the edge (i, j) in adjacent triangles and $\mathbf{R}_g$ is the rotation matrix of the edge group $\mathcal{G}_g$. We similarly use k-means clustering on the weight matrix $\mathbf{W}$ to obtain divided edge groups as $\mathbf{W}$ shows how much different handles contribute to the deformation of every mesh vertex [10].

3.3. Data-driven Energy

Let $\mathcal{Q}$ be a deformed mesh. Ideally, the weights should allow accurate reconstruction of the given deformed mesh. We introduce a data-driven energy term that measures the difference of the mesh obtained by the deformation and the known deformed mesh $\mathcal{Q}$ in the dataset. We assume some example deformations are given, so the data-driven energy will consider deviations of all the deformed meshes. For simplicity, we now define the energy for a deformed mesh $\mathcal{Q}$. Let $\mathbf{q}_i$ be the vertex coordinates of the known deformed mesh $\mathcal{Q}$, and denote by $\mathbf{p}'_i$ the vertex coordinates of the deformed mesh obtained by LBS deformation. n_p is the number of vertices on the mesh. The data-driven energy w.r.t. $\mathcal{Q}$ is

$$E_{diff} = \sum_{i=1}^{n_p} \|\mathbf{p}'_i - \mathbf{q}_i\|^2 \tag{9}$$

3.4. Sparsity Energy

With the ARAP and data-driven energy terms above, the learned weights allow the example shapes to be reconstructed. However, the effective control range of each handle may still be beyond what it should be. This is particularly problematic for piecewise rigid deformation as a handle may have effect on vertices that should be controlled by an adjacent handle. To address this, we introduce a sparsity term that promotes the change of weights to be located sparsely. More specifically, we define the sparsity term with the Laplace matrix $\mathbf{L}$

$$E_{sparse} = \sum_{j=1}^{n_h} |\mathbf{L}\mathbf{w}_j|. \tag{10}$$

The Laplace matrix captures the relationships between adjacent vertices, and the l_1 norm promotes sparse distribution. This helps make the effective range of each handle in control and avoid propagating handle deformation to regions which should not be affected.

3.5. Overall Energy

Given a reference model (the initial model before deformation), and a collection of m deformed shapes, our overall energy combines the three energy terms Eq. 7, Eq. 9 and Eq. 10 over all the m shapes. For the k^{th} deformed shape, the overall energy is defined as

$$\begin{aligned} E_k =& E_{diff}^{(k)} + \lambda_1 E_{sparse}^{(k)} + \lambda_2 E_{arap}^{(k)} \\ =& \sum_{i=1}^{n_p} \|\mathbf{p}_i^{'(k)} - \mathbf{q}_i^{(k)}\|^2 + \lambda_1 \sum_{j=1}^{n_h} |\mathbf{L}\mathbf{w}_j| \\ &+ \lambda_2 \sum_{g} \sum_{(i,j)\in\mathcal{G}_g} \tilde{w}_{ij} \|(\mathbf{p}_i^{'(k)} - \mathbf{p}_j^{'(k)}) - \mathbf{R}_g^{(k)}(\mathbf{p}_i - \mathbf{p}_j)\|^2 \end{aligned} \tag{11}$$

where λ_1 and λ_2 are the relative weights that balance the three energy terms, and $^{(k)}$ indicates the k^{th} deformed shape. We sum Eq. 11 for each model in the dataset to get our final target energy term E_{total}:

$$E_{total} = \sum_{k=1}^{m} E_k. \tag{12}$$

4. Algorithmic Solution

We now present the algorithmic solution of our approach, including working out weights and producing deformed meshes following handle movement.

4.1. Weight Optimization

Given a model dataset with more than one model with the same topology, select a model in the dataset as the reference model (the initial model before deformation), and select the handles (e.g. control point) on the reference model. Our aim is to find LBS weights that optimize E_{total} in Eq. 12. As E_{total} is related to different unknown

variables, directly optimizing it is difficult. We therefore take an iterative approach that optimizes one set of variables at a time. To start this process, we use bounded biharmonic weights in Eq. 2 to initialize the weights for given handles. We obtain the weight vector $\mathbf{w}_j$ as a column to form a weight matrix $\mathbf{W}$, which is convenient for further calculations. $\mathbf{W}$ is an $n_p \times n_h$ matrix, and w_{ij} is the weight value of the j^{th} handle on the i^{th} vertex.

Observing the energy Eq. 11, we notice that $\mathbf{p}_i^{'(k)}$ and $\mathbf{p}_j^{'(k)}$ can be represented by Eq. 1. The whole energy can be regarded as a function w.r.t. the weights, the affine matrices $\{\mathbf{T}_j^{(k)}\}$ and the rotation matrices $\{\mathbf{R}_g^{(k)}\}$. In the interactive deformation process, the affine matrices $\{\mathbf{T}_j^{(k)}\}$ are obtained through user interaction, but during weight optimization, the affine matrices are not known in advance and need to be optimized. We alternately optimize the following: Given the weights $\mathbf{W}$ initialized with the bounded biharmonic weights, which means that for a given set of rigid transformations $\{\mathbf{R}_g^{(k)}\}$, we look for a set of affine transformation matrices $\{\mathbf{T}_j^{(k)}\}$ that minimizes E_{total}. Then, based on the weights and the obtained $\{\mathbf{T}_j^{(k)}\}$, we minimize E_{total} to obtain the rigid transformations $\{\mathbf{R}_g^{(k)}\}$. Finally, based on the newly obtained $\{\mathbf{T}_j^{(k)}\}$ and $\{\mathbf{R}_g^{(k)}\}$, we optimize the weights by minimizing E_{total}. By repeating this iterative process, the weights can be continually optimized until E_{total} falls below a given threshold or reaches a minimum. We now give each step in detail.

Computing $\{\mathbf{T}_j^{(k)}\}$. Since we have iterative steps, we can initialize all $\{\mathbf{R}_g^{(k)}\}$ to unit matrices. For different models in the dataset, the affine transformation matrices $\{\mathbf{T}_j^{(k)}\}$ and the rotation matrices $\{\mathbf{R}_g^{(k)}\}$ are different and should be optimized separately. For simplicity of description, we consider the deformation energy E_k for a single model and omit the notation k. Let us first consider the partial derivative of E_k w.r.t. $\mathbf{p}'_i$, and the remaining terms are derived from the ARAP energy term and the data-driven term:

$$\frac{\partial E_k}{\partial \mathbf{p}'_i} = \frac{\partial E_{diff}}{\partial \mathbf{p}'_i} + \lambda_2 \frac{\partial E_{arap}}{\partial \mathbf{p}'_i} \tag{13}$$

Consider

$$\frac{\partial E_{arap}}{\partial \mathbf{p}'_i} = 0 \tag{14}$$

$$\frac{\partial E_{diff}}{\partial \mathbf{p}'_i} = 0 \tag{15}$$

From Eq. 14, we can get a linear equation

$$\sum_{(i,j)\in\mathcal{G}_g} \tilde{w}_{ij}(\mathbf{p}'_i - \mathbf{p}'_j) = \sum_{(i,j)\in\mathcal{G}_g} \tilde{w}_{ij}\mathbf{R}_g(\mathbf{p}_i - \mathbf{p}_j) \tag{16}$$

It can then be simplified as:

$$\mathbf{L}\mathbf{p}' = \mathbf{b} \tag{17}$$

where $\mathbf{L}$ is the Laplace matrix with edge weights, and $\mathbf{p}' = \left[\mathbf{p}'_i, \ldots, \mathbf{p}'_{n_p}\right]^T$ contains the deformed coordinates to be calculated. To minimize E_k, let $\frac{\partial E_k}{\partial \mathbf{p}'_i} = 0$, combining the linear equations obtained by Eq. 15 with Eq. 17, we can get

$$\begin{bmatrix} \lambda_2\mathbf{L} \\ \mathbf{I} \end{bmatrix} \mathbf{p}' = \begin{bmatrix} \lambda_2\mathbf{b} \\ \mathbf{q} \end{bmatrix} \tag{18}$$

Taking Eq. 1 into account, we then obtain:

$$\begin{bmatrix} \lambda_2\mathbf{L} \\ \mathbf{I} \end{bmatrix} \begin{bmatrix} \sum_j w_{1j}(\mathbf{p}_1^T, 1)\mathbf{T}_j^T \\ \sum_j w_{2j}(\mathbf{p}_2^T, 1)\mathbf{T}_j^T \\ \vdots \\ \sum_j w_{n_pj}(\mathbf{p}_n^T, 1)\mathbf{T}_j^T \end{bmatrix} = \begin{bmatrix} \lambda_2\mathbf{b} \\ \mathbf{q} \end{bmatrix} \tag{19}$$

We can get the least squares solution of $\{\mathbf{T}_j\}$. We further rewrite Eq. 19 as:

$$\mathbf{S}\mathbf{K}\mathbf{T} = \mathbf{b}' \tag{20}$$

where $\mathbf{S} = \begin{bmatrix} \lambda_2\mathbf{L} \\ \mathbf{I} \end{bmatrix}$, $\mathbf{K}$ is a sparse matrix of size $n_p \times 4n_h$, $\mathbf{T}$ is the transposed matrix of $\{\mathbf{T}_j\}$ of size $4n_h \times 3$. Then we reshape the matrix $\mathbf{T}$ and transform it into a column vector. From Eq. 20, we can directly obtain the least squares solution of $\{\mathbf{T}_j\}$.

Computing $\{\mathbf{R}_g^{(k)}\}$. Obtaining the affine matrices $\{\mathbf{T}_j^{(k)}\}$ completes the first step of the weight optimization process. The second step is to calculate the rotation matrices $\{\mathbf{R}_g^{(k)}\}$ by minimizing E_{total} based on the known weights and the obtained affine transformation matrices $\{\mathbf{T}_j^{(k)}\}$. Again, we consider E_k to simplify the problem. It can be found that in the E_k expression in Eq. 11, only the ARAP energy term E_{arap} is

related to $\{\mathbf{R}_g\}$, and the other terms are constant when given known weights and affine transformation matrices $\{\mathbf{T}_j\}$.

We briefly describe the derivation for the optimal rotation $\mathbf{R}_g$ for a fixed shape pair $\mathcal{S}$, $\mathcal{S}'$ (before and after deformation). For convenience, let us denote the edge $\mathbf{e}_{ij} := \mathbf{p}_i - \mathbf{p}_j$, and similarly $\mathbf{e}'_{ij}$ for the deformed edge $\mathcal{S}'$. From [10], we have

$$\mathbf{R}_g = \arg\max Tr(\mathbf{R}_g \sum_{(i,j)\in\mathcal{G}_g} \tilde{w}_{ij}\mathbf{e}_{ij}\mathbf{e}_{ij}^{'T}) \tag{21}$$

Denote by $\mathbf{S}_g$ the covariance matrix

$$\mathbf{S}_g = \sum_{(i,j)\in\mathcal{G}_g} \tilde{w}_{ij}\mathbf{e}_{ij}\mathbf{e}_{ij}'^{\,T} = \mathbf{P}_g\mathbf{D}_g(\mathbf{P}'_g)^T. \tag{22}$$

The matrix $\mathbf{D}_g$ is a diagonal matrix containing the edge weights $\tilde{w}_{ij}$, $\mathbf{P}_i$ is a $3 \times |\mathcal{G}_g|$ matrix containing $\mathbf{e}_{ij}$'s as its columns, and similarly for $\mathbf{P}'_i$. The rotation matrix $\mathbf{R}_g$ maximizing $Tr(\mathbf{R}_g\mathbf{S}_g)$ is obtained when $\mathbf{R}_g\mathbf{S}_g$ is symmetric positive semi-definite. $\mathbf{R}_g$ is derived from the singular value decomposition of $\mathbf{S}_g = \mathbf{U}_g\Sigma_g\mathbf{V}_g^T$:

$$\mathbf{R}_g = \mathbf{V}_g\mathbf{U}_g^T \tag{23}$$

If $\det(\mathbf{R}_g)\leq 0$, it is necessary to change the sign of the column in the matrix $\mathbf{S}_g$ corresponding to minimum singular value of the matrix, so that $\det(\mathbf{R}_g)>0$.

Optimizing Weights. The final step in optimization is the most important step, which will directly derive the optimized weights we need. Since the weights are the same in each E_k, we consider the total energy E_{total}. The ARAP energy term E_{arap} and the data-driven term E_{diff} can be converted to the following according to the

derivation of Eqs. 17 and 18:

$$
\begin{aligned}
&\sum_{k=1}^{m}(\sum_{i=1}^{n_p}\|\mathbf{p}_i^{'k}-\mathbf{q}_i^k\|^2+\lambda_2\|\mathbf{L}\mathbf{p}^{'k}-\mathbf{b}^k\|^2)\\
=&\sum_{k=1}^{m}\left\|\begin{bmatrix}\lambda_2\mathbf{L}\\ \mathbf{I}\end{bmatrix}\mathbf{p}^{'k}-\begin{bmatrix}\lambda_2\mathbf{b}^k\\ \mathbf{q}^k\end{bmatrix}\right\|^2\\
=&\sum_{k=1}^{m}\left\|\begin{bmatrix}\lambda_2\mathbf{L}\\ \mathbf{I}\end{bmatrix}\begin{bmatrix}\sum_j w_{1j}(\mathbf{p}_1^T,1)\mathbf{T}_j^{kT}\\ \sum_j w_{2j}(\mathbf{p}_2^T,1)\mathbf{T}_j^{kT}\\ \vdots\\ \sum_j w_{n_pj}(\mathbf{p}_{n_p}^T,1)\mathbf{T}_j^{kT}\end{bmatrix}-\begin{bmatrix}\lambda_2\mathbf{b}^k\\ \mathbf{q}^k\end{bmatrix}\right\|^2
\end{aligned}
\tag{24}
$$

Substituting the above equation into E_{total}'s expression, Eq. 12 gives the following:

$$
\begin{aligned}
E_{total}=&\sum_{k=1}^{m}(\lambda_1\sum_{j=1}^{n_h}|\mathbf{L}\mathbf{w}_j|+\\
&\left\|\begin{bmatrix}\lambda_2\mathbf{L}\\ \mathbf{I}\end{bmatrix}\begin{bmatrix}\sum_j w_{1j}(\mathbf{p}_1^T,1)\mathbf{T}_j^{kT}\\ \sum_j w_{2j}(\mathbf{p}_2^T,1)\mathbf{T}_j^{kT}\\ \vdots\\ \sum_j w_{n_pj}(\mathbf{p}_{n_p}^T,1)\mathbf{T}_j^{kT}\end{bmatrix}-\begin{bmatrix}\lambda_2\mathbf{b}^k\\ \mathbf{q}^k\end{bmatrix}\right\|^2)
\end{aligned}
\tag{25}
$$

E_{total} above involves both l_1 and l_2 norms, but it is still a convex optimization problem. We can efficiently solve it using CVX [27, 28] as the sparse solver to calculate the weights for all handles simultaneously.

In practice, we apply the above three steps iteratively: After determining any two of the weight matrix $\mathbf{W}$, the affine transformation matrices $\{\mathbf{T}_j^{(k)}\}$, and the rotation matrices $\{\mathbf{R}_g^{(k)}\}$, the remaining one is solved, until the total energy term E_{total} reaches a minimum or below a given threshold. The entire workflow is shown in Fig. 1.

4.2. Weight Guided Model Deformation

After having the optimized weights, the deformed shape can be obtained using Eq. 1. However, this requires specifying affine transformations for handles. This can be achieved with appropriate user interface. Alternatively, we develop an approach

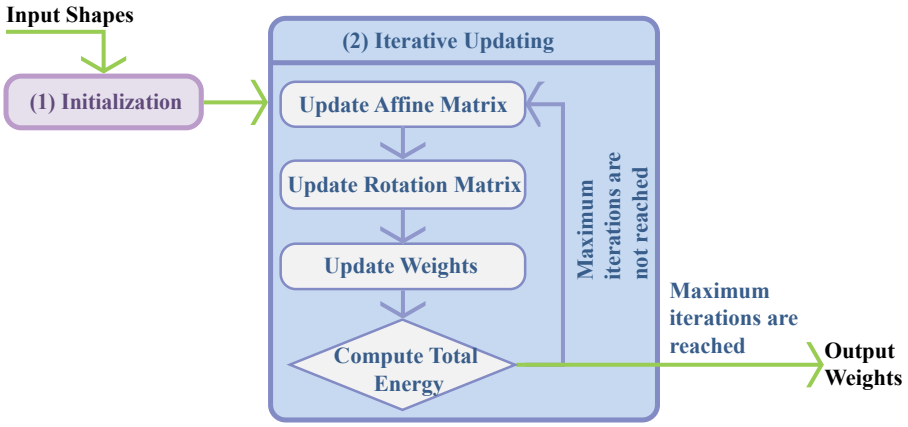


Figure 1: The entire optimization framework for weights.

which derives affine transformations using handle coordinates. To achieve this, we follow the method of weight optimization to obtain the affine matrices by optimizing the deformation energy. Ignoring the sparse regularization term which is irrelevant to the affine transformation matrices and the data-driven term only used for optimization, this only leaves the ARAP term to obtain the affine transformation matrices:

$$E_{transfer} = E_{arap} = \sum_{g} \sum_{(i,j)\in\mathcal{G}_g} \tilde{w}_{ij} \left\| (\mathbf{p}_i^{'} - \mathbf{p}_j^{'}) - \mathbf{R}_g(\mathbf{p}_i - \mathbf{p}_j) \right\|^2 \tag{26}$$

The steps for optimizing the above equation are similar to the procedure of optimizing the weights. The only difference is that the weights are already determined, so only the affine matrices $\{\mathbf{T}_j\}$ and the rotation matrices $\{\mathbf{R}_g\}$ need to be optimized.

On the basis of Eq. 26, we also need to add some model constraints to the entire optimization process, such as the coordinates of the handles given by user interactions:

$$\mathbf{p}_i^{'} = \mathbf{c}_l, l \in D \tag{27}$$

where D denotes the set of the control points in the vertex set. They are determined by the stationary points and control points selected by users. Adding these constraints to Eq. 26 only needs to assign 0 to the corresponding row and column of the Laplace matrix $\mathbf{L}$, and update the vector on the right hand side of the equation with $\mathbf{c}_l$. The same changes are required when calculating the rotation matrices $\{\mathbf{R}_g\}$. We can ob-

tain the affine matrices after several iterations of optimization. Combined with the optimized weights, the model can then be deformed by the linear blend skinning algorithm. Different from the weight optimization, when the optimized affine matrices are obtained from Eq. 26, except that the SVD decomposition of the rotation matrix $\{\mathbf{R}_g\}$ is nonlinear, the rest of the operations are linear, so the deformation can be achieved in real-time. It should be emphasized that the method for obtaining the handle coordinates and then solving the affine matrices is only a convenient way, and specifying affine transformations directly is still more general.

5. Experimental Results and Discussions

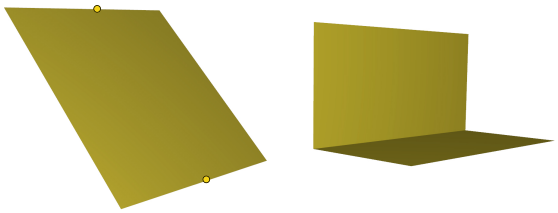
In this section, we compare our optimized weights with the original bounded biharmonic weights and several state-of-the-art deformation methods [16], [17], [10], [3] and [13]. Finally, we prove the necessity of each term in our energy function.

5.1. Comparison Experiments

We use various datasets to compare the LBS algorithm using our optimized weights with that using the original bounded biharmonic weights, and with [10]. These datasets are Card, Horse [29], Face, Pants [30], Hand, and SCAPE [31]. During comparison, we use the algorithm described in Sec. 4.2.

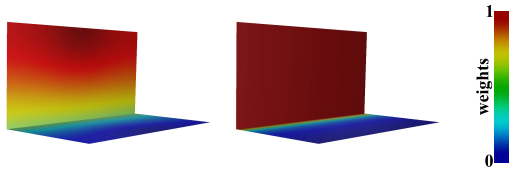
Fig. 2(a) shows a card model with 2,500 vertices and 4,802 triangular faces. We choose the two midpoints of opposite sides of the card as two control points. The bounded biharmonic weights of these two control points are calculated using Eq. 2, which are shown in Fig. 3(a). We only visualize the weights w.r.t. one control point, as the weights w.r.t. the other control point are symmetric.

The weight is large in the red area, and small in blue area (see Fig. 3(c) for the color bar). It can be seen that the bounded biharmonic weights are centered on the control point and gradually decrease towards the surrounding, showing good continuity and smoothness. As expected, this weight distribution leads to a very soft deformation result with the bounded biharmonic weights (see Fig. 4(a)). This is not natural, and in particular different from the given deformation example in Fig. 2(b) which shows the



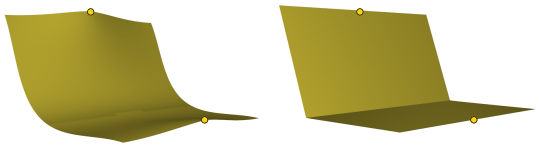
(a) Original **(b) Known Model**

Figure 2: Card models. (a) is the source Card model with two control points located in the middle of a pair of opposite sides. (b) is the example shape used in our optimization.



(a) Jacobson et al. 2011 **(b) Our** **(c)**

Figure 3: Visualization of the weights on the Card model. (a) is the bounded biharmonic weights and (b) is our optimized weights. (c) shows the color bar. We only show the weights of one control point due to the symmetry distribution of the weights.



(a) Jacobson et al. 2011 **(b) Our**

Figure 4: Card deformation comparison.

card should be piecewise rigid. We use this model and the original model to optimize the weights, as shown in Fig. 3(b). Compared with Fig. 3(a), our optimized weights are near 1 for half of the card close to the control point and near 0 for the remaining half of the card away from the control point. The weights near the folding line of the card drop from 1 to 0 rapidly. We can conclude that the card is (almost) divided into two parts, where each part is controlled by the respective control point. Fig. 4(b) shows the deformation result using our optimized weights. It is obvious that our result is consistent with expectation. By further adjusting the position of the control points, our method can easily produce the deformed card with different folding angles.

As mentioned in the previous section, clustering contributes to the optimization of weights, and the weights also determine the results of clustering. Fig. 5 shows the result of clustering, which clearly corresponds to the symmetric weight distribution of the two control points.

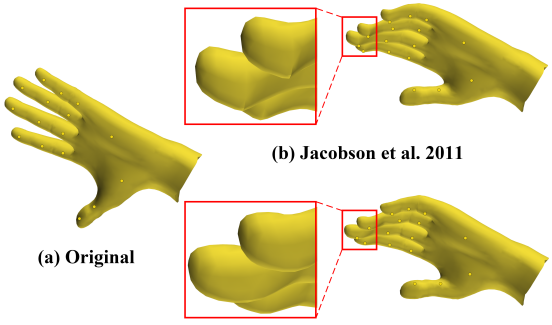
Figure 5: Clustering result on the Card model.

We also compare our weights with the bounded biharmonic weights [1] on Pants and Hand. In the Pants experiment, we use a pair of "kicking" pants as an example, while in the Hand experiment, we use six examples to optimize the weights. The deformation results are shown in Figs. 6 and 7. It can be seen from Fig. 6 that our weights can well handle rigid deformation and from Fig. 7 that our weights produce more plausible results due to the data-driven term in our total energy.

We then compare our method with [10]. We take Face, Horse [29] and SCAPE [31] as test cases. We use multiple examples (larger than 10) in these cases. Fig. 8(a) shows a face model with four control points, two of which are at the upper eyelids. The bounded biharmonic weights of the control point on the left eyelid are visualized in

(a) Original **(b) Jacobson et al. 2011** **(c) Our**

Figure 6: Deformation result comparison with [1] on the Pants model.

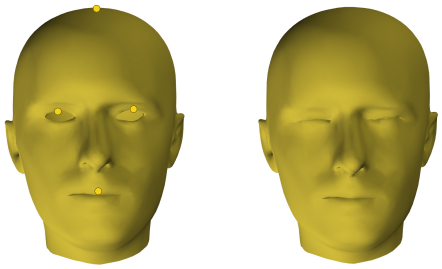


(b) Jacobson et al. 2011

(a) Original

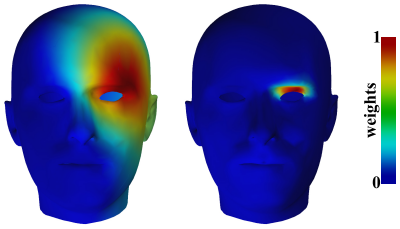
(c) Our

Figure 7: Deformation result comparison with [1] on the Hand model. It is obvious that our result is more smooth.



(a) Original (b) One of Known Models

Figure 8: Face models. (a) is the source face model with four control points, and (b) is an example shape used in our optimization.



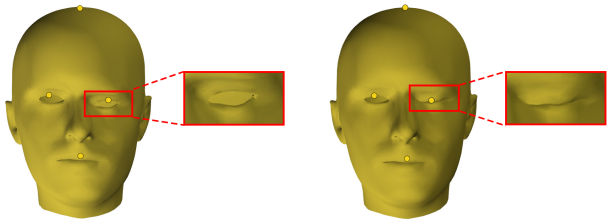
(a) Jacobson et al. 2011 (b) Our (c)

Figure 9: Weight visualization of the Face model. (a) is the bounded biharmonic weight result and (b) is our optimized weight result. (c) shows the color bar.

Fig. 9(a). It can be seen that the bounded biharmonic weights still exhibit continuous, smooth characteristics, but the control area distribution of all control points is relatively uniform, especially the control area of the control points on the eyelids greatly exceeds the range of the eyelids which means that if someone drags the control points on the eyelids, it will cause a large part of the surrounding face to deform, resulting in unnatural deformation. For our method several face models from the dataset are used to optimize the weights, and one of which has closed eyes, as shown in Fig. 8(b). The optimized weights are shown in Fig. 9(b). The area where the weight is non-zero is limited to the eyelid part.

We experiment with an open source interactive deformation platform introduced in [10] and compare with the method in [10]. Deformation result comparisons are shown in Fig. 10. Our result shows a very natural closed eye, where the result of [10] pulls part of the face around the eye.

In order to understand the optimization process more clearly, we visualize the



(a) Jacobson et al. 2012 (b) Our

Figure 10: Face deformation comparison. Our result shows a very natural closed eye, which is better than the result of [10] which obviously pulls part of the face around the eye.

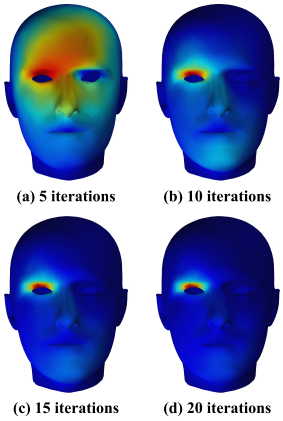


Figure 11: Weights change during the optimization on the Face model.

weights every five iterations in the optimization process, as shown in Fig. 11. From this series of figures, we can see that the weights gradually converge from the concentric distribution of the original bounded biharmonic weights until the red area representing the larger weights gradually shrinks to the area of the upper eyelid, while the weights in other areas are almost changed to 0.

To demonstrate the effect with more control points, we select nine control points on the face model, and the deformation result is shown in Fig. 12. Since the entire facial expression space cannot be modeled by skinning weights only, some expressions are hard to obtain even by using a large number of control points.

In addition, we use the Horse model [29] and test the combination of two types of handles: control points and skeleton bones. We use control points to control horsetail and skeleton to control the body of the horse. The results are shown in Fig. 13. In the

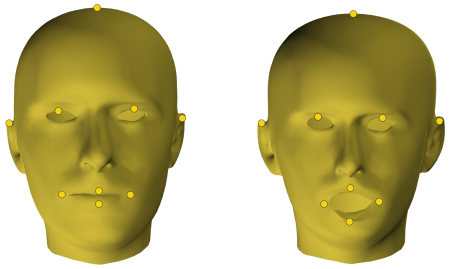


Figure 12: Face model with more control points. (a) is the original face model with nine control points, (b) is the deformed shape.

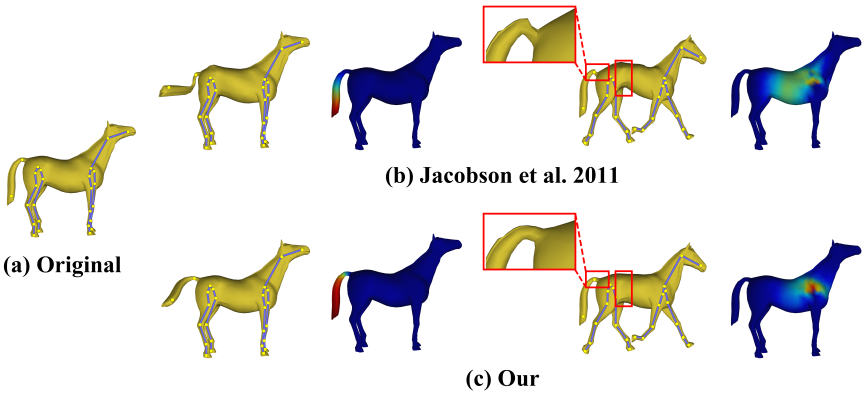


Figure 13: Weight visualization and deformation results on the Horse model [29]. The horsetail in the result of [10] has obvious distortion and unnatural bending, while in our result it is more natural and smooth, which gives a feeling that the horse is swinging the tail. The red rectangles show that in the result of [10], the joint between the tail and the body has some artifacts and the abdomen has abnormal contraction.

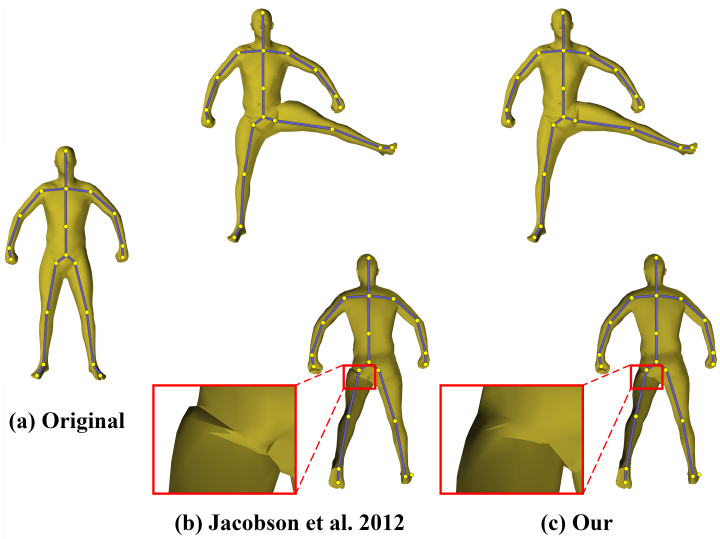


Figure 14: Deformation result comparison with [10] on the SCAPE model [31]. It can be seen that our method can well handle the deformation around the skeleton joint due to the data-driven term.

first pair of comparison, the horsetail in the result of [10] has obvious distortion and unnatural bending, while in our result it is more natural and smooth, giving us a feeling that the horse is swinging its tail. In the second pair of comparison, we use the skeleton to deform the horse body. It can be seen from the weight visualization that the bounded biharmonic weights of the leg have some redundant parts on abdomen and back of the body, which influence the abdomen to contract. Also, the joint between the tail and the body has some artifacts.

Our method is flexible with handle forms, and we can also optimize weights on skeleton handles. We use the SCAPE [31] model to compare deformation results. Fig. 14 shows the comparison results using skeleton handles. It can be seen that our weights can well handle the deformation around skeleton joints.

We further compare our method with another data-driven deformation method [13]. We use Hand and Pants models to illustrate the superiority of our method. In the comparison, the models used for our weight optimization and the models as the basis for [13] are the same. The results of comparison are shown in Figs. 15 and 16, which demonstrate that the method [13] generates results with undesired distortions whereas our results look natural.

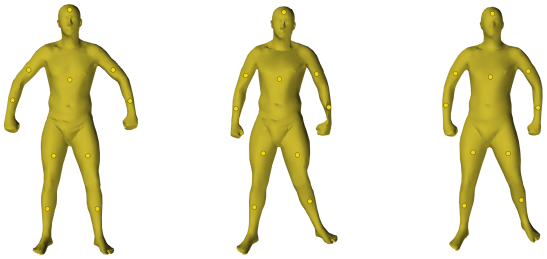
(a) Original **(b) Sumner et al. 2005** **(c) Our**

Figure 15: Deformation result comparison on the Hand model with [13] where the result of [13] has elongated fingers, while our method produces a normal result.

(a) Original **(b) Sumner et al. 2005** **(c) Our**

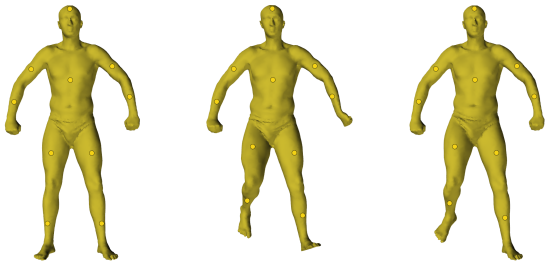
Figure 16: Deformation result comparison on the Pants model with [13]. Our result shows a good "kicking" motion while [13] shows a distorted result.

We also compare with data-driven methods [16] and [17] on the SCAPE [31] model based on control points (as these methods can only take control points as handles). When compared to [16], we randomly select five examples, whereas for [17], 40 examples are randomly selected. We use the same models to optimize weights when compared with [16] and [17] respectively. The deformation comparisons are shown in Figs. 17 and 18. Our method avoids the artifacts of alternative methods.



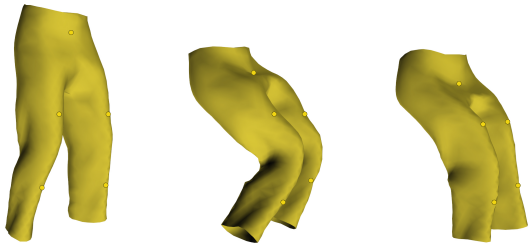
(a) Original **(b) Fröhlich and Bostch** **(c) Our**

Figure 17: Deformation result comparison on the SCAPE [31] model with [16]. There are some artifacts on the human hand in the result of [16].



(a) Original **(b) Gao et al. 2016** **(c) Our**

Figure 18: Deformation result comparison on the SCAPE [31] model with [17]. We only move the control points on the right leg, and there are some extra motions like the flipped hand in [17] while our result only has the deformation of the right leg as expected.



(a) Original **(b) Sorkine et al. 2007** **(c) Our**

Figure 19: Deformation result comparison on the Pants model with ARAP [3]. The result of [3] has blended pants legs, while our result has the correct squat posture.

Finally, we compare our method with ARAP deformation [3]. Our method has the data-driven term, which gives the deformation more information to produce desired results. The results are shown in Fig. 19.

Our method is very efficient, achieving real-time performance. We compare the online deformation time with data-driven methods [17], [30] and [13] on the SCAPE model [31]. The results are shown in Fig. 20. Our method only involves matrix multiplications and SVD decomposition which cost little time and due to the offline optimization, the time of our deformation method does not increase as the number of examples increases, while the time of other data-driven methods are affected by the number of examples (components) used, and may not maintain real-time performance when a large number of examples (components) are involved.

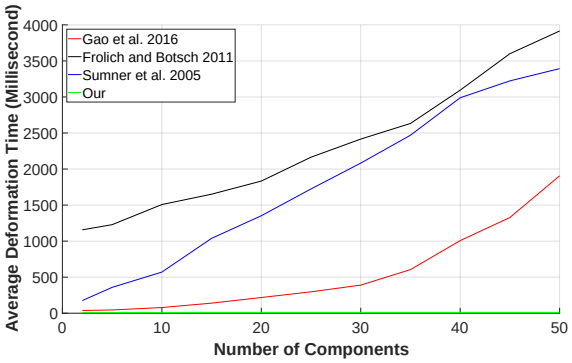
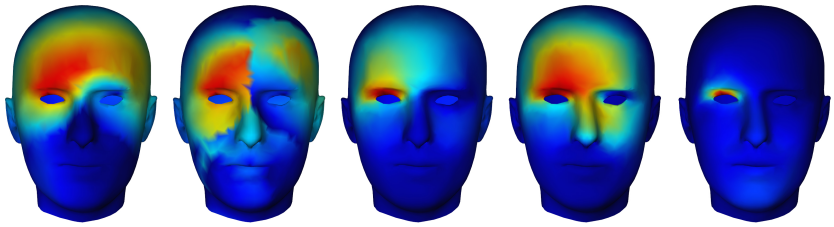


Figure 20: Deformation time comparison on SCAPE [31] model with [17], [30] and [13].



(a) Choose $\|Lw\|_2$ **(b) Without Eq.10** **(c) Without Eq.7** **(d) Without Eq.9** **(e) Total Energy**

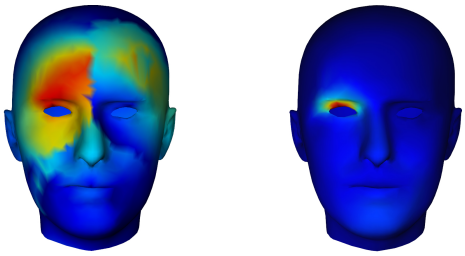
Figure 21: Validation of each energy term. It is obvious that each energy term is needed.

5.2. Ablation study

In order to verify that the energy terms we mentioned in Sec. 3 can play their role, we conduct an ablation study. We take the face model as an example, and the selection of control points is the same as Fig. 8.

To verify the role of the l_1 norm used in the sparsity term, we design two experiments, namely removing the l_1 norm sparsity term and replacing the l_1 norm with the l_2 norm. Fig. 21(a) shows the result of replacing it with the l_2 norm. The effect of the l_2 norm tends to be averaged, and it is not sparse, which is contrary to our needs. The weight distribution in Fig. 21(b) is the result of optimization after removing the l_1 norm sparsity regularization. When E_{total} no longer has a significant decrease trend, the weight distribution of the control point still covers a larger area than it should be. From these, we can clearly confirm that the l_1 norm sparsity term has played an important role in the optimization process.

The data-driven term provides essential guidance for the optimization to capture



(a) Separate **(b) Simultaneous**

Figure 22: Comparison between sparsification performed simultaneously with optimization (b) and sparsification performed separately from optimization (a). Simultaneous optimization in our method is more effective.

Weights	Separate	Simultaneous
< 0.1	834	2408
< 0.05	432	1958

Table 1: Comparison of sparsity between two optimization strategies, namely sparsification performed separately from optimization and sparsification performed simultaneously with optimization, measured by the number of vertices with small (near-zero) weights. It can be seen that simultaneous optimization has better sparsification effects.

the object deformation behavior. Fig. 21(c) shows the result of removing the data-driven term. Lacking the guidance of the data-driven term, the optimization simply makes E_{total} smaller, without following the deformation behavior. Fig. 21(d) shows the weight distribution of the experiment that removes the ARAP term for optimization, which again fails to produce desired weight distribution, as shown in Fig. 21(e).

Through these sets of experiments, we confirm that the three energy terms that make up the total energy E_{total} are all necessary and indispensable, and their role is consistent with the previous description.

Moreover, our sparsification and optimization are performed simultaneously. We compare this with an alternative strategy similar to [11] where sparsification is performed separately from the optimization. Fig. 22 shows the visual comparison and statical comparison is presented in Table 1. Our simultaneous sparsification and optimization strategy leads to significantly better sparsification effects.

5.3. Limitations

Our method has some limitations. When control points are used for deformation, since their control range is limited, they are not particularly suited to large-scale deformations. Instead, control points are more suitable for fine-tuning on small scales, such as facial expressions, muscle stretching, etc. For large-scale motions, one still needs skeletons to achieve better results. Another limitation is for complex models with a large number of vertices, the speed of offline optimization of weights can be slow, and the memory usage is high. We plan to further improve the efficiency in the offline processing. Nevertheless, in the online stage, our method remains efficient.

6. Conclusion

In this paper, we introduce a data-driven approach to optimizing weights for linear blend skinning deformation. By introducing our new data-driven and sparsity regularization terms, the deformation weights effectively follow the deformation behavior of given examples, leading to more natural deformation results and avoiding unintuitive global effects. Our online computation is very efficient, and keeps running times constant with arbitrary number of examples. Our method can also be used with different forms of handles.

Acknowledgments

This work was supported by National Natural Science Foundation of China (No. 61872440 and No. 61828204), Beijing Natural Science Foundation (No. L182016), Young Elite Scientists Sponsorship Program by CAST (No. 2017QNRC001), Youth Innovation Promotion Association CAS, Huawei HIRP Open Fund (No. HO2018085141), CCF-Tencent Open Fund, SenseTime Research Fund and Open Project Program of the National Laboratory of Pattern Recognition (NLPR).

References

[1] A. Jacobson, I. Baran, J. Popovic, O. Sorkine, Bounded biharmonic weights for real-time deformation., ACM Trans. Graph. 30 (4) (2011) 78–1.

[2] N. Magnenat-Thalmann, R. Laperrire, D. Thalmann, Joint-dependent local deformations for hand animation and object grasping, in: In Proceedings on Graphics interface88, Citeseer, 1988.

[3] O. Sorkine, M. Alexa, As-rigid-as-possible surface modeling, in: Symposium on Geometry processing, Vol. 4, 2007, pp. 109–116.

[4] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, H.-P. Seidel, Laplacian surface editing, in: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing, ACM, 2004, pp. 175–184.

[5] Y. Yu, K. Zhou, D. Xu, X. Shi, H. Bao, B. Guo, H.-Y. Shum, Mesh editing with poisson-based gradient field manipulation, ACM Transactions on Graphics (TOG) 23 (3) (2004) 644–651.

[6] O.-C. Au, C.-L. Tai, L. Liu, H. Fu, Dual laplacian editing for meshes, IEEE Transactions on Visualization and Computer Graphics 12 (3) (2006) 386–395.

[7] S.-h. Liao, R.-f. Tong, J.-x. Dong, F.-d. Zhu, Gradient field based inhomogeneous volumetric mesh deformation for maxillofacial surgery simulation, Computers & Graphics 33 (3) (2009) 424–432.

[8] S.-h. Liao, R.-f. Tong, J.-P. Geng, M. Tang, Inhomogeneous volumetric laplacian deformation for rhinoplasty planning and simulation system, Computer Animation and Virtual Worlds 21 (3-4) (2010) 331–341.

[9] M. Colaianni, C. Siegl, J. Süßmuth, F. Bauer, G. Greiner, Anisotropic deformation for local shape control, Computational Visual Media 3 (4) (2017) 305–313.

[10] A. Jacobson, I. Baran, L. Kavan, J. Popović, O. Sorkine, Fast automatic skinning transformations, ACM Transactions on Graphics (TOG) 31 (4) (2012) 77.

[11] J.-M. Thiery, E. Eisemann, Araplbs: Robust and efficient elasticity-based optimization of weights and skeleton joints for linear blend skinning with parametrized bones, Computer Graphics Forum 37 (1) (2018) 32–44.

[12] J. Bai, J. Pan, Y. Yang, H. Qin, Novel metaballs-driven approach with dynamic constraints for character articulation, Science China Information Sciences 61 (9) (2018) 094101.

[13] R. W. Sumner, M. Zwicker, C. Gotsman, J. Popović, Mesh-based inverse kinematics, ACM transactions on graphics (TOG) 24 (3) (2005) 488–495.

[14] T. Neumann, K. Varanasi, S. Wenger, M. Wacker, M. Magnor, C. Theobalt, Sparse localized deformation components, ACM Transactions on Graphics (TOG) 32 (6) (2013) 179.

[15] Z. Huang, J. Yao, Z. Zhong, Y. Liu, X. Guo, Sparse localized decomposition of deformation gradients, Computer Graphics Forum 33 (7) (2014) 239–248.

[16] S. Fröhlich, M. Botsch, Example-driven deformations based on discrete shells, Computer graphics forum 30 (8) (2011) 2246–2257.

[17] L. Gao, Y.-K. Lai, D. Liang, S.-Y. Chen, S. Xia, Efficient and flexible deformation representation for data-driven surface modeling, ACM Transactions on Graphics (TOG) 35 (5) (2016) 158.

[18] Q. Tan, L. Gao, Y.-K. Lai, J. Yang, S. Xia, Mesh-based autoencoders for localized deformation component analysis, in: AAAI Conference on Artificial Intelligence, 2018.

[19] L. Gao, Y.-K. Lai, J. Yang, L.-X. Zhang, L. Kobbelt, S. Xia, Sparse data driven mesh deformation, arXiv preprint arXiv:1709.01250.

[20] A. Murai, Q. Youn Hong, K. Yamane, J. K. Hodgins, Dynamic skin deformation simulation using musculoskeletal model and soft tissue dynamics, Computational Visual Media 3 (1) (2017) 49–60.

[21] B. H. Le, Z. Deng, Robust and accurate skeletal rigging from mesh sequences, ACM Transactions on Graphics (TOG) 33 (4) (2014) 84.

[22] L. Xu, R. Wang, J. Zhang, Z. Yang, J. Deng, F. Chen, L. Liu, Survey on sparsity in geometric modeling and processing, Graphical Models 82 (2015) 160–180.

[23] L. Gao, G. Zhang, Y. Lai, L p shape deformation, Science China Information Sciences 55 (5) (2012) 983–993.

[24] B. Deng, S. Bouaziz, M. Deuss, J. Zhang, Y. Schwartzburg, M. Pauly, Exploring local modifications for constrained meshes, Computer Graphics Forum 32 (2pt1) (2013) 11–20.

[25] B. H. Le, Z. Deng, Smooth skinning decomposition with rigid bones, ACM Transactions on Graphics (TOG) 31 (6) (2012) 199.

[26] M. Meyer, M. Desbrun, P. Schröder, A. H. Barr, Discrete differential-geometry operators for triangulated 2-manifolds, in: Visualization and mathematics III, Springer, 2003, pp. 35–57.

[27] M. Grant, S. Boyd, Graph implementations for nonsmooth convex programs, in: V. Blondel, S. Boyd, H. Kimura (Eds.), Recent Advances in Learning and Control, Lecture Notes in Control and Information Sciences, Springer-Verlag Limited, 2008, pp. 95–110, http://stanford.edu/~boyd/graph_dcp.html.

[28] M. Grant, S. Boyd, CVX: Matlab software for disciplined convex programming, version 2.1, http://cvxr.com/cvx (Mar. 2014).

[29] R. W. Sumner, J. Popović, Deformation transfer for triangle meshes, ACM Transactions on Graphics 23 (3) (2004) 399–405.

[30] R. White, K. Crane, D. A. Forsyth, Capturing and animating occluded cloth, in: ACM Transactions on Graphics (TOG), Vol. 26, ACM, 2007, p. 34.

[31] D. Anguelov, P. Srinivasan, D. Koller, S. Thrun, J. Rodgers, J. Davis, SCAPE: shape completion and animation of people, ACM Transactions on Graphics 24 (3) (2005) 408–416.