Computers & Graphics 46 (2015) 294–305

Contents lists available at ScienceDirect

Computers & Graphics

journal homepage: www.elsevier.com/locate/cag

SMI 2014

Dynamic skin deformation using finite difference solutions for character animation

E. Chaudhry [a], S.J. Bian [a], H. Ugail [b], Xiaogang Jin [c], L.H. You [a], Jian J. Zhang [a,d,*]

[a] National Centre for Computer Animation, Bournemouth University, United Kingdom
[b] Centre for Visual Computing, University of Bradford, United Kingdom
[c] State Key Lab of CAD&CG, Zhejiang University, China
[d] Traction Power State Key Laboratory, Southwest Jiaotong University, China

ARTICLE INFO

Article history:
Received 7 July 2014
Received in revised form
25 September 2014
Accepted 27 September 2014
Available online 13 October 2014

Keywords:
Dynamic skin deformations
Curve-based representation
Finite difference solution
Data-driven methods

ABSTRACT

We present a new skin deformation method to create dynamic skin deformations in this paper. The core elements of our approach are a dynamic deformation model, an efficient data-driven finite difference solution, and a curve-based representation of 3D models. We first reconstruct skin deformation models at different poses from the taken photos of a male human arm movement to achieve real deformed skin shapes. Then, we extract curves from these reconstructed skin deformation models. A new dynamic deformation model is proposed to describe physics of dynamic curve deformations, and its finite difference solution is developed to determine shape changes of the extracted curves. In order to improve visual realism of skin deformations, we employ data-driven methods and introduce skin shapes at the initial and final poses into our proposed dynamic deformation model. Experimental examples and comparisons made in this paper indicate that our proposed dynamic skin deformation technique can create realistic deformed skin shapes efficiently with a small data size.

1. Introduction

Efficiently creating realistic skin deformations is crucial for character animation. Due to its importance, many skin deformation methods have been developed. These methods can be roughly classified into four types: purely geometric; physics-based; data-driven; and curve-controlled.

Purely geometric methods do not consider any physics of skin deformations. This type of methods are efficient in creating deformed skin shapes, but less realistic. Physics-based methods consider physics of skin and the underlying tissues. This type of methods can generate more realistic skin deformations but not ideal for the applications requiring high computing efficiency. Data-driven methods create new skin deformations from known example skin shapes. When example skin shapes are sufficient, this type of methods can generate highly realistic skin deformations. How to reduce the number of example skin shapes but still keep good realism is a main issue of data-driven methods. Since data-driven methods do not consider any physics of skin deformations, they require sufficient skin shapes to achieve realism. Curve-controlled methods use a curve network to define a skin surface. The curves in the network are deformed which are used to create a deformed skin surface. Curve-controlled methods are more efficient than physics-based methods. They can be combined with data-driven methods to achieve realistic skin deformations.

The parametric representation of skin surfaces involves two parametric variables. When physics-based methods are introduced to determine shape changes of skin surfaces, complicated partial differential equations are involved which have to be solved by inefficient numerical methods. In contrast, the parametric representation of curves only involves one parametric variable. When physics-based methods are used to determine shape changes of curves, simple ordinary differential equations are formulated which can be solved efficiently. Due to this advantage, the existing work has developed the finite difference solution of static skin deformations. Since skin deformations are caused by skin surface movements, more realistic skin deformations should consider dynamic effects caused by skin surface movements.

From the above discussions, three challenges should be addressed. The first is to consider dynamic effects to further improve the realism of skin deformations. The second is how to combine physics-based methods with data-driven ones to reduce the number of example skin shapes. The third is to introduce a new representation of 3D surface models to achieve high efficiency of physics-based approaches.

In this paper, we will develop a new skin deformation technique to address these three challenges. The first challenge is met by developing a dynamic deformation model and its finite difference

* Corresponding author. Tel.: +44 1202965055.
E-mail addresses: ehtzaz@yahoo.com, jzhang@bournemouth.ac.uk (J.J. Zhang).

http://dx.doi.org/10.1016/j.cag.2014.09.029
0097-8493/

solution. The second challenge is addressed by combining dynamics of curve deformations with two example skin shapes. And the third challenge is answered by using dynamic curve shape changes to describe dynamic skin surface deformations.

Our contributions are (1) transforming complex dynamic skin surface deformations into simple dynamic curve deformations to simplify the mathematical treatment, (2) developing an efficient finite difference solution to tackle the problem of dynamic curve deformations and achieve high computational efficiency, and (3) combining data-driven techniques with our proposed dynamic deformation model to improve the realism of skin deformations.

The paper is organized as follows. Related work is briefly reviewed in Section 2. Following that, reconstruction of real skin deformation models from the taken photos, curve extraction, and transferring deformed curves to polygon models are introduced in Section 3. A dynamic deformation model and its finite difference solution are investigated in Section 4. Experimental comparisons and applications are demonstrated in Sections 5 and 6, respectively. Finally, conclusions and future work are discussed in Section 7.

2. Related work

Many approaches determining skin deformations have been developed which can be roughly classified into joint-related (purely geometric), physics-based, data-driven, and curve-based. In this section, we review some main research activities.

Joint-related approaches, which relate skin deformations to joint movements, are more efficient but less realistic than physics-based and data-driven approaches.

Thalmann et al. first examined joint-related skin deformation [1]. Following this pioneering work, Lander presented the basic principles of skeleton subspace deformation [2,3]. The shrinkage problems during bending or twisting were discovered in Ref. [4]. In order to tackle these problems, Wang and Phillips proposed the multi-weight enveloping technique [5], Mohr and Gleicher added additional joints [6], Yang et al. introduced curve skeleton skinning [7], and Kavan and Žára developed the spherical blend skinning [8].

Recently, Le and Deng investigated how to extract the linear blending skinning from example skin shapes automatically [9], and Vaillant et al. presented a purely geometric method to deal with skin contact effects and muscular bulges for realistic skin deformations [10].

Among the above methods, Classic Linear Skinning and Dual Quaternion Skinning Methods have been implemented into the Autodesk Maya. The former [1–3] is very popular and efficient. It determines the deformed position of a vertex by a convex linear combination of individual vertex transformations. This method has the limitations of collapsing joints, candy wrapper effect, and volume loss caused by the linear interpolation of vertex positions. The Dual Quaternion Skinning Method [11] provides a solution. It not only interpolates rotations but also blends the rotation centers.

Physics-based techniques use anatomy, and elastic mechanics or biomechanics of skin deformation originating from the movements of muscles and tendons.

Wilhelms and Van Gelder proposed an improved anatomically based approach by modeling muscles, bones and generalized tissues as triangle meshes or ellipsoids, treating muscles as deformable discretized cylinders, and relating skin motion to an elastic membrane model [12]. Nedel and Thalmann presented a human model with three layers, i.e., skeleton, muscle and skin, and introduced a mass-spring system with angular springs to physically simulate muscle deformations [13,14]. Capell et al. presented a framework to predict skeleton-driven dynamic deformations of elastically deformable characters [15]. Guo and Wong provided an approach to calculate and produce skin deformation of thick, irregular shape bodies by using quasi-static linear deformation and finite element method [16]. In order to deal with the wrinkle of skin, Venkataraman et al. combined a kinematic model with a variational model [17]. Galoppo et al. proposed an algorithm to capture the dynamic skeleton–skin interplay through a novel formulation of elastic deformation in the pose space of the skinned surface [18]. Lee et al. introduced a biomechanical model of the human upper body which can simulate the physics-based deformations of the soft tissues [19].

Recently, Kim and Pollard developed a fast physically based simulation system for skeleton-driven characters consisting of a reduced deformable body model with nonlinear finite elements, a linear-time algorithm for skeleton dynamics, and an explicit integration [20]. Based on a novel discretization of corotational elasticity over a hexahedral lattice, McAdams et al. presented a new algorithm to achieve near-interactive simulation of skeleton driven, high resolution elastic models [21]. By introducing a general concept of joint-based deformers, Kavan and Sorkine found a closed-form skinning method to well approximate nonlinear elastic deformations very efficiently [22]. Li et al. proposed a new approach to simulate thin hyperelastic skin [23]. By optimizing both control forces added to a linearized dynamic model and material properties, Li et al. presented a novel method for elastic animation editing with space-time constraints [24].

Data-driven approaches introduce example skin shapes to improve the realism of skin deformations. The more example skin shapes are used, the better realism is achieved.

Lewis et al. proposed the pose space deformation (PSD) technique by combining skeleton-related techniques with shape interpolation [25]. This technique was extended to the weighted pose-space deformation (WPSD) by Rhee et al. [26]. In addition, Mohr and Gleicher developed an automated method to create skin deformation from a set of examples [6]. Allen et al. presented an example-based approach to determine joint-related skin deformations [27]. Wang and Phillips used a modified least-squares fitting technique to compute the weights of a deformation equation and presented a multi-weight enveloping method to deform the skin geometry [5]. Kurihara and Miyata reconstructed a human hand model from CT scans [28]. Weber et al. used given deformation examples to develop a system for skeletal shape deformation [29]. Park and Hodgins created a database of dynamic skin deformations by recording the motion of the skin surface with many motion capture markers, and presented a data-driven technique to synthesize skin deformations by relating static deformations to the poses and dynamic deformations to the actions of muscles [30]. Feng et al. extracted sparse control points and a skinned mesh with bones and bone influence weights from example meshes, proposed a learning method to capture connections between control points and bone deformations, and generated new deformations from the movements of control points [31].

Recently, Le and Deng proposed an automated algorithm called the Smooth Skinning Decomposition with Rigid Bones (SSDR) to extract the linear blending skinning from a set of examples. The algorithm can effectively approximate the skin deformation by a low number of rigid bones and a sparse, convex bone-vertex weight map [9]. In order to tackle the problems that geometric skinning techniques cannot mimic realistic deformations, and other methods using physical simulation or control volume cannot deliver real-time feedback, Vaillant proposed a purely geometric method handling skin contact effects and muscular bulges in real-time [10]. Neumann et al. presented a data-driven statistical model for muscle deformation of the human shoulder–arm region [32]. By formulating a linear blend skinning as an optimization and solving it with an iterative rigging algorithm, Le and Deng introduced an example-based rigging approach to obtain skeleton, joint positions, weights and corresponding bone transformations [33].

Curve-controlled approaches use a curve network to define a 3D surface model. Such a treatment transforms a complex two-dimensional surface deformation problem into a simple one-dimensional curve deformation problem to greatly raise the computational efficiency.

Early research into curve-based approaches is due to the work of Scheepers et al. [34]. They used cross-sectional contours to represent skin surfaces and manipulate these contours to deform human limbs and torso. Pyun et al. investigated how to extract wire curves and deformation parameters from a facial model [35]. Hyun et al. approximated human body parts with elliptic cross sections [36]. You et al. used closed curves to represent skin surfaces and proposed an analytical solution to determine skin deformations [37]. Chaudhry et al. presented a finite difference solution of static skin deformation based on open curves [38].

The work given in this paper is related to but different from the above four types of approaches. It applies rigid-body transformations to exclude shape and position changes caused by translations and joint rotations, relates sculpting forces and skin deformations to joint rotations, combines physics-based with data-driven approaches to achieve realistic skin deformations with two example skin shapes only, and use dynamic curve deformations to obtain high efficiency of dynamic skin deformations.

3. Skin shape reconstruction and conversions between polygon and curve defined models

Since the realism of skin deformations is essential for a desirable skinning method, we will make a comparison among the real deformed skin shapes and the calculated ones by both our solution and other approaches. We first reconstruct real deformed skin shapes from the photos taken from real human arm movements. Then we transfer reconstructed polygonal models into curve defined models. After that, we use our proposed finite difference solution of dynamic curve deformations to obtain deformed shapes of the curves. And finally we determine the new shapes of the polygon models from these deformed curves to obtain dynamic skin deformations. The process is shown in Fig. 1.

3.1. Reconstructing real deformed skin shapes from photos

Comparing with real skin deformation shapes is the best way to examine the realism of a skin deformation method. In order to make such a comparison, we introduce how to obtain real skin deformation shapes through surface reconstruction in this section.

There is a lot of work on surface reconstruction, especially on facial reconstruction. Depending on different applications, different techniques have been developed. In film and game production, facial markers [39], camera arrays [40], and structured light projectors [41] are used to obtain 3D facial geometry of high fidelity. For ordinary users, video-based facial tracking and animation provide a more practical solution. The related techniques include feature displacements in expression change [42], physically-based deformable mesh models [43], data-driven face models [44], Cascaded Pose Regression [45], Constrained Local Model [46], and Supervised Descent Method [47].

In this paper, the goal of reconstructing skin shapes of human arm movements is to provide (1) real skin deformations for comparison, and (2) two example skin shapes used in our proposed approach. Our research focus is the dynamic deformation model and its finite difference solution.

The real skin deformation shapes are reconstructed from the photos taken from a male human arm movement. The movement is divided into a lot of poses. At each pose, 20 photos from different views are taken.

In Fig. 2, we give eight photos selected from the 20 photos taken at the rest (initial) pose of the male human arm movement.

In Fig. 3, we give the photos of skin deformation shapes of the male human arm movement at five different poses. Then, we upload all photos to Autodesk 123D Catch to reconstruct 3D skin deformation models. The reconstructed models are input into Autodesk Maya for further processing. In Fig. 4a, we give the reconstructed skin deformation models at five different poses obtained through Autodesk Maya. Fig. 4b gives the close-up images of the human elbow in Fig. 4a.

3.2. Transferring polygonal models into curve defined models

Each of the reconstructed models is a polygon model. We first transfer it into a curve network, i.e., a curve defined model.

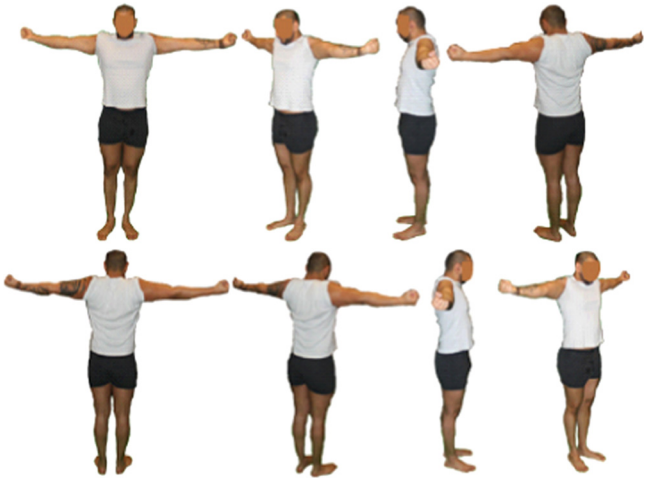

Fig. 2. Photos of a male human arm at the rest pose.

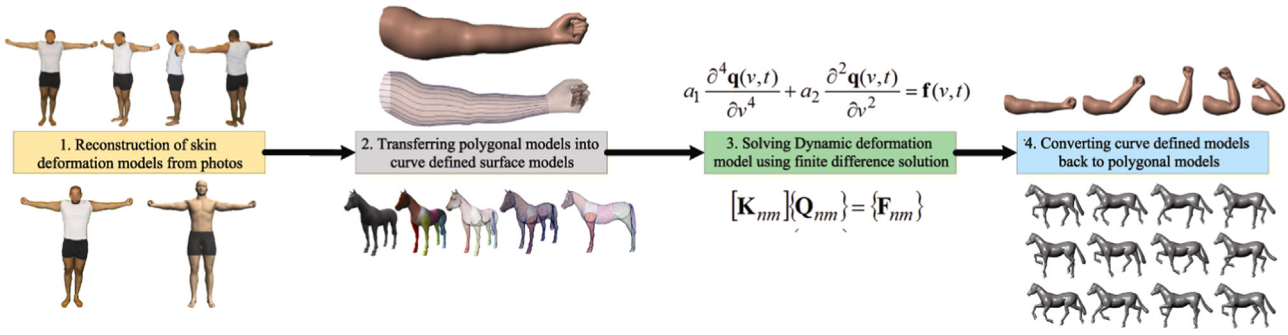

Fig. 1. Process of our method.

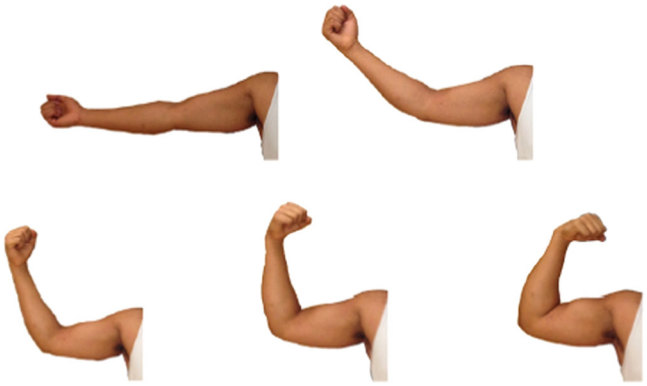

Fig. 3. Skin shapes at five different poses.

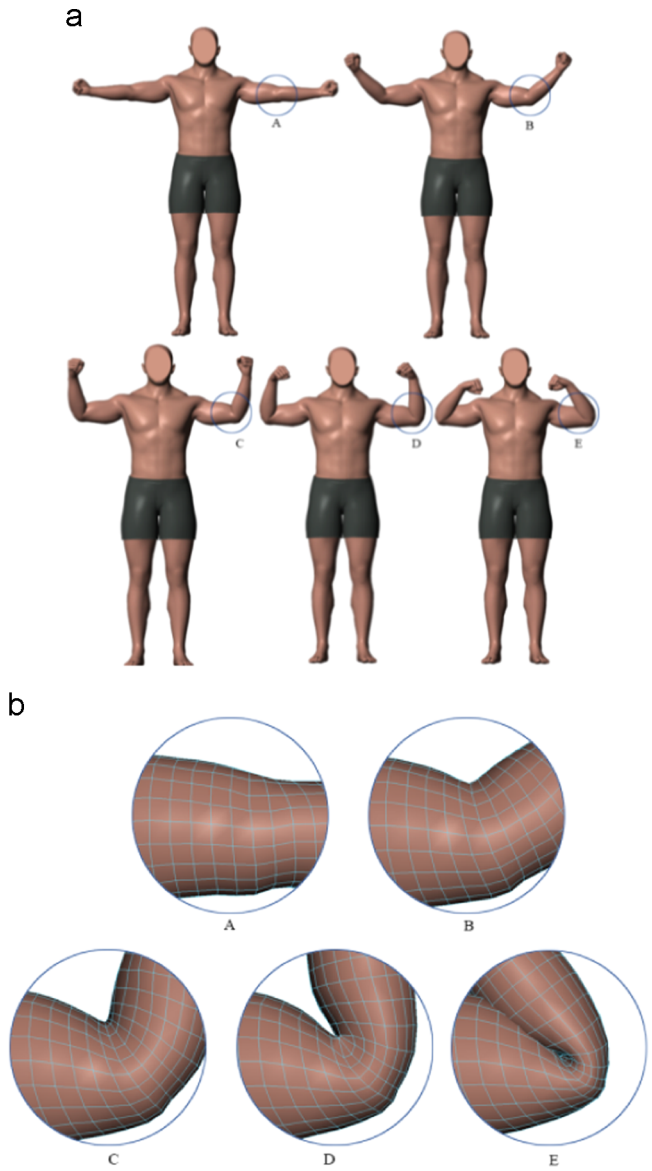

Fig. 4. Reconstructed skin deformation models.

In order to use the curve network to represent the deformations at the same surface positions of a polygon model but different poses, we use surface vertices to define the curve network, and obtain deformations of the curve network at different poses as explained below.

Transferring a polygon model into a curve network is achieved through three simple stages: dividing a polygon model into different parts; obtaining vertex indices of the curves to be extracted; and extracting the coordinate values of the curve vertices from the vertex indices.

First, we divide a polygon model into different parts by using manual operations or surface segmentation. In doing so, we divide the horse model shown in Fig. 6a into different parts highlighted with different colors in Fig. 6b.

Second, we develop a Maya Embedded Language (MEL) script to obtain the vertex indices. The process involves the following steps: (1) Manually specify a vertex in yellow which is the starting vertex of a curve to be extracted and an edge in red from the vertex (Fig. 5a). (2) Carry out Maya Select Edge Loop MEL command to extend the curve in black (Fig. 5b). (3) If the curve has passed its ending vertex in blue, select the edge in green before the ending vertex (Fig. 5b) and use Maya Stop Edge Loop Mel command to complete the extraction of the curve (Fig. 5c). (4) If a curve in black to be extracted goes in a wrong direction at the vertex in pink, select the correct edge in red (Fig. 5d) and conduct Maya Change Edge Loop Mel command to change the direction of the curve to the correct one (Fig. 5e). With these steps, we obtain the vertex indices of all curves from the vertices of the horse model indicated in Fig. 6c, and transform the polygon horse model in Fig. 6a into a curve network, i.e., a curve defined model demonstrated in Fig. 6d and e.

Third, after obtaining the vertex indices of all curves, we automatically extract the coordinate values of all curve vertices from the vertex indices at different poses by using our developed Maya Plug-in.

This method is efficient. The horse model in Fig. 6a has 30,134 vertices. Using manual operations to divide the horse model into parts, the total time used to extract all curves shown in Fig. 6d and e and obtain all coordinate values of the curve vertices of the horse model at different poses only takes two hours. With the same method, we extract 22 curves of a male human arm and depict them in Fig. 9.

3.3. Creating skin deformations from deformed curves

Before carrying out the dynamic deformation calculations described in Section 4 below, rigid-body transformations are applied to the polygon model and extracted curves to exclude the influences caused by geometric rotations and translations.

With the dynamic deformation model and its finite difference solution described in Section 4, we calculate the deformations of the extracted curves. These deformations are transferred back to the polygon model to achieve new skin shapes. Depending on whether the vertices of the polygon model are on the extracted curves or not, different methods will be used to determine their new positions.

If a vertex $\mathbf{v}$ of the polygon model is between the two vertices $\mathbf{p}_1$ and $\mathbf{p}_2$ of the curve, we can obtain the new position $\mathbf{v}'$ of the vertex $\mathbf{v}$ after the dynamic deformations through $\overline{\mathbf{p}'_1\mathbf{v}'}/\overline{\mathbf{p}'_1\mathbf{p}'_2} = \overline{\mathbf{p}_1\mathbf{v}}/\overline{\mathbf{p}_1\mathbf{p}_2}$ where $\mathbf{v}$, $\mathbf{p}_1$, and $\mathbf{p}_2$ are positions before the dynamic deformations, and $\mathbf{v}'$, $\mathbf{p}'_1$, and $\mathbf{p}'_2$ are positions after the dynamic deformations.

If a vertex $\mathbf{v}$ of the polygon model is among the four vertices $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$ and $\mathbf{p}_4$ of two adjacent extracted curves $\mathbf{c}_1$ and $\mathbf{c}_2$ where $\mathbf{p}_1$ and $\mathbf{p}_2$ are on $\mathbf{c}_1$ and $\mathbf{p}_3$ and $\mathbf{p}_4$ are on $\mathbf{c}_2$, the new position $\mathbf{v}'$ of the vertex $\mathbf{v}$ after the dynamic deformations is determined below.

First, we calculate the center $\mathbf{p}$ of the four vertices $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$ and $\mathbf{p}_4$, and obtain $\overline{\mathbf{pv}}$. After the dynamic deformations, the four vertices $\mathbf{p}_1$, $\mathbf{p}_2$, $\mathbf{p}_3$ and $\mathbf{p}_4$ move to new positions $\mathbf{p}'_1, \mathbf{p}'_2, \mathbf{p}'_3$ and $\mathbf{p}'_4$. The center of these four vertices is $\mathbf{p}'$. The new position $\mathbf{v}'$ of the vertex $\mathbf{v}$ after the dynamic deformations is obtained by superimposing $\overline{\mathbf{pv}}$ to $\mathbf{p}'$.

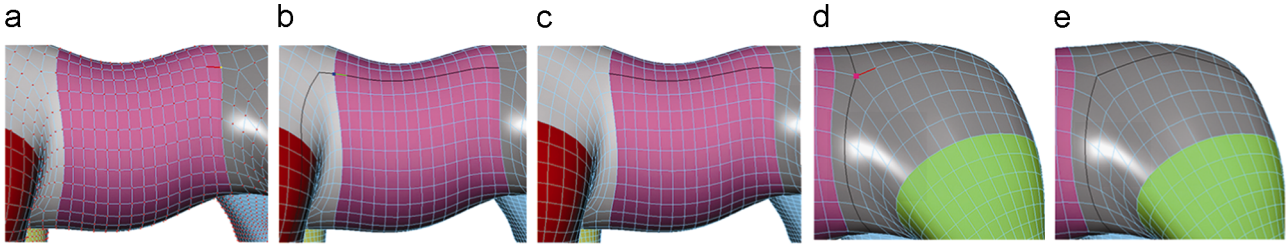

Fig. 5. Curve extraction with MEL commands.

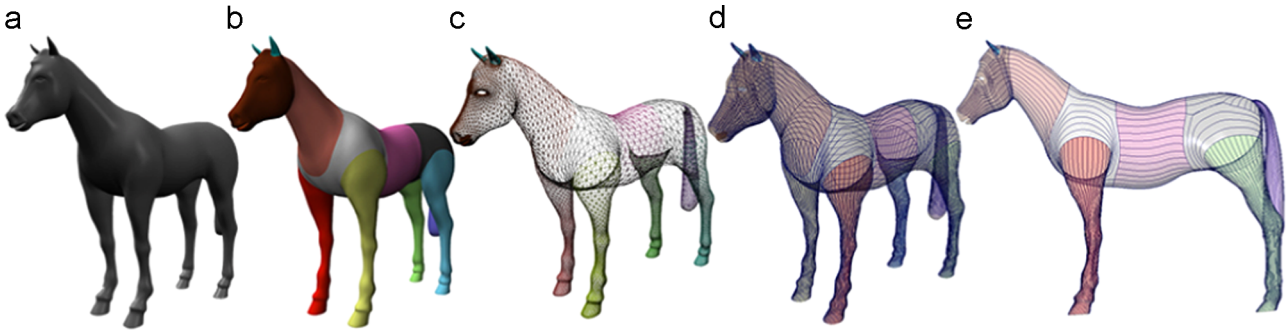

Fig. 6. Transforming a polygon horse model into a curve network.

With the above treatment, we can obtain new shapes of the polygon model after dynamic deformations.

4. Dynamic deformation model and finite difference solution

This section discusses how to combine dynamics of curve deformations with two example skin shapes to achieve dynamic skin deformations.

Dynamic skin deformations are determined by dynamic shape changes of the curves defining skin surfaces. Dynamic deformation of curves is similar to dynamic bending of homogeneous elastic beams. Therefore, we first develop the dynamic deformation model of curves consisting of Eqs. (2) and (6) below from the dynamic beam equation (Eq. (1)). The sculpting forces $F_q(v,\ t)$ embedded in Eq. (2) drive dynamic deformations of curves.

Then, we investigate how to transform the dynamic deformation model of curves into the finite difference equations. These finite difference equations can be used to tackle two different situations below.

When the sculpting forces are known, they are substituted into the finite difference equations, and the equations are solved to obtain curve deformations which are transferred to skin surfaces to achieve dynamic skin deformations.

When the sculpting forces are unknown, but example skin shapes at two different poses are known, a data-driven algorithm is proposed below to determine the unknown sculpting forces from the finite difference equations. The obtained sculpting forces are introduced back into the finite difference equations, and the equations are solved to obtain dynamic shape changes of curves which are used to determine dynamic skin deformations.

4.1. Dynamic deformation model

The dynamic bending equation of homogeneous elastic beams can be written as [48]

$$EI\frac{\partial^4 w}{\partial x^4}+\rho A\frac{\partial^2 w}{\partial t^2}=F(x) \tag{1}$$

where E is the Young's modulus, I is the second moment, ρ is the mass density, A is the cross-section area of the beam, $F(x)$ is a lateral load acting on the beam, w is the lateral deformation, x is a position variable in the beam direction, and t is the time variable.

The above equation has already been widely applied in engineering to achieve accurate predictions of dynamic beam bending. Curve deformations following the same physics should give a more realistic result.

For the parametric representation of a curve, there are three position components x, y and z. For dynamic deformations, each of them is the function of the parametric variable v and the time variable t. Accordingly, the function w in the above equation should be replaced by the three position components x, y and z, respectively, and the variable x in Eq. (1) should be replaced by the parametric variable v. Using the symbol $\mathbf{q}$ to represent a vector-valued position function which has three position components x, y and z ($\mathbf{q}=\begin{bmatrix}x & y & z\end{bmatrix}^T$), and a_1 and a_2 to represent EI and ρA, Eq. (1) is changed into the following differential equations:

$$a_1\frac{\partial^4 \mathbf{q}(v,\ t)}{\partial v^4}+a_2\frac{\partial^2 \mathbf{q}(v,\ t)}{\partial v^2}=\mathbf{f}(v,\ t) \tag{2}$$

Since a_1 and a_2 have a big impact on shape changes of deformed curves, they are called shape control parameters. The lateral load $\mathbf{f}(v,\ t)=\begin{bmatrix}f_x(v,\ t) & f_y(v,\ t) & f_z(v,\ t)\end{bmatrix}^T$ is called a vector-valued sculpting force.

Eq. (2) governs the physics-based deformations of a curve. It does not include any rigid-body transformations. When a 3D skin surface is deformed into a new one, it usually involves rigid-body transformations such as translations and rotations. These rigid-body transformations must be firstly removed before Eq. (2) is applied. This can be easily done by applying the operations of translation and rotation transformations.

In order to ensure two connected curves always maintain position and tangential continuities during their deformations described by Eq. (2), boundary conditions must be applied when solving Eq. (2). These boundary conditions can be determined below.

Position continuity requires two connected curves having the same deformation, and tangential continuity requires them to

have the same deformation rate at their joint position. For a curve, its joint positions are its two end points which are at $v=0$ and $v=1$. According to Eq. (2), the deformation is $\mathbf{q}(v, t)$ and the deformation rate in the length direction of a curve is $\partial\mathbf{q}(v, t)/\partial v$. If the deformation is $\mathbf{d}_0$ at $v=0$ and $\mathbf{d}_1$ at $v=1$, and the deformation rate is $\mathbf{s}_0$ at $v=0$ and $\mathbf{s}_1$ at $v=1$. At the final pose $t=1$ of a movement, the boundary conditions can be written as

$$t=1 \quad \mathbf{q}(0,\ 1)=\mathbf{d}_0 \left.\frac{\partial\mathbf{q}(v,1)}{\partial v}\right|_{v=0}=\mathbf{s}_0$$
$$\mathbf{q}(1,1)=\mathbf{d}_1 \left.\frac{\partial\mathbf{q}(v,\ 1)}{\partial v}\right|_{v=1}=\mathbf{s}_1 \tag{3}$$

The deformations $\mathbf{d}_0$ and $\mathbf{d}_1$ and the deformation rates $\mathbf{s}_0$ and $\mathbf{s}_1$ can be determined below.

Assuming that the position value of a curve at the initial pose is $\mathbf{p}_{00}$ at $v=0$ and $\mathbf{p}_{01}$ at $v=1$, and the position value of the curve at the final pose is $\mathbf{p}_{10}$ at $v=0$ and $\mathbf{p}_{11}$ at $v=1$, the deformation after excluding rigid-body transformations is $\mathbf{d}_0=\mathbf{p}_{10}-\overline{\mathbf{p}}_{00}$ at $v=0$ and $\mathbf{d}_1=\mathbf{p}_{11}-\overline{\mathbf{p}}_{01}$ at $v=1$ where $\overline{\mathbf{p}}_{00}$ and $\overline{\mathbf{p}}_{01}$ are respectively the position value of $\mathbf{p}_{00}$ and $\mathbf{p}_{01}$ after rigid-body transformations. With the same method, we can find the deformation $\overline{\mathbf{d}}_0$ of the node next to the node at $v=0$ and the deformation $\overline{\mathbf{d}}_1$ of the node next to the node at $v=1$. The deformation rates of the curve at $v=0$ and $v=1$ are determined by $\mathbf{s}_0=\left(\overline{\mathbf{d}}_0-\mathbf{d}_0\right)/h_0$ and $\mathbf{s}_1=\left(\mathbf{d}_1-\overline{\mathbf{d}}_1\right)/h_1$, respectively, where h_0 is the length between two adjacent nodes at $v=0$ and h_1 is the length between two adjacent nodes at $v=1$.

At the initial pose $t=0$, the deformations and deformation rates are zero, and the following initial conditions can be reached:

$$t=0 \quad \mathbf{q}(v,\ 0)=0 \quad \left.\frac{\partial\mathbf{q}(v,\ t)}{\partial t}\right|_{t=0}=0 \tag{4}$$

Eq. (4) means that the deformations and deformation rates at the two ends of a curve in the length direction at the initial pose are zero, i.e.,

$$v=0 \quad \mathbf{q}(0,\ 0)=0 \quad \left.\frac{\partial\mathbf{q}(v,\ 0)}{\partial v}\right|_{v=0}=0$$
$$v=1 \quad \mathbf{q}(1,\ 0)=0 \quad \left.\frac{\partial\mathbf{q}(v,\ 0)}{\partial v}\right|_{v=1}=0 \tag{5}$$

Using a linear relationship to describe how boundary conditions change against the time, we reach the following boundary conditions at any time instant t:

$$v=0 \quad \mathbf{q}(0,\ t)=t\mathbf{d}_0 \quad \left.\frac{\partial\mathbf{q}(v,\ t)}{\partial v}\right|_{v=0}=t\mathbf{s}_o$$
$$v=1 \quad \mathbf{q}(1,\ t)=t\mathbf{d}_1 \quad \left.\frac{\partial\mathbf{q}(v,\ t)}{\partial v}\right|_{v=1}=t\mathbf{s}_1 \tag{6}$$

After the above treatment, determination of dynamic curve deformations is transformed into solving the dynamic mathematical model of curves consisting of Eq. (2) and boundary conditions (6).

4.2. Finite difference equations

As demonstrated in Table 1, the implicit finite difference solution of the dynamic deformation model is very efficient.

In order to develop such a finite difference solution, we first transform the above dynamic deformation model into the finite difference equations.

According to the node distribution shown in Fig. 7 and the mathematical derivation given in Appendix A, we obtain the following finite difference equations in a matrix form:

$$[\mathbf{K}_{nm}]\{\mathbf{Q}_{nm}\}=\{\mathbf{F}_{nm}\} \tag{7}$$

where the column vector $\{\mathbf{Q}_{nm}\}$ consists of the unknown constants $\mathbf{q}_n^{m+1}$, the column vector $\{\mathbf{F}_{nm}\}$ is from the contributions of the sculpting forces $\mathbf{f}_n^{m+1}$ and the previous skin deformations $\mathbf{q}_n^{m-1}$ and $\mathbf{q}_n^m$, and the square matrix $[\mathbf{K}_{nm}]$ consists of the coefficients of the unknown constants $\mathbf{q}_n^{m+1}$ in Eq. (A5).

Setting $m=0$ in Eq. (7) and carrying out the mathematical derivation shown in Appendix A, the above equation is changed into the following form:

$$[\mathbf{K}_{n0}]\{\mathbf{Q}_{n0}\}=\{\mathbf{F}_{n0}\} \tag{8}$$

where the column vector $\{\mathbf{Q}_{n0}\}$ consists of the unknown constants $\mathbf{q}_n^1$, the column vector $\{\mathbf{F}_{n0}\}$ is from the contributions of the sculpting forces $\mathbf{f}_n^1$, $\mathbf{d}_0$, $\mathbf{d}_1$, $\mathbf{s}_0$ and $\mathbf{s}_1$, and the square matrix $[\mathbf{K}_{n0}]$ consists of the coefficients of the unknown constants $\mathbf{q}_n^1$ in Eq. (A7).

The finite difference equations (Eq. (8)) are a special form of the finite difference equations (Eq. (7)) at $t=0$ i.e., $m=0$.

Table 1
Time used to determine skin deformations.

Application examples	No. of curves	No. of nodes	Dynamic solution			Static solution
			Time (s) to get $[\mathbf{K}_{nm}]^{-1}$	Time (s) to get solution for 30 frames	Total Time (s) for 30 frames	Total time (s) for 30 frames
Human arm	22	34	0.00072	0.01548	0.01692	0.01546
Horse Leg1	32	68	0.00316	0.08854	0.09486	0.08638
Horse Leg2	32	68	0.00316	0.08854	0.09486	0.08638
Horse Leg3	32	66	0.00292	0.08361	0.08945	0.08460
Horse Leg4	32	66	0.00292	0.08361	0.08945	0.08460
Horse Neck	32	21	0.00046	0.00860	0.00952	0.00813
Horse Torso	32	17	0.00042	0.00556	0.00640	0.00590
Horse Tail	32	28	0.00050	0.01520	0.01620	0.01477

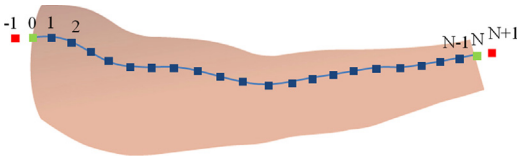

Fig. 7. Nodes on a curve.

4.3. Solution of finite difference equations

Depending on whether the sculpting forces are known or not, the finite difference equations (Eqs. (7) and (8)) can be solved with two different algorithms below.

When the sculpting forces $\mathbf{f}_n^m=\mathbf{f}(n/N,\ m/M)$ $(m=1,\ 2,\ \ldots,\ M-1;\quad n=1,\ 2,\ 3,\ \ldots,\ N-1)$ are known, we can substitute them into Eq. (8) and solve the equations to obtain all the unknown constants $\mathbf{q}_n^1$ $(n=1,\ 2,\ 3,\ \ldots,\ N-1)$.

Next, we set m to 1 in Eq. (7). Since $\mathbf{q}_n^{m-1}=\mathbf{q}_n^0$ and $\mathbf{q}_n^m=\mathbf{q}_n^1$ are known, we can use Eq. (7) to determine all the unknown constants $\mathbf{q}_n^2(n=1,\ 2,\ 3,\ \ldots,\ N-1)$. Repeating the process, all the unknown constants $\mathbf{q}_n^m$ $(m=0,\ 1,\ 2,\ \ldots,\ M;\quad n=1,\ 2,\ 3,\ \ldots,\ N-1)$ are determined to generate the deformed skin shapes at these poses.

When the sculpting forces $(m=1,\ 2,\ \ldots,\ M-1;n=1,\ 2,\ 3,\ \ldots,\ N-1)$ are unknown, but the skin shape $\mathbf{q}_n^M=\tilde{\mathbf{q}}_n$ at the final pose is known, we can use the following data-driven algorithm to determine the unknown sculpting forces and deformed skin shapes from two example skin shapes which are at the initial and final poses, respectively.

First, we set $M=1$ in Eq. (8). Since $\mathbf{q}_n^1=\mathbf{q}_n^M$ ($n=1, 2, 3, \ldots, N-1$), we can use Eq. (8) to determine $\mathbf{f}_n$ ($n=1, 2, 3, \ldots, N-1$). After $\mathbf{f}_n$ have been obtained, the sculpting forces at the time t_m are obtained by

$$\mathbf{f}_n^m=\frac{m}{M}\mathbf{f}_n \quad (m=1, 2, \ldots, M-1; \quad n=1, 2, \ldots, N-1) \tag{9}$$

Substituting Eq. (9) into Eqs. (8) and (7), we use it to obtain all the unknown constants $\mathbf{q}_n^m$ ($m=1, 2, \ldots, M-1$; $n=1, 2, 3, \ldots, N-1$).

If the obtained $\mathbf{q}_n^M$ is different from the known skin deformations $\tilde{\mathbf{q}}_n$ at the final pose, we assume the sculpting forces corresponding to the known skin deformations are $\tilde{\mathbf{f}}_n$. Then we use

$$\frac{\tilde{\mathbf{f}}_n}{\tilde{\mathbf{q}}_n}=\frac{\mathbf{f}_n}{\mathbf{q}_n^M} \tag{10}$$

to obtain the new sculpting forces $\tilde{\mathbf{f}}_n$

$$\tilde{\mathbf{f}}_n=\frac{\tilde{\mathbf{q}}_n}{\mathbf{q}_n^M}\mathbf{f}_n \tag{11}$$

Using the new sculpting forces $\tilde{\mathbf{f}}_n$ to replace $\mathbf{f}_n$ in Eq. (9), and introducing the updated $\mathbf{f}_n^m$ into Eqs. (8) and (7) to determine the new skin deformation $\mathbf{q}_n^m$ ($m=1, 2, \ldots, M$; $n=1, 2, \ldots, N-2, N-1$). The iteration is carried out until the required accuracy is reached.

The deformed skin shapes at different poses are obtained by superimposing the calculated skin deformations to the skin shapes at the initial pose after rigid-body transformations. In the following two sections, we will make a comparison among various skin deformation techniques and give two examples to demonstrate the applications of our proposed technique.

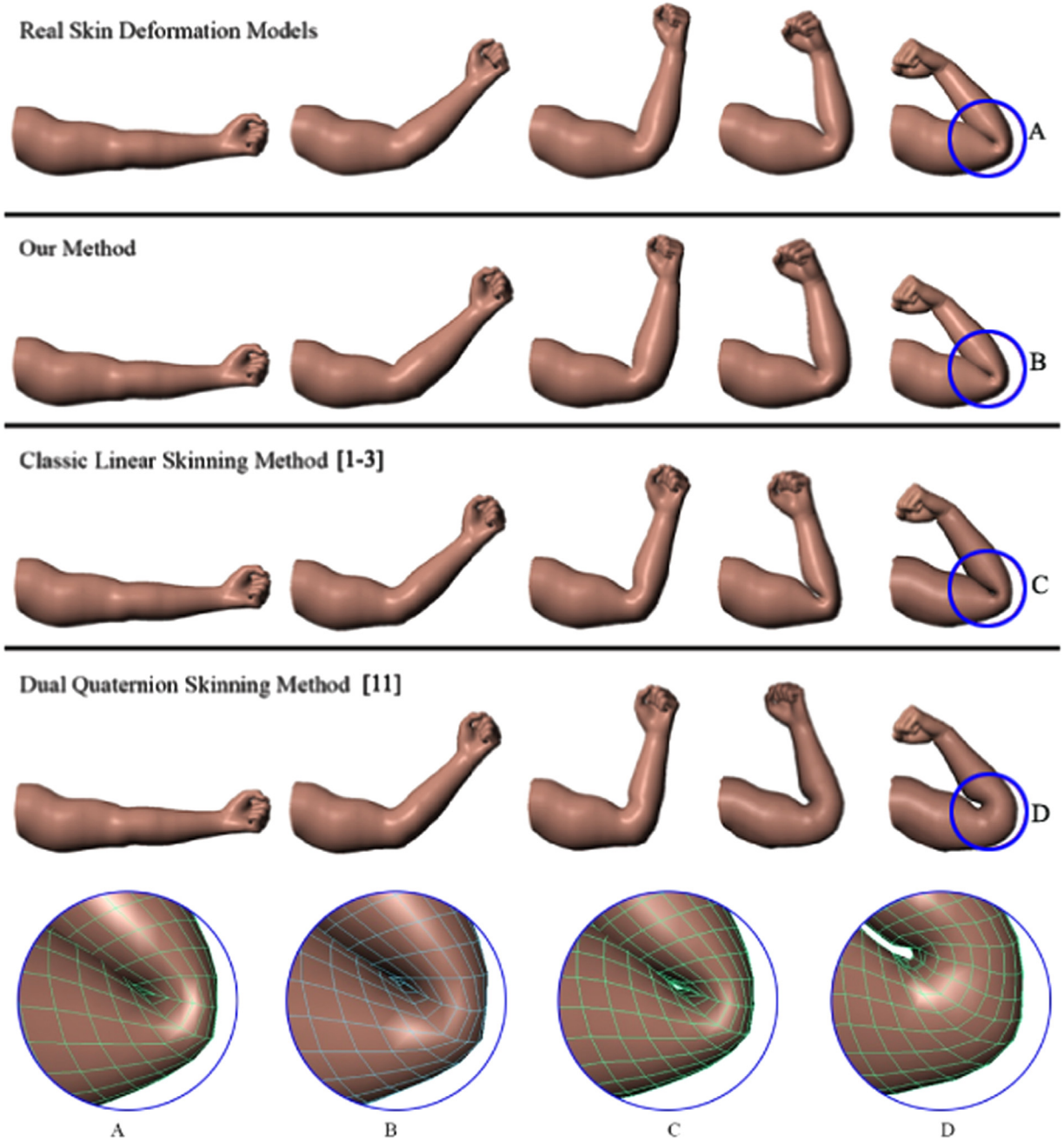

Fig. 8. Comparison of deformed skin shapes obtained with different techniques.

5. Experimental comparisons

In this section, we make three experimental comparisons: (1) among real models and those from different approaches, (2) between the dynamic finite difference solution and the static finite difference solution, and (3) between our proposed dynamic deformation model and the Smooth Skinning Decomposition with Rigid Bones.

5.1. Comparison among real models and those from different approaches

The male human arm models shown in Fig. 3 give the real deformed skin shapes of the male human arm movement. In order to facilitate the comparison among various skin deformation techniques, they are given in first row of Fig. 8, and the corresponding animation is shown in Video S1.

Supplementary data associated with this article can be found in the online version at http://dx.doi.org/10.1016/j.cag.2014.09.029.

Taking the first and fifth models given in the first row of the figure as the example skin shapes at the initial and final poses and using our proposed technique, the deformed skin shapes at the second, third and fourth poses are obtained and depicted in the second row of the figure, and the corresponding animation of skin deformations is shown in Video S1. It can be seen that the deformed skin shapes obtained with our proposed solution are very close to the real ones, indicating our proposed technique generates realistic skin deformations.

Taking the real arm model shown in the first row and first column of the figure as the skin shape at the initial pose, two different techniques available in the Autodesk Maya are applied to create the skin shapes at the other poses. These two techniques are Classic Linear Skinning Method [1–3] and Dual Quaternion Skinning Method [11]. The created skin shapes are depicted in the third and fourth rows where the third row is from the Classic Linear Skinning Method and the fourth row is from the Dual Quaternion Skinning Method. Comparing these skin deformation models with the real ones and our calculated ones, it is clear that our proposed skin deformation technique generate more realistic skin deformation shapes than the two skin deformation techniques available in the Autodesk Maya.

Although the deformed skin shape at the final pose created by the Dual Quaternion Skinning Method is less realistic compared to the real one, this method avoids the artefact of collapsing joint as indicated by the deformed skin shapes at the third and fourth poses of the fourth row. In contrast, the Classic Linear Skinning Method suffers from the artefact of collapsing joint as indicated by the deformed skin shape at the fourth pose of the third row.

In addition to its modeling capacity in creating realistic skin deformations, the technique proposed in this paper is also very efficient as demonstrated below.

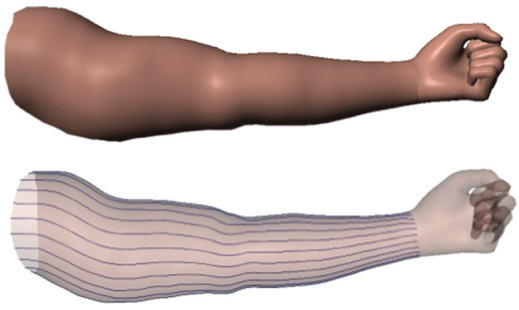

Fig. 9. Comparison between the real model and the one from the curve-based representation.

For $m=0$, the square matrix $[\mathbf{K}_{n0}]$ is determined by Eq. (8) due to the contributions of the previous skin deformations $\mathbf{q}_n^{-1}=\mathbf{q}_n^1$. For all other values of m, the previous skin deformations $\mathbf{q}_n^{m-1}$ and $\mathbf{q}_n^m$ have no contributions to the square matrix $[\mathbf{K}_{nm}]$. If a_1, a_2, M and N for all the curves and iterations are the same, the square matrix $[\mathbf{K}_{nm}]$ for all the values of $m>0$ is the same. That is to say, we only require computing the inverse matrix $[\mathbf{K}_{nm}]^{-1}$ once. In total, only two inverse matrices are calculated: one for $m=0$ and the other for all the values of $m>0$. Then, all the skin deformations can be quickly obtained by carrying out the following calculations:

$$\{\mathbf{Q}_{nm}\}=[\mathbf{K}_{nm}]^{-1}\{\mathbf{F}_{nm}\} \quad (m=0,\ 1,\ 2,\ \ldots,\ M-1) \tag{12}$$

For the skin deformations of the male human arm movement, we use 22 curves to describe the arm model. On each curve, 34 points are uniformly collocated. In total, the male human arm model is described by 748 points. In contrast, the original polygon model has 1390 vertices which are far more than the points of the curve-based representation. Both the real arm model at the initial pose and the corresponding model described by the curve-based representation are shown in Fig. 9.

Comparing both models, we cannot find any visible differences, indicating that the curve-based representation can describe a model of the same details with a much smaller data size than the polygon-based representation.

In Table 1, we give basic data using our proposed technique to determine skin deformations and the time used to obtain the inverse matrix and determine skin deformations. The Gauss–Jordan elimination is used to find the inverse matrix $[\mathbf{K}_{nm}]^{-1}$. All the calculations are carried out on a Hewlett–Packard HP Z420 Workstation with 3.2 GHz Intel(R) CPU E5-1650 32 GB of memory. As shown in the table, the total time used to obtain deformed skin shapes of 30 frames is 0.01692 s. It indicates our proposed technique is efficient enough to generate real-time skin deformations for character animation.

5.2. Comparison between dynamic and static finite difference solutions

Our proposed dynamic deformation model has a bigger coverage than the static deformation model and creates more realistic skin deformations.

When setting $a_2=0$ in Eq. (2), the dynamic deformation model becomes static one, indicating the former covers the latter. Using the finite difference method to solve the static deformation model, we obtain deformed skin shapes at the second, third and fourth poses and depicted them in the third row of Fig. 10 where the first row is from real skin deformations, and the second row is from the dynamic deformation model. Comparing the real skin deformation models with those from the dynamic and static deformation models, we can conclude that the dynamic deformation model give more realistic skin deformations than the static deformation model.

The efficiency comparison between the dynamic finite difference solution and the static finite difference solution is given in Table 1. It can be seen that the total time for 30 frames required by the former is very close to the latter although it is a little bigger.

5.3. Comparison between the dynamic deformation model and DDSR model

The Smooth Skinning Decomposition with Rigid Bones (SSDR) proposed in Ref. [9] is a state-of-art approach. Here we compare our approach with SSDR in terms of both error metric and computational efficiency.

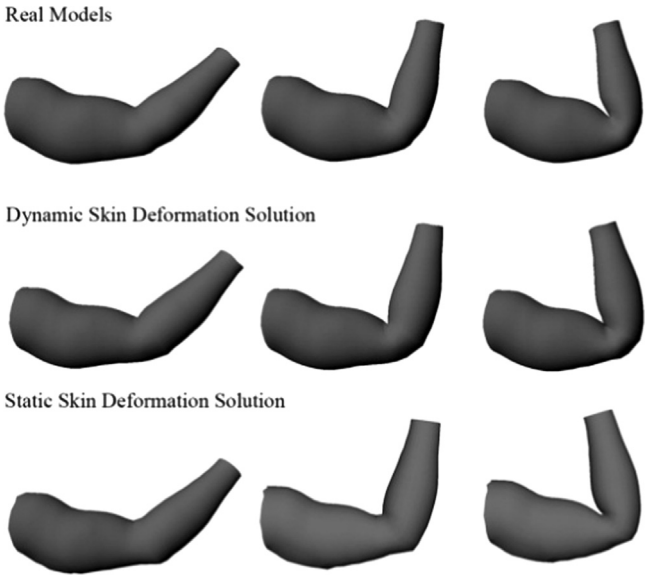

Fig. 10. Comparison among real deformed shapes and those from dynamic and static deformation models.

First, we describe how our proposed approach can be modified to determine dynamic skin deformations with $J+1$ ($J>1$) example skin shapes.

If $J+1$ example skin shapes at the poses $j=0, 1, 2, \ldots, J-1, J$, i.e. at the time $t_j = 1/J$ ($j=0, 1, 2, 3, \ldots, J-1, J$) are known, we can determine the deformations and deformation rates at the poses $j=1, 2, \ldots, J-1, J$ with the following algorithm.

Assuming that the position value of a curve at the pose j is $\mathbf{p}_0^j$ at $v=0$ and $\mathbf{p}_1^j$ at $v=1$, and the position value of the curve at the pose $j+1$ is $\mathbf{p}_0^{j+1}$ at $v=0$ and $\mathbf{p}_1^{j+1}$ at $v=1$, the deformation at the pose $j+1$ after excluding rigid-body transformations is $\mathbf{d}_{q0}^{j+1} = \overline{\mathbf{p}}_{q0}^{j+1} - \overline{\mathbf{p}}_{q0}^{j}$ at $v=0$ and $\mathbf{d}_1^{j+1} = \overline{\mathbf{p}}_1^{j+1} - \overline{\mathbf{p}}_1^{j}$ at $v=1$ where $\overline{\mathbf{p}}_0^j$, $\overline{\mathbf{p}}_0^{j+1}$, $\overline{\mathbf{p}}_1^j$ and $\overline{\mathbf{p}}_1^{j+1}$ are respectively the position value of $\mathbf{p}_0^j, \mathbf{p}_0^{j+1}, \mathbf{p}_1^j$ and $\mathbf{p}_1^{j+1}$ after rigid-body transformations. With the same method, we can find the deformation $\overline{\mathbf{d}}_0^{j+1}$ of the node next to the node at $v=0$ and the deformation $\overline{\mathbf{d}}_1^{j+1}$ of the node next to the node at $v=1$. The deformation rates of the curve at $v=0$ and $v=1$ are determined by $\mathbf{s}_0^{j+1} = \left(\overline{\mathbf{d}}_0^{j+1} - \mathbf{d}_0^{j+1}\right)/h_0^{j+1}$ and $\mathbf{s}_1^{j+1} = \left(\overline{\mathbf{d}}_1^{j+1} - \mathbf{d}_1^{j+1}\right)/h_1^{j+1}$, respectively, where h_0^{j+1} is the length between two adjacent nodes at $v=0$ and h_1^{j+1} is the length between two adjacent nodes at $v=1$ at the pose $j+1$.

Using a linear relationship to describe how boundary conditions change against the time, we reach the following boundary conditions at any instant $t_j \le t \le t_{j+1}$:

$$v=0 \quad \mathbf{q}(0, t) = \frac{t-t_j}{t_{j+1}-t_j}\mathbf{d}_0^{j+1}$$

$$\frac{\partial \mathbf{q}(0, t)}{\partial t} = \frac{t-t_j}{t_{j+1}-t_j}\mathbf{s}_0^{j+1}$$

$$v=1 \quad \mathbf{q}(1, t) = \frac{t-t_j}{t_{j+1}-t_j}\mathbf{d}_1^{j+1}$$

$$\frac{\partial \mathbf{q}(1, t)}{\partial t} = \frac{t-t_j}{t_{j+1}-t_j}\mathbf{s}_1^{j+1}$$

$$(t_j \le t \le t_{j+1}) \tag{13}$$

Since the skin shape $\mathbf{q}_n^{j+1}$ at pose j ($j=-1, 0, 1, 2, \ldots, J-1$; $n=0, 1, 2, \ldots, N$) is known, we substitute $\mathbf{q}_n^{j+1}$ into (A3) and obtain the following equation:

$$a_1 N^4 \left[6\mathbf{q}_n^{j+1} - 4(\mathbf{q}_{n-1}^{j+1} + \mathbf{q}_{n+1}^{j+1})\right] + \mathbf{q}_{n-2}^{j+1} + \mathbf{q}_{n+2}^{j+1}$$

$$+ a_2 M^2 (\mathbf{q}_n^{j+1} - 2\mathbf{q}_n^{j} + \mathbf{q}_n^{j-1}) = \mathbf{f}(v_n, t_{j+1})$$

$$(j=0, 1, 2, \ldots, J-1; \quad n=1, 2, \ldots, N-1) \tag{14}$$

Considering Eq. (A6), we can solve Eq. (14) subjected to boundary conditions (13) to obtain the sculpting forces at all poses $\mathbf{f}(v_n, t_{j+1})$. Then we can use the sculpting forces, Eqs. (14) and (13) to calculate skin deformations between all two adjacent ones of the example poses.

The error metric $E_{RMS} = 1000\sqrt{E/(3.|V|.|t|)}$ has been used in Ref. [9] to quantitatively evaluate the SSDR model where E is the sum of the squared errors between the example skin shapes and those determined by the SSDR model at the example skin poses. For the horse-gallop with 49 example skin shapes [49] used in Ref. [9], the approximate error generated by the SSDR model with 10 bones ($|B|=10$) is $E_{RMS}=4.6$. In comparison, our approach gives the same skin deformations as the example skin shapes at the example skin poses, leading to zero error metric $E_{RMS}=0$.

For the same horse-gallop, the SSDR model takes 2.1 min on a computer with a 2 GHz single core CPU to obtain the weights and the bone transformations which are used to calculate new skin shapes. On a computer with a 3.2 GHz single core CPU, our approach takes 0.045 s to determine all the inverse matrices $[\mathbf{K}_{nm}]^{-1}$ of all the curves defining the horse model which are also used to calculate new skin shapes. The time using all obtained inverse matrices $[\mathbf{K}_{nm}]^{-1}$ to get 30 frames is 0.2568 s. Putting the two numbers together, we obtain the total time of getting 30 frames which is 0.3018 s.

6. Applications

In this section, we give two examples to demonstrate how to use our proposed technique to create skin deformations.

The first example is to create twisting deformations with our proposed technique, and examine whether the candy-wrapper anomaly arises. We build an original arm model at the time $t=0$ (pose 0) shown in the top row of the images in Fig. 11 and one adjacent twisted shape at the time $t=0.125$ (pose 1) which is twisted a little downwards, and depict it in the middle row of the images in Fig. 11. Solving Eq. (8) with $M=8$, we obtain the twisting forces $\mathbf{f}_n^1$ at the pose 1. Substituting the twisting forces $\mathbf{f}_n^j = j\mathbf{f}_n^1$ ($j=2, 3, \ldots, M$) into Eq. (7) with $M=8$ and $m=1, 2, \ldots, M-1$, we obtain the twisting deformations at the pose 2 to pose 8, and depict the twisting deformations at pose 8 in the bottom row of the images in Fig. 11. The image and Video S1 indicate that no candy-wrapper collapse problem is caused by our proposed approach.

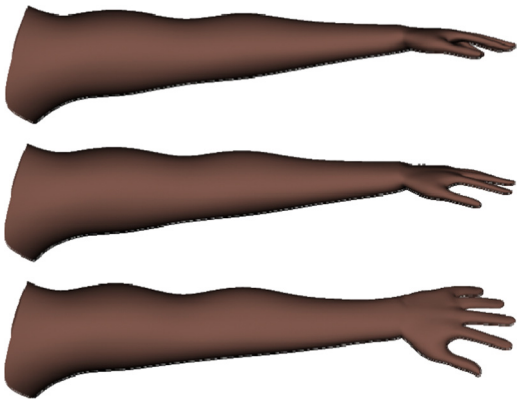

Fig. 11. Twisting deformations of a human arm.

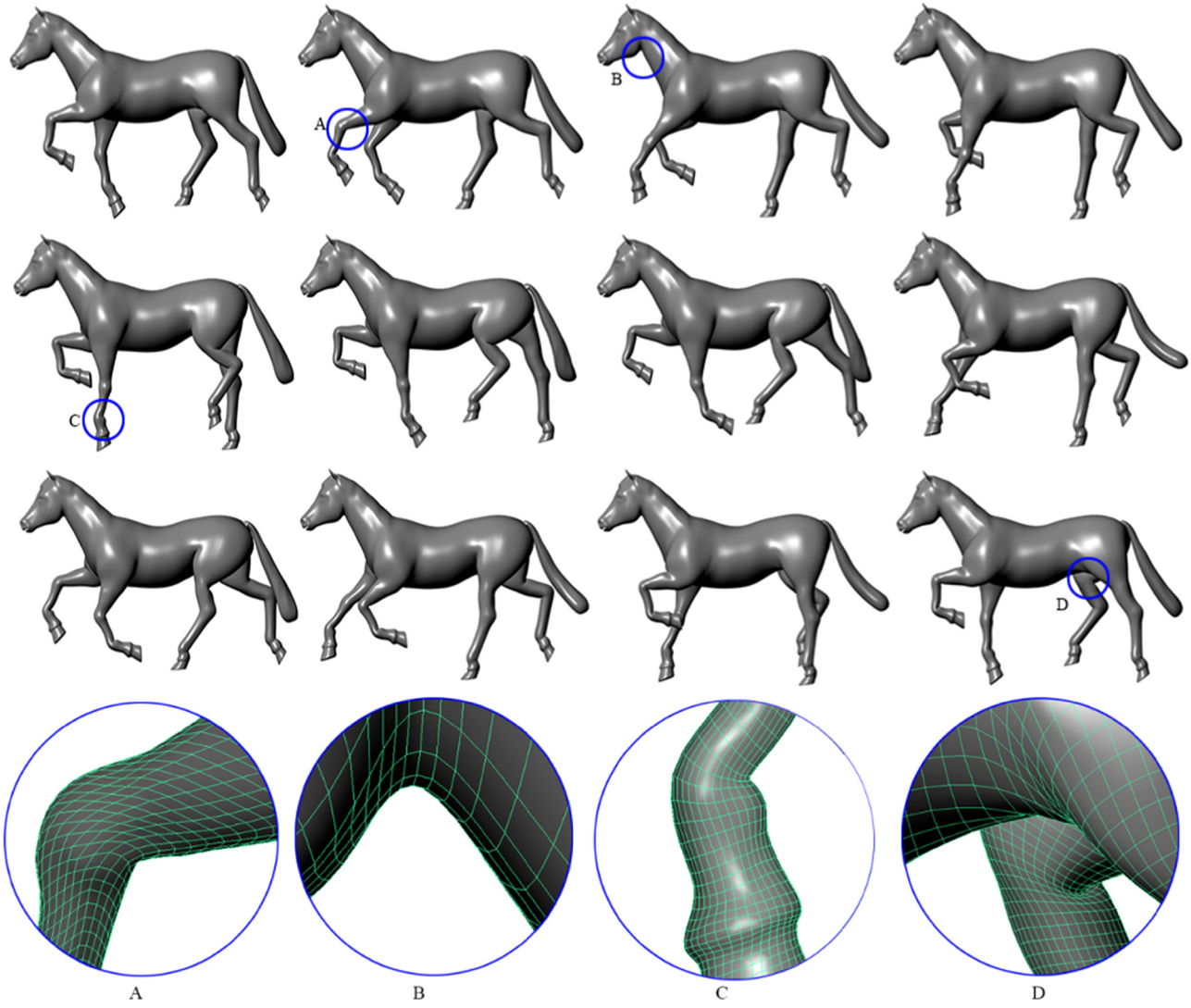

Fig. 12. Skin deformations of horse running.

The second example is to create skin deformations of horse running. As shown in Fig. 6, we extract all curves of a horse model. The information about the curves and the points on the curves is given in Table 1. With our proposed curve-based representation, the whole horse model is defined by 11,641 points in total. In contrast, the original polygon horse model has 30,134 vertices. Once again, the curve-based representation uses much fewer data to describe a same horse model.

With our proposed technique, we calculate skin deformations of a running horse and depict some of the obtained deformed models in Fig. 12, and give the animation of the horse running in Video S1. These images and the animation also demonstrate that our proposed technique produces realistic skin deformations of horse running. The time used to obtain deformed skin shapes of 30 frames for horse legs, neck, torso, and tail is also given in Table 1. Once again, these time data indicate high efficiency of our proposed technique.

7. Conclusions and future work

We have presented a new skin deformation technique in this paper. This new technique is based on a new model of dynamic deformations and an efficient finite difference solution of the model. Its combination with data-driven methods and curve-based representation of 3D models brings a number of advantages, including realistic skin deformation creation, high computing efficiency, and small data size.

Realism of skin surfaces can be greatly enhanced by taking into account skin wrinkles. We will investigate this issue by developing a curve-based dynamic winkle model.

How muscle and other tissues affect skin deformations has not been investigated in this paper. In our future work, we intend to incorporate these effects and develop a more realistic skin deformation model to address this issue.

Acknowledgments

This research is supported by the grant of 2013 International Exchanges Scheme (Grant no. IE131367), the Royal Society, United Kingdom; the National Natural Science Foundation of China (Grant no. 51475394). Xiaogang Jin was supported by the National Natural Science Foundation of China (Grant nos. 61272298, 61472351), China.

Appendix A. Derivation of finite difference equations

As shown in Fig. 7, we divide the domain $0 \le v \le 1$ into N equal intervals. The parametric value between any two adjacent nodes is $\Delta v = 1/N$, and the parametric value of the n^{th} node is $v_n = n/N$ ($n = 0, 1, 2, \ldots, N$). Similarly, we divide the time range $0 \le t \le 1$ into M equal time intervals. The time increment is $\Delta t = 1/M$, and the time variable takes the value of $t_m = m/M$ ($m = 0, 1, 2, \ldots, M$) at the m^{th} time step. The derivatives of $\mathbf{q}(v, t)$ with respect to the parametric variable v and time variable t at the $(m+1)^{\text{th}}$ time step and the n^{th} node can be written as

$$\left[\frac{\partial \mathbf{q}(v,t)}{\partial t}\right]_n^{m+1} = \frac{\mathbf{q}_n^{m+1} - \mathbf{q}_n^{m-1}}{2\Delta t}$$
$$\left[\frac{\partial \mathbf{q}^2(v,t)}{\partial t^2}\right]_n^{m+1} = \frac{\mathbf{q}_n^{m+1} - 2\mathbf{q}_n^{m} + \mathbf{q}_n^{m-1}}{\Delta t^2}$$
$$\left[\frac{\partial \mathbf{q}(v,t)}{\partial v}\right]_n^{m+1} = \frac{\mathbf{q}_{n+1}^{m+1} - \mathbf{q}_{n-1}^{m+1}}{2\Delta v}$$
$$\left[\frac{\partial^4 \mathbf{q}(v,t)}{\partial v^4}\right]_n^{m+1} = \frac{1}{\Delta v^4}\left[6\mathbf{q}_n^{m+1} - 4\mathbf{q}_{n-1}^{m+1} + \mathbf{q}_{n+1}^{m+1}\right) + \mathbf{q}_{n-2}^{m+1} + \mathbf{q}_{n+2}^{m+1}\right] \quad \text{(A1)}$$

Substituting $\Delta t = 1/M$ and $\Delta v = 1/N$ into the above equations, we have

$$\left[\frac{\partial \mathbf{q}(v,t)}{\partial t}\right]_n^{m+1} = \frac{M}{2}\left[\mathbf{q}_n^{m+1} - \mathbf{q}_n^{m-1}\right]$$
$$\left[\frac{\partial \mathbf{q}^2(v,t)}{\partial t^2}\right]_n^{m+1} = M^2(\mathbf{q}_n^{m+1} - 2\mathbf{q}_n^{m} + \mathbf{q}_n^{m-1})$$
$$\left[\frac{\partial \mathbf{q}(v,t)}{\partial v}\right]_n^{m+1} = \frac{N}{2}(\mathbf{q}_{n+1}^{m+1} - \mathbf{q}_{n-1}^{m+1})$$
$$\left[\frac{\partial^4 \mathbf{q}(v,t)}{\partial v^4}\right]_n^{m+1} = N^4\left[6\mathbf{q}_n^{m+1} - 4\mathbf{q}_{n-1}^{m+1} + \mathbf{q}_{n+1}^{m+1}\right) + \mathbf{q}_{n-2}^{m+1} + \mathbf{q}_{n+2}^{m+1}\right] \quad \text{(A2)}$$

Substituting Eq. (A2) into Eq. (2), the differential equations (Eq. (2)) becomes

$$a_1N^4\left[6\mathbf{q}_n^{m+1} - 4(\mathbf{q}_{n-1}^{m+1} + \mathbf{q}_{n+1}^{m+1}) + \mathbf{q}_{n-2}^{m+1} + \mathbf{q}_{n+2}^{m+1}\right] + a_2M^2(\mathbf{q}_n^{m+1} - 2\mathbf{q}_n^{m} + \mathbf{q}_n^{m-1}) = \mathbf{f}(v_n, t_{m+1})$$
$$(m = 0, 1, 2, \ldots, M-1; \quad n = 1, 2, \ldots, N-1) \quad \text{(A3)}$$

If $\mathbf{q}_n^{m-1}$ and $\mathbf{q}_n^{m}$ and are known, the total unknown constants $\mathbf{q}_n^{m+1}$ in Eq. (A3) are $N+3$ since the green boundary nodes $n=1$ and $n=0$ and the red virtual nodes $n=-1$ and $n=N+1$ will be involved when formulating the finite difference equation at the blue internal nodes $n=1$ and $n=N-1$ (Fig. 7). Since Eq. (A3) only includes $N-1$ linear algebra equations, we require four additional linear equations to determine these $N+3$ unknown constants. These four additional equations can be obtained by considering boundary conditions (6).

Substituting Eq. (A2) into Eq. (6), the following boundary conditions are obtained:

$$\mathbf{q}_0^{m+1} = \tfrac{m+1}{M}\mathbf{d}_0 \quad \tfrac{N}{2}(\mathbf{q}_1^{m+1} - \mathbf{q}_{-1}^{m+1}) = \tfrac{m+1}{M}\mathbf{s}_0$$
$$\mathbf{q}_N^{m+1} = \frac{m+1}{M}\mathbf{d}_1 \quad \frac{N}{2}(\mathbf{q}_{N+1}^{m+1} - \mathbf{q}_{N-1}^{m+1}) = \frac{m+1}{M}\mathbf{s}_1 \quad \text{(A4)}$$

Introducing Eq. (A4) into Eq. (A3), and denoting $\mathbf{f}_n^m = \mathbf{f}(n/N, m/M)$, the unknown constants in Eq. (A3) are reduced to $N-1$ and determined by $N-1$ equations below

$$(6a_1N^4 + a_2M^2)\mathbf{q}_n^{m+1} + a_1N^4\left[-4(\mathbf{q}_{n-1}^{m+1} + \mathbf{q}_{n+1}^{m+1}) + \mathbf{q}_{n-2}^{m+1} + \mathbf{q}_{n+2}^{m+1}\right] + a_2M^2(-2\mathbf{q}_n^{m} + \mathbf{q}_n^{m-1}) = \mathbf{f}_n^{m+1}$$
$$(m = 0, 1, 2, \ldots, M-1; \quad 2 < n < N-2)$$
$$(6a_1N^4 + a_2M^2)\mathbf{q}_1^{m+1} + a_1N^4\left[-4\left(\frac{m+1}{M}\mathbf{d}_0 + \mathbf{q}_2^{m+1}\right) + \mathbf{q}_1^{m+1} - \frac{2(m+1)}{MN}\mathbf{s}_0 + \mathbf{q}_3^{m+1}\right] + a_2M^2(-2\mathbf{q}_1^{m} + \mathbf{q}_1^{m-1}) = \mathbf{f}_1^{m+1}$$
$$(m = 0, 1, 2, \ldots, M-1; \quad n = 1)$$
$$(6a_1N^4 + a_2M^2)\mathbf{q}_2^{m+1} + a_1N^4\left[-4(\mathbf{q}_1^{m+1} + \mathbf{q}_3^{m+1}) + \frac{m+1}{M}\mathbf{d}_0 + \mathbf{q}_4^{m+1}\right] + a_2M^2(-2\mathbf{q}_2^{m} + \mathbf{q}_2^{m-1}) = \mathbf{f}_2^{m+1}$$
$$(m = 0, 1, 2, \ldots, M-1; \quad n = 2)$$
$$(6a_1N^4 + a_2M^2)\mathbf{q}_{N-2}^{m+1} + a_1N^4\left[-4(\mathbf{q}_{N-3}^{m+1} + \mathbf{q}_{N-1}^{m+1}) + \frac{m+1}{M}\mathbf{d}_1 + \mathbf{q}_{N-4}^{m+1}\right] + a_2M^2(-2\mathbf{q}_{N-2}^{m} + \mathbf{q}_{N-2}^{m-1}) = \mathbf{f}_{N-2}^{m+1}$$
$$(m = 0, 1, 2, \ldots, M-1; \quad n = N-2)$$
$$(6a_1N^4 + a_2M^2)\mathbf{q}_{N-1}^{m+1} + a_1N^4\left[-4\left(\frac{m+1}{M}\mathbf{d}_1 + \mathbf{q}_{N-2}^{m+1}\right) + \mathbf{q}_{N-3}^{m+1} - \frac{2(m+1)}{MN}\mathbf{s}_1 + \mathbf{q}_{N-1}^{m+1}\right] + a_2M^2(-2\mathbf{q}_{N-1}^{m} + \mathbf{q}_{N-1}^{m-1}) = \mathbf{f}_{N-1}^{m+1}$$
$$(m = 0, 1, 2, \ldots, M-1; \quad n = N-1) \quad \text{(A5)}$$

Writing the above finite difference equations into a matrix form, Eq. (7) is obtained.

Since the iterative calculations start from $t=0$, $m=0$ and $\mathbf{q}_n^{m-1}$ and $\mathbf{q}_n^{m}$ become $\mathbf{q}_n^{-1}$ and $\mathbf{q}_n^{0}$. They are determined through Eq. (A6) obtained by substituting Eq. (A2) into Eq. (4):

$$\mathbf{q}_n^0 = 0 \quad \mathbf{q}_n^{-1} = \mathbf{q}_n^1 \quad (n = 0, 1, 2, \ldots, N) \quad \text{(A6)}$$

Considering Eq. (A6), Eq. (A5) becomes

$$(6a_1N^4 + 2a_2M^2)\mathbf{q}_n^1 + a_1N^4\left[-4(\mathbf{q}_{n-1}^1 + \mathbf{q}_{n+1}^1) + \mathbf{q}_{n-2}^1 + \mathbf{q}_{n+2}^1\right] = \mathbf{f}_n^1$$
$$(m = 0; \quad 2 < n < N-2)$$
$$(6a_1N^4 + 2a_2M^2)\mathbf{q}_1^1 + a_1N^4\left[-4\left(\frac{1}{M}\mathbf{d}_0 + \mathbf{q}_2^1\right) + \mathbf{q}_1^1 - \frac{2}{MN}\mathbf{s}_0 + \mathbf{q}_3^1\right] = \mathbf{f}_1^1$$
$$(m = 0; \quad n = 1)$$
$$(6a_1N^4 + 2a_2M^2)\mathbf{q}_2^1 + a_1N^4\left[-4(\mathbf{q}_1^1 + \mathbf{q}_3^1) + \frac{1}{M}\mathbf{d}_0 + \mathbf{q}_4^1\right] = \mathbf{f}_2^1$$
$$(m = 0; \quad n = 2)$$
$$(6a_1N^4 + 2a_2M^2)\mathbf{q}_{N-2}^1 + a_1N^4\left[-4(\mathbf{q}_{N-3}^1 + \mathbf{q}_{N-1}^1) + \frac{1}{M}\mathbf{d}_1 + \mathbf{q}_{N-4}^1\right] = \mathbf{f}_{N-2}^1$$
$$(m = 0; \quad n = N-2)$$
$$(6a_1N^4 + 2a_2M^2)\mathbf{q}_{N-1}^1 + a_1N^4\left[-4\left(\frac{1}{M}\mathbf{d}_1 + \mathbf{q}_{N-2}^1\right) + \mathbf{q}_{N-3}^1 + \frac{2}{MN}\mathbf{s}_1 + \mathbf{q}_{N-1}^1\right] = \mathbf{f}_{N-1}^1 (m = 0; \quad n = N-1) \quad \text{(A7)}$$

Similarly, the finite difference equations (Eq. (A7)) can be written as a matrix equation (Eq. (8)).

References

[1] Thalmann NM, Laperriére R, Thalmann D. Joint-dependent local deformations for hand animation and object grasping. In: Proceedings on CIPSG; 1988. p. 26–33.

[2] Lander J. Skin them bones: game programming for the web generation. Game Dev Mag 1998:11–6.

[3] Lander J. Over my dead, polygonal body. Game Dev Mag 1999:1–4.

[4] Weber J. Run-time skin deformation. In: Proceedings of the game developers conference; 2000.

[5] Wang XC, Phillips C. Multi-weight enveloping: least-squares approximation techniques for skin animation. In: Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation; 2002. p. 129–38.

[6] Mohr A, Gleicher M. Building efficient, accurate character skins from examples. ACM Trans Graph 2003;22(3):562–8.

[7] Yang XS, Somasekharan A, Zhang JJ. Curve skeleton skinning for human and creature characters: research articles. CASA 2006;17(3–4):281–92.

[8] Kavan L, Žára J. Spherical blend skinning: a real-time deformation of articulated models. In: Proceedings of the symposium on interactive 3D graphics and games. ACM; 2005. p. 9–16.
[9] Le BH, Deng Z. Smooth skinning decomposition with rigid bones. ACM Trans Graph 2012;31(6):199:1–10.
[10] Vaillant R, Barthe L, Guennebaud G, Cani MP, Rohmer D, Wyvill B, et al. Implicit skinning: real-time skin deformation with contact modeling. ACM Trans Graph 2013;32(4):125:1–12.
[11] Kavan L, Collins S, Žára J, O'Sullivan C. Geometric skinning with approximate dual quaternion blending. ACM Trans Graph 2013;27(4):105:1–23.
[12] Wilhelms J, Van Gelder A. Anatomically based modeling. In: Proceedings of the conference on computer graphics and interactive techniques; 1997. p. 173–80.
[13] Nedel L, Thalmann D. Modeling and deformation of the human body using an anatomically-based approach. In: Proceedings of the Computer Animation; 1998. p. 34–40.
[14] Nedel L, Thalmann D. Anatomic modelling of deformable human bodies. Visual Comput 2000;16:306–21.
[15] Capell S, Green S, Curless B, Duchamp T, Popović Z. Interactive skeleton-driven dynamic deformations. In: Proceedings of the conference on computer graphics and interactive techniques; 2003. p. 586–93.
[16] Guo Z, Wong KC. Skinning with deformable chunks. Comput Graph Forum 2005;24(3):373–81 (EUROGRAPHICS).
[17] Venkataraman K, Lodha S, Raghavan R. A kinematic variational model for animating skin with wrinkles. Comput Graph 2005;29:756–70.
[18] Galoppo N, Otaduy MA, Tekin S, Gross M, Lin MC. Soft articulated characters with fast contact handling. Comput Graph Forum 2007;26(3):243–53.
[19] Lee SH, Sifakis E, Terzopoulos D. Comprehensive biomechanical modeling and simulation of the upper body. ACM Trans Graph 2009;28(4):99:1–17.
[20] Kim J, Pollard NS. Fast simulation of skeleton driven deformable body characters. ACM Trans Graph 2011;30(5):121:1–19.
[21] McAdams A, Zhu Y, Selle A, Empey M, Tamstorf R, Teran J, et al. Efficient elasticity for character skinning with contact and collisions. ACM Trans Graph 2011;30(4):37:1–12.
[22] Kavan L, Sorkine O. Elasticity-inspired deformers for character articulation. ACM Trans Graph 2012;31(6):196:1–8.
[23] Li D, Sueda S, Neog DR, Pai DK. Thin skin elastodynamics. ACM Trans Graph 2013;32(4):49:1–10.
[24] Li S, Huang J, de Goes F, Jin X, Bao H, Desbrun M. Space-time editing of elastic motion through material optimization and reduction. ACM Trans Graph 2014;33(4):108:1–10.
[25] Lewis JP, Cordner M, Fong N. Pose space deformation: a unified approach to shape interpolation and skeleton-driven deformation. In: Proceedings of the SIGGRAPH; 2000. p. 165–72.
[26] Rhee T, Lewis JP, Neumann U. Real-time weighted pose space deformation on the GPU. Comput Graph Forum 2006;25:439–48.
[27] Allen B, Curless B, Popović Z. Articulated body deformation from range scan data. In: Proceedings of the SIGGRAPH; 2002. p. 612–9.
[28] Kurihara T, Miyata N. Modeling deformable human hands from medical images. In: Proceedings of the symposium on computer animation; 2004. p. 355–63.
[29] Weber O, Sorkine O, Lipman Y, Gotsman C. Context aware skeletal shape deformation. Comput Graph Forum 2007;26(3):265–74.
[30] Park SI, Hodgins JK. Data-driven modeling of skin and muscle deformation. ACM Trans Graph 2008;27:96:1–6.
[31] Feng WW, Kim BU, Yu Y. Real-time data driven deformation using kernel canonical correlation analysis. ACM Trans Graph 2008;27:91:1–9.
[32] Neumann T, Varanasi K, Hasler N, Wacker M, Magnor M, Theobalt C. Capture and statistical modeling of arm-muscle deformations. Comput Graph Forum 2013;32:285–94.
[33] Le BH, Deng Z. Robust and accurate skeletal rigging from mesh sequences. ACM Trans Graph 2014;33(4):84:1–10.
[34] Scheepers F, Parent RE, Carlson WE, May SF. Anatomy-based modeling of the human musculature. In: Proceedings of conference on computer graphics and interactive techniques; 1997. p. 163–72.
[35] Shin SY, Pyun H, Shin HJ, Sung S, Shin Y. On extracting the wire curves from multiple face models for facial animation. Comput Graph 2001;28:757–65.
[36] Hyun DE, Yoon SH, Chang JW, Kim MS, Jüttler B. Sweep-based human deformation. Visual Comput 2005;21:542–50.
[37] You LH, Yang XS, Zhang JJ. Dynamic skin deformation with characteristic curves. Comput Animat Virtual Worlds 2008;19:433–44.
[38] Chaudhry E, You LH, Jin X, Yang XS, Zhang JJ. Shape modeling for animated characters using ordinary differential equations. Comput Graph 2013;37(6):638–44.
[39] Huang H, Chai J, Tong X, Wu HT. Leveraging motion capture and 3D scanning for high-fidelity facial performance acquisition. ACM Trans Graph 2011;30(4):74:1–10.
[40] Beeler T, Hahn F, Bradley D, Bickel B, Beardsley P, Gotsman C, et al. High-quality passive facial performance capture using anchor frames. ACM Trans Graph 2011;30(4):75:1–10.
[41] Zhang L, Snavely N, Curless B, Seitz SM. Space time faces: high resolution capture for modeling and animation. ACM Trans Graph 2004;23(3):548–58.
[42] Chai JX, Xiao J, Hodgins J. Vision-based control of 3d facial animation. In: Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation; 2003. p. 193–206.
[43] Decarlo D, Metaxas D. Optical flow constraints on deformable models with applications to face tracking. Int J Comput Vis 2000;38(2):99–127.
[44] Pighin F, Salesin DH, Szeliski R. Resynthesizing facial animation through 3D model-based tracking. In: Proceedings of the seventh international conference on computer vision; 1999. p. 143–50.
[45] Cao C, Hou Q, Zhou K. Displaced dynamic expression regression for real-time facial tracking and animation. ACM Trans Graph 2014;33(4):43:1–10.
[46] Saragih J, Lucey S, Cohn J. Deformable model fitting by regularized landmark mean-shift. Int J Comput Vis 2011;91(2):200–15.
[47] Xiong X, De la Torre F. Supervised descent method and its applications to face alignment. In: Proceedings of the IEEE conference on computer vision and pattern recognition; 2013. p. 532–9.
[48] Coşkun1 SB, Atay MT, Öztürk B. Advances in vibration analysis research. Rijeka, Croatia: InTech; 2011.
[49] Sumner RW, Popović J. Deformation transfer for triangle meshes. ACM Trans Graph 2004;23(3):399–405.