Fiber-dependent Approach for Fast Dynamic Character Animation

Ayano Kaneda
Waseda University
dizzy-miss-lizzy@moegi.waseda.jp

Tsukasa Fukusato
Waseda Research Institute for Science and Engineering
tsukasa@moegi.waseda.jp

Yoshihiro Fukuhara
Waseda University
fyoshi@ruri.waseda.jp

Takayuki Nakatsuka
Waseda University
t59nakatsuka@fuji.waseda.jp

Shigeo Morishima
Waseda Research Institute for Science and Engineering
shigeo@waseda.jp

Abstract
Creating secondary motion of character animation including jiggling of fat is demanded in the computer animation. In general, secondary motion from the primary motion of the character are expressed based on shape matching approaches. However, the previous methods do not account for the directional stretch characteristics and local stiffness at the same time, that is problematic to represent the effect of anatomical structure such as muscle fiber. Our framework allows user to edit the anatomical structure of the character model corresponding to creature’s body containing muscle and fat from the tetrahedral model and bone motion. Our method then simulates the elastic deformation considering anatomical structures defined directional stretch characteristics and stiffness on each layer. In addition, our method can add the constraint for local deformation (e.g. biceps) considering defined model’s characteristics.

Keywords: fiber definition, shape matching, linear blend skinning, multi-layer system

1 Introduction
Character animation has been advanced using physically-based simulations for expressive animation, in particular, secondary motion, which breathe life into characters with the jiggling and wobbly action. Using a rig or a skeleton is an efficient method for generating the secondary motion of a character. One of the approaches is skinning, which conditions the surface mesh of the object by mapping from a skeletal structure. In some approaches, the model is deformed into the required shape using example data (e.g., character pose designed by artists and result shapes based on physical simulation) assigned to each model. Others approaches integrate voxels or anatomical construction into the character model to generate more lifelike movement and to preserve the volume of the character model. However, due to the complexity of setting anatomical effects such as inter-
action between muscle and fat, dynamic animation remains an unintuitive and time-consuming process. Moreover, the anisotropic deformation based on muscle fiber orientation requires heavy burden on setting apposite parameters to design lifelike character animation. In this paper, we propose a novel method for expressing secondary motion easily from a skeletal character animation with a multi-layer system and anatomical fiber vectors. We integrate the anatomical parameters (local anisotropy and stiffness) into the input model, which is a volumetric tetrahedral mesh. Then, we simulate the elastic deformation using the shape matching algorithm owing to the small computational cost and robust simulation. Our method can add stiffness and fiber orientation to local regions in the model and can express elastic deformation with anisotropy response. To consider secondary motion, our framework propagates both the internal and the external force over the object volume. In addition, the volume of the model is preserved. Our method makes three contributions to the developments of the lifelike character animation:

- allows easy setting of anatomical parameters based on intuition
- incorporates elastic deformation considering the fiber orientation
- adds constraint for the partial deformation (e.g., biceps)

### 2 Related Work

In character animation linear blending / interpolation approaches, as well as non-linear mapping approaches, which involve controlling a skeleton using a range of values from real-time and closed-form skinning, are used. Jacobson et al. [1] propose an efficient algorithm that computes skinning weights by minimizing the non-linear rigidity energy. In addition, Kavan et al. [2] improve skeleton-based character animation, which approximates a non-linear elastic deformation to an elastic energy function and a joint-based deformer. While these approaches deliver high-quality primary animations, designing life-like animations including secondary motion is still difficult using these methods only. In general, dynamic skin deformation with interactive applications and frequent conformations of the result action are in high demand for animators.

Mukai et al. [3] propose a helper bone controller that maps the primary skeleton motion to the dynamic movement of helper bones using the trained motion sequences consisting of skeleton motions and corresponding skin deformations. The helper bone controller enables non-linear and complicated deformations by considering the effect of soft-tissue dynamics, but does not consider mesh-to-mesh interaction (collision and self-collision) against external force. To overcome these problems, two approaches have been investigated: (1) rig-space approaches (the subspace energy minimization with rigging controls and (2) physically-based approaches. Rig-space approaches [4, 5, 6] describe the mesh deformation as linear modes. In these approaches, it is difficult to consider an anisotropic response and design a multi-layer system composed of spaces of bone, muscle, fat, and skin to simulate the interaction between layers. In addition, rig-space approaches require high computational costs due to the non-linear properties of the rig. In contrast, physically-based approaches integrate anatomical knowledge into computing. Teran et al. [7] and Fan et al. [8] indicate that the volumetric shape with bones, which function as rigid constraints and preserving the pose is informative for a lifelike character animation. Liu et al. [9] and Ruman et al. [10] report handling of mesh deformation with collisions against surrounding environment. In particular, McAdams et al. [11] demonstrated self-collision of the model with soft tissue deformation in interactive time. Unfortunately, previous physically-based proposals have common problems: the stability of simulation and the computational cost depend on time step width. We solve these problems using a shape matching algorithm [12], which is a fast and robust method for simulating elastic deformations.

Use of a shape matching algorithm allows secondary movements to be controlled with physical parameters and designed intuitively by a multi-layer model. Shape matching imposes geometric constraints which optimize the positions of vertexes using elastic energy to the simulation of a deformable object. Chen et al. [13] propose a robust method to preserve the ob-
We make the clusters for each body layer to the mesh model. Each layer has the constraints (stiffness and fiber vector) set by a user. We then compute the mesh deformation using Linear Blending Skinning for the primary deformation and fiber-dependent Shape Matching for the secondary deformation.

However, they do not focus on anatomical feature of character model with a multi-layer system because the volume spaces overlap each other. Their method does not properly account for secondary motion such as fat jiggling because of its strong volume preservation. Iwamoto et al. [14] suggest a simple multi-layer approach which can handle secondary motion. They divide the model into multiple layers imitating anatomical structure such as bone, muscle, and fat. They then simulate elastic deformation of the body by considering stiffness defined on each layer. This simple multi-layer approach succeeded in simulating secondary motion with the different stiffness values for each layer. However, as the directional stretch characteristics in the model are not considered, that is problematic to consider the effect of anatomical structure such as muscle fiber. Iijiri et al. [15, 16] add fiber orientation constraints which represent the expansion and contraction direction of the motion to generate character animation of flexible bodies. However, many complicated inputs are required to adjust the model behavior. In addition, as the partial stiffness cannot be represented without directly adding motion data into local regions, generating secondary motion such as jiggling of fat from the skeletal motion remains problematic.

Our method solves these problems by elastic deformation considering partial directional stretch characteristics and stiffness at the same time. Our goal is to express the effect of anatomical fiber vectors which have the local directional stretch characteristic for secondary motion.

3 Methods

Our system generates 3D dynamic animation from skeletal motion data and a tetrahedron character model. We construct a local region, a set of one vertex connecting its immediate neighbors. The local region can represent overlap of neighboring local regions, which play a role in transmitting the force from the outside gradually using shape matching. We first specify several fiber orientations of local region such as muscle and skin for stretchable (or non-stretchable) constraints. When a user sets the constraints, the global fiber field is calculated automatically calculated automatically from a character structure such as the bone direction, and the user specified constraints. By the fiber vector field, it is possible to handle stretchable orientation of model mainly inspired by the human body structure. In ad-
In addition, we define a multi-layered model, where each layer has two parameters for designing different movements. In addition, our method enables to represent the locally large deformation using the constraints based on joints’ angle information with volume preservation (Section 3.5).

### 3.1 Body Structure Classification

We partition the model into multi-layers to form the anatomical structure of the character model corresponding to creature’s body. These layers mainly imitate the anatomical tissue layers such as bone, muscle, and fat, and each cluster can be given different motion. In the bone layer, we manipulate the surface mesh of bone layer by linear blend skinning. This illustrates the primary deformation of the model. To express the secondary deformation, the other clusters simulate the elastic deformation based on shape matching (Section 3.4). In this paper, we divide into the layers by distance from the bone. In addition, our method allows a user to set optional layers manually to improve the quality of the mesh deformation.

### 3.2 Setting Fiber Parameters

We set the fiber parameters on each body layer to express the detailed secondary deformation. Each body layer has two parameters: (1) $\alpha \in [0.0, 1.0]$ is the stiffness that determines how the model maintain the original shape, and (2) $\beta \in [0.0, 1.0]$ is the weight of the fiber vector, which represents how the fiber vector effect. We adopt $\alpha$ which decreases the value according to the distance from the bone layer. In addition, the muscle layer has bigger value of $\beta$ than the other layers. Our method allows a user to choose $\beta \in [0.0, 1.0]$ in each layer.

### 3.3 Global Fiber Vector Field Definition

We need to assign the fiber vector $\delta \in \mathbb{R}^3$ to each local region (one-ring neighborhood) in order to apply the fiber orientation to the model. Given the sets of the constraint vectors, our method automatically computes the global fiber vectors by interpolating. In this paper, the interpolation is propagated for the entire tetrahedral mesh by Laplacian smoothing [17] as follows:

$$l_i = \delta_i - \sum_{j \in M} \rho_j^i \delta_j$$

$$\min_{\delta_i} \left( \sum_i ||l_i||^2 + \lambda \sum_{k \in C} ||\delta_k - c_k||^2 \right)$$

where $N \ni i$ is the number of vertexes, $M \ni j$ is the set of one-ring neighboring vertexes of $i$-th vertex, $C \ni k$ is the number of user inputs, $q_i$ is the position of a vertex, $\lambda$ is the coefficient (in this paper, $\lambda = 1.0 \times 10^3$), $\rho_j^i$ is the weight from $N_j$ and $c_k$ is the user-specified constraint. The weight $\rho_j^i$ is given by the following:

$$\rho_j^i = \frac{\exp(-r_{ij})}{\sum_{k \in N_j} \exp(-r_{kj})}$$

where $r_{ij}$ is the Euclidean distance between the position $q_i$ and $q_j$. In consequence, the global fiber vector field can be automatically generated. For a character model, we set the constraints on the bone layer and the surface mesh based on the anatomical structure of the body. Our method is inspired from Saito et al. [18] that muscle fibers tend to point in the same direction from one tendon to another. We set the constraint vectors on the bone layer, to be parallel to the bone direction. In addition, we add constraint vectors along the surface using an orthogonal unit vector to the the surface.

### 3.4 Fiber-dependent Shape Matching Algorithm

We update the position $q_i$ and the velocity $p_i$ of $i$-th vertex of the model based on the shape
Figure 3: The orientation of fibers for each local region. The result is interpolated by constraints tangential to bone and surface mesh.

Figure 4: Algorithm of shape matching. To compute the goal position \( g_i \) of particle \( i \) in local region \( r \), the rest pose of the region is appropriately deformed by scale matrix \( T_r \). We then minimize the rotation matrix \( R_r \) between a current pose and a rest pose.

Matching algorithm as follow:

\[
p'_i = p_i + \alpha \frac{g_i - q_i}{h} + h f^{ext}_i \frac{f^{ext}_i}{m_i} \tag{4}
\]
\[
q'_i = q_i + h p'_i \tag{5}
\]

where \( h \) is the time step, \( f^{ext}_i \) is the external force against \( i \)-th vertex, \( m_i \) is the mass, and \( g_i \) is the goal position for \( i \)-th vertex respectively. \( \alpha \in [0, 1] \) is the stiffness parameter. Then, we need to determine the goal position \( g_i \). To focus on the local region \( N_r \ni i \), the relative vectors with respect to its center of mass \( c_i \), \( b_i^{curr} \) and \( b_i^{est} \) are described as below:

\[
b_i^{curr} = (q_i - c_i) \tag{6}
\]
\[
b_i^{est} = T_r(\theta)(q_i^0 - c_i^0) \tag{7}
\]

where \( N_r \ni i \) is the set of vertexes in the local shape, \( T_r \) is the scaling matrix, and \( c_i \) is the center of the mass for the local region \( N_r \) as following equation:

\[
c_i = \frac{\sum_{i \in N_r} w_i^r q_i}{\sum_{i \in N_r} w_i^r} \tag{8}
\]

where \( w_i^r \) represents orientation-dependent weights of the local shape and we explain in Section 3.5. The relation between the rest position \( q_i^0 \) and its center of the mass \( c_i^0 \) is the same as \( q_i \) and \( c_i \) in equation (8). In addition, we compute the rotation matrix \( R_r \) to fit the previous pose into the current pose for each local shape. In our method, we determine \( R_r \) of \( N_r \) using the method of Muller et al. [19].

\[
\arg\min_{R_r} \sum_{i \in N_r} w_i^r (R_r b_i^{est} - b_i^{curr})^2 \tag{9}
\]

Given the fitting rotation matrix \( R_r \), we estimate the goal position of \( i \)-th vertex in \( N_r \) as follow equation:

\[
g_i^r = R_r T_r (q_i^0 - c_i^r) + c_r \tag{10}
\]

Then, as one vertex presents in multiple local regions, the goal position \( g_i \) is obtained by the linear sum which is weighted from the each local shape and the fiber constraints as following equation:

\[
g_i = \sum_{r | i \in N_r} \frac{w_i^r g_i^r}{\sum_{r | i \in N_r} w_i^r} \tag{11}
\]

3.5 Fiber-dependent Anisotropy Response

Fiber Orientation

We deform the vertexes considering the fiber orientation defined for each local region, which represents the direction of easy contraction. In the process of computing the mass center of local regions and blending the effect of belonging local regions on the given vertex, weighting coefficients could affect the deformation
anisotropy.

\[
w_i^r = m_i \times \frac{(s \cdot r_i)^2}{\|r_i\|^2} \tag{12}
\]

\[
r_i = q_i^0 - \frac{\sum_{k \in N_i} m_k q_k^0}{\sum_{k \in N_i} m_k} \tag{13}
\]

\[
s = \frac{1}{\sqrt{\beta^2 + (1 - \beta)^2}} \times \left( \beta v_i + (1 - \beta) \frac{r_i}{\|r_i\|} \right) \tag{14}
\]

By setting \(\beta\) for each layer (Section 3.1), we adjust the effect of fiber vector on each layer. In our framework, for the fat layer \(\beta\) is set as 0.0, denoting the layer has no fiber, and in muscle layer almost 1.0, which expresses that the muscle is difficult to move in the direction along the fiber orientation.

### 3.6 Muscle-inspired constraint and Volume Preserving

Our method can add local deformation into the model easily, by setting a additional constraint. In character animation, when the character bends his arm, his biceps builds. When the human flexes his arm, biceps brachii muscle stands out and triceps brachii muscle extend at the same time. In addition, we assume that upper muscle bulges perpendicularly to the muscle fiber orientation as in [18]. In this paper, we add a simple constraint to elbow joint, and represent the flexing model’s biceps using defined fiber orientation.

\[T_i(\theta) = DMD^T \tag{15}\]

\[
M = \begin{pmatrix}
c(\theta) & 0 & 0 \\
0 & c(\theta)^{-\frac{1}{2}} & 0 \\
0 & 0 & c(\theta)^{-\frac{1}{2}}
\end{pmatrix} \tag{16}
\]

\[c(\theta) = \beta \sin \theta \times n_i \cdot \tau + 1.0 \tag{17}\]

where \(\theta\) is horn angle of the forearm and bicep, \(n_i\) is the unit vector, perpendicularly to the fiber orientation of \(i\)-th local region from the closest bone layer’s surface, and \(\tau\) represent the unit vector orthogonal to the plane made by the arm sections. Matrix \(D\) consists of three vectors, \(d_1\) is represented by \(n_i\) and two vectors \(d_2\) and \(d_3\), and \(d_1, d_2, d_3\) constructs a normalized orthogonal basis at \(N_i\). \(D\) represents the orientation field. This effect is added on the muscle layer by adjusting \(\beta\), which is almost zero in fat layer. In addition, the determinant of matrix \(T_i = 1.0\). Matrix \(T_i\) represents the scale change of the local region \(N_i\). Therefore, the volume could be preserved in each region when the determinant of matrix \(T_i = 1.0\). In particular, it is difficult to handle the partial deformation without pre-computation and example data. However, our approach allows users to easily design the motion in the local region using defined fiber vector field.
4 Result

**Bar (gravity):** Figure 6 shows the dependency of fiber vector. As shown in Figure 6, the bar is more rigid parallel to the fiber vector than orthogonal to one. As set the weight in equation (12), calculated goal position tends to get the larger weighted parallel to the fiber vectors. In consequence, our method can add fiber properties to the model using this weight.

**Bar (beta):** We also represent the effect of β parameter in Figure 7. These four bars have different β parameter, the left as β = 0.0, the left middle as β = 0.6, the right middle as β = 0.75 and the right as β = 1.0 and all fiber vectors
Figure 10: Arm simulation with the additional constraint on biceps. The results obviously prove the difference in the deformation because of the value of $\beta$. (a) and (b) set $\beta$ as 0.0, and (c) and (d) set $\beta$ as 1.0.

are orthogonal to the floor. As shown in Figure 7, each bar demonstrates different deformation due to the effect of $\beta$. The bar with $\beta = 0.0$ represents the isotropic rigidness and then tends to have directional stretch feature with increasing $\beta$. In these model, the fiber vectors work as bellows structure. Then, as increasing $\beta$ parameter, these model tend to more and more hang down under gravity.

Bar (multi-layer): We also compare the bar model deformation under only the effect of $\alpha$ and $\alpha$ and $\beta$ in Figure 8. These bars divided into multi layers. The left layer as $\alpha = 1.0$ and layer as $\alpha = 0.3$ and two layer set $\beta$ as zero. It shows the deformation like Iwamoto et al. [14] with only stiffness parameter model. The layer set as small $\beta$ to decrease the rigidness and Iwamoto et al. [14] expresses the jiggling of fat using the difference of rigidness. The other bars are set non-zero value for each layer and $\alpha$, $\beta$. In addition to the effect of the difference of the rigidness, these model represent the effect of fiber vectors and its difference of weights between layers. The bar bends more smoothly by the effect of the fiber vectors and we can design the deformation than only stiffness model. As a result, we design these three differential motion using these two parameters (our method (i), (ii) and (iii)). Our method allows user to extend the expression.

Patrik (jumping-animation): As a result, we generate character animation in Figure 9 under the effect of $\alpha$ and $\beta$. In Figure 9, we set muscle layer as $\alpha_M = 1.0$, $\beta_M = 1.0$, and fat layer as $\alpha_F = 0.3$, $\beta_F = 0.0$. As shown in Figure 9, the movements of the fat lags behind the input skeletal movement when the model jumps. Based on fiber vector field defined by constraint on bone and surface mesh, we easily add the fiber vectors effect to secondary motion generation.

Biceps: In Figure 10, we simulate the biceps. Figure 10 shows the effect of constraint for biceps. The results obviously prove the difference in the deformation from the value of $\beta$. (a) and (b) set $\beta$ as 0.0, and (c) and (d) set $\beta$ as 1.0. We evaluate the volume change rate of (c) and (d) $\beta$ as 1.0 for (a) and (b) $\beta$ as 0.0 through the arm bending motion. Then, we success in expressing biceps building with constraint. The average rate of the volume change of the model with Muscle-inspired constraint is only 0.291 percent from non-constraint model’s volume. Although we contract local regions in the model with constraint, the total volume of the computed goal position is almost same as the result of normal shape matching. Thus, our framework can design the partial deformation using defined fiber vector field and additional constraint.

Computational cost: In table 2, we reports our testing scenarios and the run time of our method. We executed on an Intel i7-4910MQ CPU at 2.90GHz. All scenarios are produced relatively fast. In addition, we would also like to further enhance the computational speed by implementing GPU-based solutions in the future.

4.1 Limitation

Our approach is inherently limited in physical accuracy because of its pure and geometrical nature, since it is based on shape matching algorithm. However, allowing a user to edit and simulate in real time is more important in the context of art-directed animation than physical accuracy.
Table 2: Model statistics and simulation timings

<table>
<thead>
<tr>
<th>model</th>
<th>vertex</th>
<th>tetrahedron</th>
<th>fps</th>
</tr>
</thead>
<tbody>
<tr>
<td>bar (gravity)</td>
<td>2297</td>
<td>9216</td>
<td>12.5</td>
</tr>
<tr>
<td>bar (beta)</td>
<td>817</td>
<td>3072</td>
<td>30.861</td>
</tr>
<tr>
<td>bar (multi)</td>
<td>817</td>
<td>3072</td>
<td>31.645</td>
</tr>
<tr>
<td>patrik</td>
<td>1574</td>
<td>6534</td>
<td>22.22</td>
</tr>
<tr>
<td>arm</td>
<td>1941</td>
<td>7753</td>
<td>28.57</td>
</tr>
</tbody>
</table>

accuracy. Our framework can express secondary motion such as jiggling of fat and biceps. However, computing sagging faces is challenging in our framework. It needs additional constraint between skin and fat, then we have to consider skin layers. Due to lack of muscle fiber connections, when a character twists his arm, the streaks of muscles’ effect need optional constraints on fibers.

4.2 Conclusion and Future work

We have presented a system for generating dynamic character animation from a skeletal motion. Our proposed method consists two steps: (i) defining the multi-layer system with fiber vectors, (ii) computing a global shape with anisotropy responses using the shape matching algorithm. Our method employs fiber-dependent parameters into the shape matching algorithm; it consequently achieves the simple and intuitive control for the anisotropic stiffness and motion of the character shape. As a result, our method performs expressing the effects of anatomical fiber vectors which have the local directional stretch characteristics for secondary motion. We believe that our system can assist to apply the unwieldy physically-based animation into user-friendly application systems. In addition, we would like to attempt automatically estimating the parameters from captured data such as sensing devices or video sequence.

Acknowledgements

We would like to thank Naoya Iwamoto and Tatsuya Yatagawa for their precious advices. This project was supported in part by ACCEL, JST.

References


