Efficient Dynamic Skinning with Low-Rank Helper Bone Controllers

Tomohiko Mukai*
Tokai University

Shigeru Kuriyama
Toyohashi University of Technology

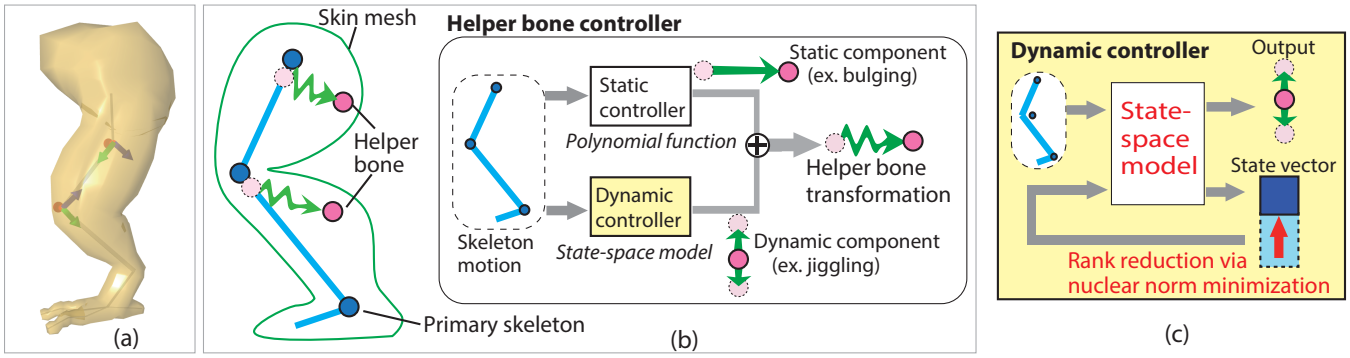

Figure 1: *(a) Linear blend skinning with two helper bones. (b) Skin deformation is synthesized using helper bones that are procedurally controlled according to the primary skeleton motion using two types of controllers. (c) The stable and efficient dynamic controller is constructed by low-rank approximation using the nuclear norm optimization method.*

Abstract

Dynamic skin deformation is vital for creating life-like characters, and its real-time computation is in great demand in interactive applications. We propose a practical method to synthesize plausible and dynamic skin deformation based on a helper bone rig. This method builds helper bone controllers for the deformations caused not only by skeleton poses but also secondary dynamics effects. We introduce a state-space model for a discrete time linear time-invariant system that efficiently maps the skeleton motion to the dynamic movement of the helper bones. Optimal transfer of nonlinear, complicated deformations, including the effect of soft-tissue dynamics, is obtained by learning the training sequence consisting of skeleton motions and corresponding skin deformations. Our approximation method for a dynamics model is highly accurate and efficient owing to its low-rank property obtained by a sparsity-oriented nuclear norm optimization. The resulting linear model is simple enough to easily implement in the existing workflows and graphics pipelines. We demonstrate the superior performance of our method compared to conventional dynamic skinning in terms of computational efficiency including LOD controls, stability in interactive controls, and flexible expression in deformations.

Keywords: Dynamic skinning, helper bone rig, state-space model, nuclear norm optimization

Concepts: •**Computing methodologies** → **Animation;**

*e-mail:tmki@acm.org

SIGGRAPH '16 Technical Paper, July 24-28, 2016, Anaheim, CA,
ISBN: 978-1-4503-4279-7/16/07
DOI: http://dx.doi.org/10.1145/2897824.2925905

1 Introduction

In computer animation, deformation of skin surface significantly affects the visual appearance of the character's motion. Building a good character rig is a key component in synthesizing skin deformation; it should produce plausible behaviors while allowing a full range of motion control for animators via a few intuitive parameters. Efficient computation is also an important factor, especially for interactive applications such as games. A common approach incorporates an articulated skeleton as the primary rig and computes vertex positions of skin surface according to the skeleton pose. Linear blend skinning (or LBS) techniques have been widely used in interactive applications because of their efficiency and simplicity. This kinematic model has been successfully applied to actual production workflows because most artists and animators are very skilled at skeleton-based rigging. However, these skinning models cannot produce dynamic deformation caused by external force, momentum and inertia of soft tissues such as muscles and fats.

There have been many studies on synthesizing realistic skin deformation that take into account the effects of soft-tissue dynamics. Physics-based volumetric simulation, which includes finite element methods and musculoskeletal systems, is a promising approach for generating physically-valid skin deformation. Physics-based simulation, however, requires expensive computational costs and careful rigging of the character's musculoskeletal model. Moreover, the simulation is inferior to kinematic models in terms of controllability and stability, both of which are necessary for real-time, procedural generation of character animations.

Our goal is to develop a skeleton-based technique to synthesize visually-plausible dynamic skin deformation in real-time. One of the most important factors for our target is computational efficiency rather than physical validity. The expected performance should be in the order of tens of microseconds per frame per character at worst because only a limited time budget is allowed for computing skin deformation, especially with commercial products. The computational cost should also be predictable and constant regardless of changes in a character's surrounding, i.e., no iterative method is acceptable at runtime computation. Furthermore, the skinning model should be simple and compact for minimizing both model complex-

ity and memory requirements. Compatibility with standard graphics pipelines is also important when incorporating a new technology into the existing production workflows.

The helper bone rig has become a practical real-time technology for synthesizing stylized skin deformation based on LBS [Parks 2005; Kim and Kim 2011]. The helper bone is a secondary rig that influences the skin deformation, and its pose is procedurally controlled according to the pose of the primary skeleton. This technique has been widely used in many products because of its efficiency, flexibility, and compatibility with the standard graphics pipeline. Designing dynamic skin deformation with helper bone rigs, however, is still an annoying and intricate task even for expert riggers and animators.

We propose a practical method for dynamic skinning with a helper bone rig. Our main challenge is to establish plausible skin deformation with minimal computational cost, while preserving sufficient quality with conventional data-driven or physics-based methods. Our key idea is to procedurally control helper bones according to skeleton motion in order to imitate dynamic skin deformation as summarized in Figure 1. We build the helper bone controller based on a state-space model (or SSM) of a discrete time linear time-invariant system [Ljung 1999] for mapping the movements of a character's skeleton to those of helper bones. The dynamic system of bone controllers is learned from sample sequences of skeleton motions and skin deformations. We also employ a nuclear norm optimization method: a generalization of compression-sensing methods, such as l_1-norm optimization, for effectively minimizing the computational redundancy of a bone controller whose behavior is learned from physically simulated animations. This optimal low-rank approximation of the controller ensures sufficiently accurate reproduction of training data while reducing the computational cost.

Our approach includes the following technical contributions:

- An SSM-based helper bone controller that can stably and efficiently synthesize dynamic skin deformation while allowing flexible configurations.
- Robust SSM estimation with nuclear norm optimization that effectively builds the compact bone controller by learning sample skin deformations.
- A dynamic skinning scheme with a helper bone rig that is applicable to various types of character models, and is fully compatible with current production pipelines based on LBS technique.

A main drawback of our approach is the lack of details on skin deformation caused by wrinkles or self-collisions, etc., because the sparse set of helper bones merely linearly approximates the dynamics of soft tissues. Moreover, our kinematic approach does not guarantee volume preservation and complex secondary dynamics caused by external forces or interaction with other objects. The effect of gravity is also not taken into account. Yet, our method is practical for adding plausible secondary dynamics effects to the character's skin without requiring extra computational cost or a major modification of the runtime system.

2 Related Work

Accurate skin deformation is often generated using physics-based musculoskeletal [Li et al. 2013] or volumetric [Fan et al. 2014] simulations. These simulation-based methods are unsuited to intuitively or intentionally changing the style of deformations and real-time applications because of their high computational cost. DyRT employs precomputed modal vibration models to efficiently synthesize the dynamic skin deformation [James and Pai 2002]. Although this technique is useful for emulating physically-valid deformation, artist-crafted stylized behavior is not taken into account. A data-driven approach learns the dynamical properties of soft-tissue materials from the training data [Shi et al. 2008], but still suffers from the computational complexity. Position-based dynamics (or PBD) was used to synthesize skin deformation caused by self-collisions and the secondary effects of soft tissues [Rumman and Fratarcangeli 2015]. This method provides plausible and stable soft body motion at interactive rates, but requires the elaborate construction of a PBD-based rig. For this reason, the kinematics-based approach is still widely used because it requires no additional mechanism on a standard character rig.

The kinematic skinning method efficiently computes vertex positions of the skin mesh based on the pose of the internal skeleton structure. Linear blend skinning is a standard technique for synthesizing skin deformation in real-time applications [Magnenat-Thalmann et al. 1988], and many refinements have been proposed to overcome undesirable artifacts such as the so-called candy-wrapper and elbow-collapse [Wang and Phillips 2002; Kavan et al. 2007; Kavan and Sorkine 2012]. EigenSkin constructs an efficient model of additive vertex displacement for LBS using principal component analysis [Kry et al. 2002]. Corrective blendshape techniques like pose-space deformation [Lewis et al. 2000] have also been widely used to enhance the quality of skin deformation by interpolating a set of sample shapes. Such kinematic, data-driven techniques, however, do not generate dynamic skin deformation and have not been supported yet by many graphics engines, especially on mobile devices.

A helper bone rig has been used to compensate for unnatural deformations of the naive LBS [Mohr and Gleicher 2003] and to create stylized skin deformations such as muscle bulging [Parks 2005; Kim and Kim 2011; Mukai 2015]. Such helper bone rigs can be easily integrated into the standard LBS-based system, and have often been extended to generate secondary dynamics effects. For example, a simple mass-spring network is used for driving helper bones in order to mimic the jiggling of soft tissues. However, such a simple physics model is difficult to stably control, especially for simulations with large time steps. We use a state-space model to robustly approximate the dynamics of helper bones.

Other approaches use secondary rigs for synthesizing dynamic skin deformation. The kinodynamic skinning technique [Angelidis and Singh 2007] provides volume-preserving deformation based on proxy muscles. A rig-space physics technique optimizes free parameters of the hand-crafted kinematic rig to approximate physically-simulated skin deformation [Hahn et al. 2012; Hahn et al. 2013]. These approaches have a relation with our method in that the degree of freedom in controlling skin deformation is reduced by procedurally controlling the secondary rig parameters according to the pose of the primary rig.

The learning-based approach for skin deformation analyzes the relationship between a skeleton pose and its corresponding skin shape using a large set of samples. A regression technique was proposed to estimate linear mapping from the skeletal pose to the deformation gradient of the skin surface polygons [Wang et al. 2007]. The seminal work of Park and Hodgins [2008] predicts optimal mapping from skeletal motion to dynamic motion of several markers placed on a skin surface. Neumann et al. [2013] have proposed a statistical model of skin deformation that is learned from human skin shapes captured with range scan devices. The MoSh model estimates dynamic skin deformation from a sparse set of motion capture markers using a statistical model of human skin shapes [Loper and Black 2014]. This method synthesizes the skin shape in a low-dimensional subspace so as to approximate the movement of the markers. Dyna model also constructs a dynamic skin defor-

mation model using the subspace analysis of 4D scans of human subjects [Pons-Moll et al. 2015]. The triangle deformation of the skin mesh is generated using a modified version of second-order autoregressive model with exogenous inputs (or ARX) in the low-dimensional subspace. The SMPL model learns corrective blend-shape models from shape samples [Loper et al. 2015], and is extended to the DMPL model that synthesizes dynamic deformation by incorporating a dynamics model similar to Dyna. These methods successfully produce realistic deformation of human skin. However, these learning-based techniques require the runtime cost for each skin vertex increasing in proportion to the number of primary bones, due to its global property. In contrast, our method only drives fewer helper bones by primary skeleton, and the runtime cost of the LBS-based skin deformation is unaffected by the number of primary bones.

Soft body dynamics is often modeled using an ARX model, similar to the conventional learning-based dynamic skinning approach [Pons-Moll et al. 2015; Loper et al. 2015]. The stable real-time clothing model uses the second-order ARX model [de Aguiar et al. 2010] to predict cloth shape by the linear combination of skeleton pose and the shape history over the past two frames. This method takes a non-parametric approach, i.e., cloth dynamics is synthesized in the low-dimensional subspace identified using principal component analysis (or PCA). We use a state-space model (or SSM) which is known as a generalization of the ARX model. In the control theory community, SSM has actually become a standard parametric representation for modeling an unknown dynamic system from observed input/output data. Also in the graphics community, the effectiveness of SSM was demonstrated in the motion style translation method [Hsu et al. 2005]. Moreover, we employ a nuclear norm optimization method for constructing a helper bone controller based on a low-rank SSM.

3 Preliminary

3.1 Skinning with helper bones

Here we review the typical procedure of LBS with helper bones. Given a set of indices of primary skeleton bones $\mathcal{P}$, a global transformation matrix of a primary bone is computed as 4×4 homogeneous matrices $\mathbf{G}_{p\in\mathcal{P}}$. Let $\bar{\mathbf{G}}_p$ and $\bar{\mathbf{v}}_i$ denote the matrix and the positions of the i-th vertex on a skin at the rest (or initial) pose, and the skinning transformation matrix of the primary bone be represented by $\mathbf{M}_p = \mathbf{G}_p\bar{\mathbf{G}}_p^{-1}$. We also apply accompanying secondary bones called *helper bones* whose index set is given by $\mathcal{H}$. Our skinning formula is then represented as

$$\mathbf{v}_i = \left(\sum_{p\in\mathcal{P}} w_{i,p}\mathbf{M}_p + \sum_{h\in\mathcal{H}} w_{i,h}\mathbf{S}_h\right)\bar{\mathbf{v}}_i, \tag{1}$$

where $\mathbf{v}_i$ is the position of the deformed vertex and $\mathbf{S}_h$ denotes the skinning matrix of the h-th helper bone. The first term of Equation 1 corresponds to skin deformations driven by primary skeleton, and the second term contributes an additional control for deformations using helper bones. The number of helper bones $|\mathcal{H}|$ is manually given by designers so as to balance the deformation quality and computational cost. Helper bones are procedurally controlled with simple expressions according to the pose of the primary skeleton in common practice.

3.2 Skinning decomposition

In traditional rigging processes, the skinning weights and helper bone controllers are manually designed by riggers. Our goal is to automatically optimize these variables using the training data of skeleton motion and its accompanying skin deformation. This optimization problem, however, is difficult to directly solve due to the high nonlinearity of multiple constraints. We therefore divide it into two subproblems; the first subproblem estimates all optimal skinning weights $w^*_{j,p}, w^*_{j,h}$ and skinning matrix $\mathbf{S}^*_{h,\tau}$ at each time frame $\tau \in \mathcal{T}$ so as to best approximate the training data, and this subproblem is regarded as a skinning decomposition problem [Le and Deng 2012; Le and Deng 2014; Mukai 2015]. The second subproblem approximates the estimated discrete-time transformations $\mathbf{S}^*_{h,\tau}$ by the bone controller model with respect to the primary skeleton pose. We address the first subproblem in the rest of this section and the second subproblem is detailed in the next section.

Given the training sequence of the skinning matrix of a primary skeleton $\tilde{\mathbf{M}}_{p,\tau}$ and the corresponding vertex animation $\tilde{\mathbf{v}}_{i,\tau}$, our skinning decomposition problem is formulated as a constrained least square problem to minimize the sum of the squared reconstruction error between the training skin deformation $\tilde{\mathbf{v}}_{i,\tau}$ and the reconstructed one $\mathbf{v}_i$ as

$$(w^*, \mathbf{S}^*) = \underset{w,\mathbf{S}}{\operatorname{argmin}} \sum_{\tau\in\mathcal{T}}\sum_{i\in\mathcal{V}} \left|\tilde{\mathbf{v}}_{i,\tau} - \left(\sum_{p\in\mathcal{P}} w_{i,p}\tilde{\mathbf{M}}_{p,\tau} + \sum_{h\in\mathcal{H}} w_{i,h}\mathbf{S}_{h,\tau}\right)\bar{\mathbf{v}}_i\right|_2^2 \tag{2}$$

$$\text{subject to} \quad w_{i,p} \geq 0,\ w_{i,h} \geq 0,\ \forall\, i \in \mathcal{V}, p \in \mathcal{P}, h \in \mathcal{H} \tag{3}$$

$$\sum_{p\in\mathcal{P}} w_{i,p} + \sum_{h\in\mathcal{H}} w_{i,h} = 1,\ \forall\, i \in \mathcal{V} \tag{4}$$

$$\sum_{p\in\mathcal{P}} |w_{i,p}|_0 + \sum_{h\in\mathcal{H}} |w_{i,h}|_0 \leq \kappa,\ \forall\, i \in \mathcal{V} \tag{5}$$

where $|\cdot|_n$ denotes l_n-norm and $\mathcal{V}$ is the set of vertices' indices. The constant κ indirectly constrains the number of bones contributing to deformations for adjusting the tradeoff between computational cost and accuracy. In addition to these three constraints of (3) non-negativity, (4) partition of unity, and (5) sparsity constraints on the skinning weights, we apply the rigidity constraint for enforcing all $\mathbf{S}_{h,\tau}$ to consist of only translational and rotational components. This nonlinear constrained optimization problem is robustly solved using a block coordinate descent algorithm [Mukai 2015]. Notice that all constraints must be satisfied to utilize APIs supported by common graphics engines.

4 Helper Bone Controller

4.1 Controller model

Our controller model assumes that helper bones are driven by the primary skeleton that has only spherical joints. The poses of helper bones are uniquely determined from all rotational components of the primary bones $\mathbf{r}_p \in SO(3)$ and the velocity of the root node, which is denoted by a column vector $\mathbf{u}$ as

$$\mathbf{u} := \triangle\mathbf{t}_0 \parallel \triangle\mathbf{r}_0 \parallel \mathbf{r}_1 \parallel \mathbf{r}_2 \parallel \cdots \parallel \mathbf{r}_{|\mathcal{P}|}\ , \tag{6}$$

where $\mathbf{u} \in \Re^{3|\mathcal{P}|+6}$, $\parallel$ is a concatenation operator for vector values, $|\mathcal{P}|$ is the number of primary bones, and $\triangle\mathbf{t}_0 \in \Re^3$ and $\triangle\mathbf{r}_0 \in SO(3)$ respectively denote the time-variation of the root's position and that of its orientation, both of which are computed in the moving coordinate system of the root node. We use exponential maps for the rotation variable $\mathbf{r}$ because it allows gimbal lock-free, approximately linearized parameterization.

Each helper bone is attached to the primary skeleton as a child of a primary bone, i.e., the $\phi(h)$-th primary bone is regarded as a parent of the h-th helper bone. The skinning matrix $\mathbf{S}_h$ is therefore

composed of the local transformation $\mathbf{L}_h$ and the parent's global transformation as $\mathbf{S}_h = \mathbf{G}_{\phi(h)}\mathbf{L}_h\bar{\mathbf{G}}^{-1}_{\phi(h)}$ by assuming $\mathbf{L}_h$ is identity at the rest pose.

Our control model imposes rigidity constraints on helper bone transformation, i.e., the local transformation $\mathbf{L}_h$ is composed of only translational and rotational components, which are denoted by $\mathbf{t}_h$ and $\mathbf{r}_h$, respectively, as

$$\mathbf{L}_h = \mathbf{m}\left(\mathbf{q}_h\right) \ , \ \mathbf{q}_h := \left[\begin{array}{c} \mathbf{t}_h \\ \mathbf{r}_h \end{array} \right] \in \Re^6 \tag{7}$$

where $\mathbf{m} : \Re^6 \rightarrow \Re^{4\times4}$ is the function that composes a transformation matrix from the rigid transformation components $\mathbf{q}_h$. The helper bone controller is then defined as a function with respect to the pose of the primary skeleton as $\mathbf{q}_h(\mathbf{u})$.

We also assume that skin deformation is modeled as the concatenation of static and dynamic deformation [Park and Hodgins 2008]; the former is determined according to the pose of the primary skeleton at the current time, and the latter depends on the history of both skeleton motion and skin deformation over the past time steps. The skinning transformation of a helper bone $\mathbf{q}$ is therefore represented by the concatenation of a static component $\mathbf{x}$ and a dynamic one $\mathbf{y}$; $\mathbf{q} = \mathbf{x} + \mathbf{y}$ (for now, we omit the subscript h for indexing each helper bone because the following computations are separately applied). The static transformation $\mathbf{x}$ is computed according to the skeleton pose. The dynamic transformation $\mathbf{y}$, on the other hand, is controlled using a state-space model which takes into account cumulative information of past skeletal poses and helper bone transformations.

4.2 Overview of learning

Helper bone controllers $\mathbf{x}(\mathbf{u})$ and $\mathbf{y}(\mathbf{u})$ are estimated to accurately approximate the discrete-time skinning matrix $\mathbf{S}^*_\tau$ which is the output of Equation 2. Given the parent bone $\phi(h)$, the local transformation component $\mathbf{q}^*_\tau$ is analytically extracted from the skinning matrix $\mathbf{S}^*_\tau$ as the reference by $\mathbf{q}^*_\tau = \mathbf{m}^{-1}(\mathbf{G}^{-1}_{\phi(h),\tau}\mathbf{S}^*_\tau\bar{\mathbf{G}}_{\phi(h)})$. The static controller $\mathbf{x}(\mathbf{u}) : \Re^{\dim(\mathbf{u})} \rightarrow \Re^6$ is first learned so as to best approximate these discretized references: $\mathbf{x}(\mathbf{u}_\tau) \approx \mathbf{q}^*_\tau, \forall\tau \in \mathcal{T}$ where $\mathbf{u}_\tau$ denotes the skeletal pose vector of Equation 6 at the τ-th frame. The residual between the reference transformation and the static component $\tilde{\mathbf{y}}_\tau = \mathbf{q}^*_\tau - \mathbf{x}(\mathbf{u}_\tau)$ is used to learn the dynamic controller $\mathbf{y}(\mathbf{u}) : \Re^{\dim(\mathbf{u})} \rightarrow \Re^6$.

The optimal parent bone $\phi(h)$ is exhaustively searched among all primary bones so as to minimize the sum of the squared norm of the residual $|\tilde{\mathbf{y}}_\tau|^2_2$ over the entire training sequence. Although this approximation neglects the fitness for dynamic components, the resulting accuracy is acceptable in practice because the static deformation is usually dominant over the dynamic one, especially for a slow and smooth skeletal motion.

4.3 Static controller

The static controller of helper bone is modeled as a polynomial regression function with respect to the primary skeleton pose $\mathbf{u}$ [Mukai 2015]. This simple linear model provides an efficient computation that can be further accelerated by imposing a sparsity constraint using the LASSO technique [Tibshirani 2011]. Let $\boldsymbol{\gamma}_p \in \Re^{{}_4\mathrm{H}_\chi - 1}$ be composed of variable terms of the χ-th order polynomial of $\{\mathbf{r}_p, 1\}$, where ${}_n\mathrm{H}_\chi$ denotes the number of combinations with repetitions for χ. For example, we obtain the second-order polynomials $\boldsymbol{\gamma} = [r_1,\ r_2,\ r_3,\ r_1^2,\ r_2^2,\ r_3^2,\ r_1r_2,\ r_1r_3,\ r_2r_3]$ for $\{\mathbf{r}, 1\} = \{r_1, r_2, r_3, 1\}$. The static component $\mathbf{x}(\mathbf{u})$ is then expressed as

$$\mathbf{x}(\mathbf{u}) = \mathbf{F}\boldsymbol{\Gamma} \ , \ \boldsymbol{\Gamma} := \left[1, \boldsymbol{\gamma}_1, \cdots, \boldsymbol{\gamma}_{|\mathcal{P}|}\right]^T \ , \tag{8}$$

where $|\cdot|^T$ denotes the transpose of the matrix, $\boldsymbol{\Gamma}$ is the concatenated independent vector composed without the root velocities $\triangle\mathbf{t}_0$ and $\triangle\mathbf{r}_0$, and $\mathbf{F} \in \Re^{6\times\dim(\boldsymbol{\Gamma})}$ is the coefficient matrix whose dimension is usually large, e.g., 6×190 for the polynomial order $\chi = 3$ and the number of primary bones $|\mathcal{P}| = 10$. The matrix $\mathbf{F}$ is, however, inherently sparse because of the local region of influence by each bone. We therefore convert $\mathbf{F}$ into a sparse coefficient matrix for optimally reducing the number of non-zero entries. Given the reference transformation $\mathbf{q}^*_\tau$ and the matrix $\boldsymbol{\Gamma}_\tau$ constructed from the primary bone rotation $\mathbf{r}_{p,\tau}$ at the τ-th frame, the sparse solution $\mathbf{F}^*$ can be approximately obtained by solving the l_1-norm optimization problem as

$$\mathbf{F}^* = \underset{\mathbf{F}}{\mathrm{argmin}} \left(\sum_{\tau\in\mathcal{T}} |\mathbf{q}^*_\tau - \mathbf{F}\boldsymbol{\Gamma}_\tau|^2_2 + \alpha\,|\mathbf{F}|_1 \right) \ , \tag{9}$$

where α is the positive shrinkage parameter that controls the trade-off between model accuracy and the number of non-zero entries. We refer readers to [Mukai 2015] for details about the effect of α. This Lasso problem can be solved using a stock solver. The training data of dynamic component $\tilde{\mathbf{y}}_\tau$ is then obtained by the residual between the static component and reference transformation as $\tilde{\mathbf{y}}_\tau = \mathbf{q}^*_\tau - \mathbf{F}^*\boldsymbol{\Gamma}_\tau$.

4.4 Dynamic controller

The dynamic transformation of the helper bones $\mathbf{y}_\tau$, which is the residual from Section 4.2, is assumed to be expressed as a linear time-invariant system represented by the SSM as

$$\left[\begin{array}{c} \mathbf{z}_{\tau+1} \\ \mathbf{y}_\tau \end{array} \right] = \left[\begin{array}{cc} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{array} \right] \left[\begin{array}{c} \mathbf{z}_\tau \\ \mathbf{u}_\tau \end{array} \right] \ . \tag{10}$$

System matrices $\mathbf{A} \in \Re^{k\times k}$, $\mathbf{B} \in \Re^{k\times\dim(\mathbf{u})}$, $\mathbf{C} \in \Re^{\dim(\mathbf{y})\times k}$ and $\mathbf{D} \in \Re^{\dim(\mathbf{y})\times\dim(\mathbf{u})}$ represent the mapping from the current internal state $\mathbf{z}_\tau \in \Re^k$ and pose $\mathbf{u}_\tau$ to the next state $\mathbf{z}_{\tau+1}$ and the current dynamic transformation $\mathbf{y}_\tau$, where k represents the reduced dimension of a system matrix (see Appendix A). The k-dimensional state vector $\mathbf{z}_\tau$ contains the minimal subset of cumulative information of past inputs and outputs, which can be interpreted as a reduced representation of internal dynamics such as momentum and potential energy. This SSM-based dynamic controller does not require rotational velocity nor acceleration of primary bones, since the state vector implicitly contains information of such model-specific dynamics. Moreover, our learning method estimates its minimum dimensionality k from the training data.

The system identification method estimates an SSM to describe unknown dynamic systems from measured input/output data. We here briefly review the standard system identification, and explain later about the computation of $\mathbf{A}$, $\mathbf{B}$, $\mathbf{C}$ and $\mathbf{D}$ in Appendix A. Please refer to [Ljung 1999] for more details. The subspace system identification method uses singular value decomposition to estimate the minimum order of the state vector and system matrices [Moor et al. 1988]. The basic approach of the subspace method constructs the matrix $\boldsymbol{\Theta}$ from input/output data. This is given by Equation A.2 in Appendix A, and contains the essential information of state-to-output transfer. The state-to-state transfer matrix $\mathbf{A}$ and the state-to-output transfer matrix $\mathbf{C}$ are then estimated via truncated singular value decomposition (or TSVD) of the matrix $\boldsymbol{\Theta}$. The remaining system matrices $\mathbf{B}$ and $\mathbf{D}$ are uniquely determined by solving a least square problem using the estimated $\mathbf{A}$ and $\mathbf{C}$.

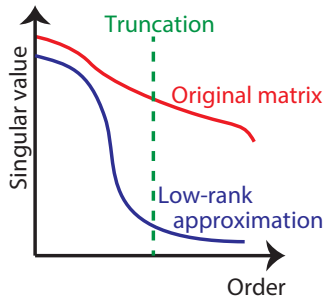

Figure 2: *Distribution of singular values. Sharper fall-off is observed in low-rank approximation via nuclear norm minimization.*

Nuclear norm system identification The standard subspace method determines the dimension of the internal state vector $\mathbf{z}$ and that of the system matrices by discarding the smallest singular values of $\mathbf{\Theta}$. Although more compact bone controllers can be obtained by decreasing the dimension of the state vector while discarding small singular values, an excessive truncation causes significant loss of prediction accuracy, especially when singular values of $\mathbf{\Theta}$ gradually decrease in order, as shown by the red curve in Figure 2. The matrix rank reduction method is therefore required to obtain the low-rank matrix whose singular values have sharply fallen off as shown by the blue curve. Ideally, a matrix rank should be minimized while keeping the approximation error within some threshold. However, such minimization is generally an NP-hard problem [Recht et al. 2010].

Several recent studies on system identification have shown that the nuclear norm is an alternative heuristic for relaxing the matrix rank minimization, which can be regarded as a generalization of l_1-norm optimization [Liu and Vandenberghe 2009]. We employ a nuclear norm optimization for system identification (or N2SID) [Liu et al. 2013], which provides a more flexible and stable solution than other methods [Liu and Vandenberghe 2009; Recht et al. 2010; Liu and Vandenberghe 2009]. Let $\mathbf{\Upsilon} := \mathbf{y}_1 \parallel \mathbf{y}_2 \parallel \cdots \parallel \mathbf{y}_{|\mathcal{T}|}$ be the concatenated vector composed of system outputs $\mathbf{y}_\tau$, and N2SID forms the following minimization problem:

$$\mathbf{\Upsilon}^* = \underset{\mathbf{\Upsilon}}{\operatorname{argmin}} \left(|\mathbf{\Theta}(\mathbf{\Upsilon})|_{\mathcal{N}} + \beta \sum_{\tau \in \mathcal{T}} |\tilde{\mathbf{y}}_\tau - \mathbf{y}_\tau|_2^2 \right) , \qquad (11)$$

where β is the positive shrinkage parameter, and $|\cdot|_{\mathcal{N}}$ denotes the sum of singular values, called a nuclear norm (or trace norm). This objective is to find the optimized output $\mathbf{y}_\tau^*$ that reduces the rank of $\mathbf{\Theta}(\mathbf{\Upsilon})$, while maintaining the variables close to the training data $\tilde{\mathbf{y}}_\tau$ by which a sparse solution of internal state $\mathbf{z}_\tau$ is obtained. This minimization problem is formulated as a semidefinite programming which can be solved using a convex optimization solver. Although the nuclear norm optimization does not guarantee the minimality of the matrix rank, empirical observations in system identification studies have shown that it often provides effective low-rank approximation. Please refer to [Liu et al. 2013] for details regarding nuclear norm optimization for system identification.

System matrices are finally estimated via TSVD of $\mathbf{\Theta}(\mathbf{\Upsilon}^*)$. The accuracy of the estimated model is competitive with existing subspace identifications because it preserves linear structure in low-rank approximation. Naive TSVD and the other dimensionality reduction methods, such as probabilistic PCA and non-negative matrix factorization, destroy the structure of the matrix $\mathbf{\Theta}$ in estimating its low-rank approximation. In contrast, the nuclear norm method optimizes only output variables while preserving the linear structure of the matrix. This property is important for effective system identification as explained in Appendix A.

Heuristic stabilization The solution of N2SID becomes unstable if at least one eigenvalue of the system matrix $\mathbf{A}$ is greater than or equal to one, which causes divergence of the states. The most typical heuristic to make $\mathbf{A}$ divergence-free is to invert such unstable eigenvalues [Liu 2009]. Let $\mathbf{A} = \mathbf{Q}^{-1}\mathbf{T}\mathbf{Q}$ be the Schur decomposition of $\mathbf{A}$ where $\mathbf{Q}$ is a unitary matrix, $\mathbf{Q}^{-1}$ is its conjugate transpose, and $\mathbf{T}$ is an upper triangular matrix whose diagonal elements are the eigenvalues of $\mathbf{A}$. All eigenvalues greater than one are replaced by their inverse to update $\mathbf{A}$. This process is repeated until all eigenvalues become less than one. Although a stabilization method based on semidefinite programming [Mari et al. 2000] would further improve model accuracy, this simple heuristic worked out well in all of our experiments.

Component-wise control The computational cost of nuclear norm optimization increases according to the dimension of the matrix $\mathbf{\Theta}, \propto \dim(\mathbf{y}_\tau) \times (\dim(\mathbf{y}_\tau) + \dim(\mathbf{u}_\tau))$, where our experimental data have $\dim(\mathbf{u}_\tau) = 15$. Naive implementation took over an hour to solve the optimization in computing all output variables of $\dim(\mathbf{y}_\tau) = 6$ at once, which is impractical and often unstable due to its large dimensions. We therefore compute each component of $\mathbf{y}_\tau$ separately on each individual dynamic controller, by which the dimension is reduced to $\dim(\mathbf{y}_\tau) = 1$. We experimentally confirmed that each controller learning was stably computed within minutes. Such component-wise modeling neglects the correlation among the variables of 3D translation and rotation, by which a redundant controller may be unintentionally constructed. Although this redundancy does not damage the accuracy, a more compact controller could be built by simultaneously analyzing all transformation variables with a more robust and efficient semidefinite programming solver.

5 Experimental Evaluations

We evaluated approximation capability and computational performance using training datasets of dynamic skin deformations. All experiments set up the maximum number of blended transformations to $\kappa = 4$, and third-order polynomial $\chi = 3$ is used for all static controllers. The reconstruction error was evaluated using root mean squared error (or RMSE) of the vertex position. The learning of the helper bone controller was parallelized over vertices, helper bones, and samples using Intel Threading Building Blocks. We used a Python implementation of the interior-point method, distributed as a part of CVXOPT [Andersen et al.], for nuclear norm approximation. The runtime execution time of the helper bone controllers was measured on a workstation with Dual Intel Xeon E5-2687W CPUs (40 logical processors) at 3.1GHz and 192 GB RAM.

5.1 Performance evaluation

Sample data We used a muscle function of Autodesk Maya 2016 to generate samples of dynamic skin deformation driven by skeleton motion. The muscle system simulates muscle-skin dynamic deformation caused by primary skeleton motion, bone acceleration, and inertia of soft tissues. A monster's leg asset of a Maya tutorial, supplied by Autodesk Inc., was used for training and evaluation (Figure 3). The skeleton has 3 animating primary bones and 5 degrees of freedom (DOFs) including hip swing and twist (3 DOFs), knee bend (1 DOF) and ankle bend (1 DOF). The eleven muscles expand and contract according to movement of the primary skeleton, and drive the deformation of 552 vertices using the proprietary algorithm.

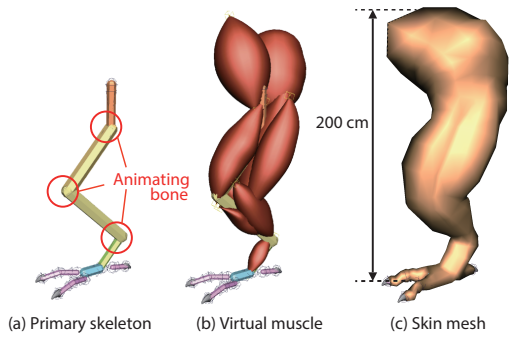

Figure 3: *Monster leg model used for performance evaluation. The skin deformation is driven by transformations of virtual muscles.*

Sample data of about 4000 frames were created by randomly sampling joint rotation angles of the primary skeleton at 250 randomly selected keyframes to cover the wide range of dynamics. We used common spline-based keyframe interpolation for the first half of the sequence, and the last half was interpolated using a keyframe freezing technique that holds a keyframe value until the next keyframe. Corresponding skin vertex animation was finally simulated using the built-in muscle function.

Computational performance Table 1 shows the computational performance by using the relationship between the number of helper bones and reconstruction errors, runtime execution time, average dimensionality of the state vector, and data size. The increase in the number of helper bones leads to the increase of execution time and data size in a monotonic manner. Nevertheless, they are efficient enough to ensure real-time responses. The static controller requires less computational cost than the dynamic one since SSM has more parameters than the polynomial function.

On the other hand, the approximation error was not monotonically decreased with respect to the number of helper bones. The lowest RMSE is obtained for 4 helper bones in spite of the low accuracy of the corresponding static controller at the same time. This shows the larger compensational effect from the dynamic deformation than the static one. This result also implies that the static controller partly reflects the secondary dynamics in training data. Our ad-hoc approach, which trains both controllers from the same data, does not guarantee pure separation of the static component, i.e., the static controller involves frequently-appearing patterns of the secondary deformations. This property contributes efficiency because the static controller is computed faster than SSM and is convenient for level-of-detail controls as explained in Section 5.5.

Table 2 summarizes the time required to construct the helper bone rig from training data. The computational time proportionally increases with the number of helper bones. The most expensive part is the dynamic controller learning that solves semidefinite programming. This computational cost is quadratic in the duration of the training sequence and the dimensionality of input/output data.

Figure 4 visualizes the initial configuration and skinning weight map for each of the 6 helper bones, with each of them located at the position of the corresponding parent bone. The helper bone (a) has a large influence on the skin and mostly contributes to the dynamic deformation around hip and thigh. The helper bones (b) and (e) correct the deformation of hip and thigh, and (c), (d) and (e) drive the deformation of lower leg. For example, the skinning weight of two helper bones (a) and (b) is distributed around the same area behind the knee, by which complex skin deformation was produced with multiple helper bones.

Table 1: *Statistics for the various numbers of helper bones. The learning parameters were set to $\alpha = 10$ and $\beta = 10^2$. Two kinds of reconstruction errors were measured: with only the static control (upper) and with both the static and dynamic control (lower), respectively. The execution time (left in each cell) is divided into those for the static (upper right) and dynamic components (lower right).*

# of helper bones	RMSE (cm)	Execution time (μs/frame)	avg. $\dim(\mathbf{z})$	Data size (KB)
0	5.18	-		-
1	2.92	4.3 / 1.7	3.83	2.2 / 0.3
	2.83	/ 2.6		/ 1.9
2	2.91	15.9 / 2.0	3.33	3.9 / 0.6
	2.73	/ 13.9		/ 3.3
3	2.88	17.9 / 2.6	3.39	6.0 / 0.9
	2.64	/ 15.3		/ 4.1
4	2.92	19.5 / 3.3	3.25	8.0 / 1.1
	2.47	/ 16.2		/ 6.9
5	2.83	21.5 / 3.6	3.37	9.9 / 1.5
	2.55	/ 17.9		/ 8.4
6	2.81	22.8 / 4.0	3.22	11.5 / 1.8
	2.53	/ 18.8		/ 9.7

Table 2: *Computational time of rig construction (in seconds). The learning process of a static controller includes the exhaustive search for an optimal parent bone.*

# of helper bones	Total	Skinning decomposition	Controller learning	
			Static	Dynamic
1	51.0	3.0	1.4	46.6
2	97.7	4.3	2.8	90.7
3	141.3	5.6	4.1	131.6
4	186.7	7.8	5.8	173.1

Effect of nuclear norm optimization The effectiveness of nuclear norm optimization was evaluated using various settings of shrinkage parameter β. We also used the numerical algorithm for system identification (or N4SID) [Ljung 1999] as a baseline, which is the most widely-used system identification method based on TSVD.

Table 3 summarizes the performance with respect to shrinkage parameter β for 4 helper bones. About 70% of SSM data was reduced from the N4SID method for $\beta = 10^2$. The computational performance of bone controller was improved as β decreased. However, the increases of the approximation error, computational time and data size are observed around $\beta = 10$, which indicates that the es-

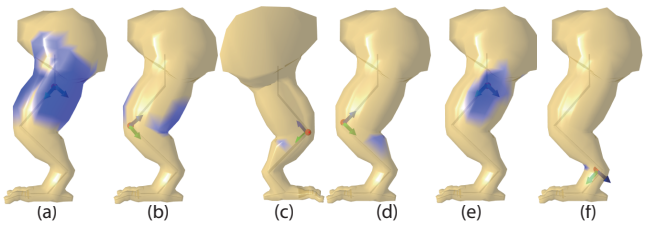

Figure 4: *Distribution map of skinning weight of each helper bone by coloring the area of larger weights with a darker blue, where the figure in (c) is rotated to effectively display the distribution around the inner calf.*

Table 3: *Performance for various settings of the shrinkage parameter β of N2SID.*

Method		RMSE (cm)	Execution time (μs/frame)	avg. $\dim(\mathbf{z})$	Data size (KB)
N4SID		3.07	22.5	9.17	51.0
N2SID	$\beta = 10^5$	2.50	19.3	3.21	7.3
	$\beta = 10^4$	2.49	19.2	3.08	7.1
	$\beta = 10^3$	2.48	19.2	3.04	7.0
	$\beta = 10^2$	2.47	19.5	3.25	8.0
	$\beta = 10$	2.49	20.4	4.04	9.0

sential information of the system input/output was lost by excessive reduction of the nuclear norm. Although the best shrinkage parameter β is yet unknown, this result suggests that the accuracy has little impact on these performance indicators. We therefore trained helper bones with the learning parameters of $\alpha = 10$ and $\beta = 10^2$ for all experiments. Note that some heuristics, such as Akaike information criterion and Bayesian information criterion, are possible tools for optimally determining these learning parameters to balance the reconstruction accuracy and the computational cost.

Generalization performance For evaluating generalization performance, we created two additional sequences of 4000 frames using the abovementioned procedure to test the 4 trained helper bones. The average RMSE was 2.43 cm, which is slightly better than the measurement obtained by using training data as testing ones ($=$ 2.47 cm). This high generalization performance is made possible by having a sufficient amount of training data, since the accuracy of SSM is increased monotonically by using a longer training sequence in N2SID [Liu et al. 2013].

We also evaluated the generalization ability of our method for the same monster's leg model, using the 4 trained helper bones. Two types of animation data were used for testing: one is hand-crafted to contain only slow and smooth motions and the other to include rapid stretching and stamping motions. These behaviors did not appear in the training data. The RMSE was 1.69 cm for the smooth motions, which is smaller than those obtained with the procedurally generated test data. On the other hand, the RMSE for the rapid motions was increased to 3.06 cm. This result suggests the lower generalization capability of our dynamic controller because the near-static slow motions are reproduced only with static controllers and the rapid motions mainly with dynamic controllers. The errors caused by dynamic controllers, however, are still small enough and acceptable because no serious artifact occurs, thanks to the stability of SSMs.

Robustness for behavioral modifications Figure 5(b) shows the modified version of the monster leg model (Figure 5(a)), created by tweaking the muscle parameter to magnify the bulging deformation and to make the skin softer. The skin mesh was smoothened by subdividing polygonal faces in order to evaluate detailed deformation. The training data were created by randomly sampling keyframes in the same way as the first experiment.

We built 4 helper bones and their controllers as seen in Figure 5(d). The helper bone rig reproduces the overall behavior of soft skin well. The resulting RMSE was 2.60 cm, which was larger than measurements from previous experiments due to the increase of muscle softness. Especially, self-penetration and collapse of the skin mesh were often caused when heavy movement was applied to the primary skeleton as shown in the supplemental video. Maya muscle function simulates the complex skin deformation accompanied by the self-collisions of skin surface. Such highly-nonlinear, local behavior is difficult to approximate by polynomial- or SSM-based control with the sparse set of helper bones. Undesirable artifacts could be reduced by limiting the range of helper bones' motions; accurately estimating the safe range, however, is an open question.

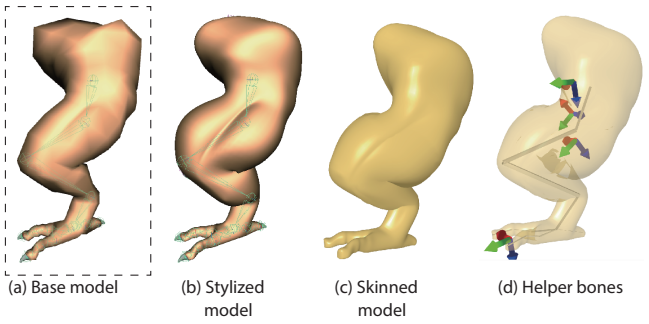

Figure 5: *Modified version of monster's leg.*

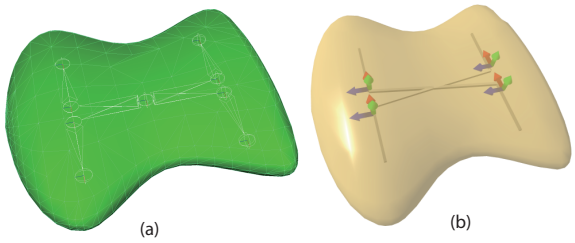

Figure 6: *(a) Non-articulated model with H-shaped skeleton which drives the skin deformation using a mass-spring simulation. (b) Approximation with 4 helper bones.*

The execution time and data size for all helper bone controllers were 19.2 μs per frame and 6.9 KB, respectively. Both values depend on the numbers of primary and helper bones and the dimension of the helper bone controllers. The number of skin vertices affects neither the execution time of the helper bone controller nor its memory requirement. On the other hand, the rig construction took 72.0, 5.5, and 172.0 seconds for skinning decomposition, static controller learning, and dynamic controller learning, respectively. Notice that increasing mesh resolution increases the cost of skinning decomposition without affecting the controller learning.

Flexibility in rig structure and simulation model Our helper bone rig can be built for non-articulated elastic models as shown in Figure 6. An H-shaped skeleton ($|\mathcal{P}| = 4$) was embedded in this flat soft body whose training deformations were generated using a simple mass-spring simulation. The elastic behavior of the soft model was reproduced with 4 helper bones to exhibit long-period vibrations. This shows the capability of our SSM-based controller for well-imitating the deformations of different types of rig structures and dynamics property as demonstrated in the supplemental video.

5.2 Comparison against ARX model

We have compared our SSM-based dynamic controller model against the ARX-based one [de Aguiar et al. 2010; Pons-Moll et al. 2015; Loper et al. 2015] using the same sample data of the monster's leg. The standard formulation of the ρ-th order ARX model

Table 4: *Comparison with ARX models.*

	SSM	ARX		
		2nd	3rd	4th
RMSE (cm)	2.47	4.40	4.53	4.50
Data (KB)	8.0	4.0	5.3	6.6

is expressed for the dynamic component $\mathbf{y}(\mu)$ as

$$\mathbf{y}(\mu) = \mathbf{\Psi}\boldsymbol{\mu}\ ,\ \boldsymbol{\mu} := \mathbf{y}_{\tau-\rho} \parallel \cdots \parallel \mathbf{y}_{\tau-1} \parallel \mathbf{u}_{\tau-\rho+1} \parallel \cdots \parallel \mathbf{u}_{\tau}\ , \tag{12}$$

where $\boldsymbol{\mu} \in \Re^{\rho\cdot(\dim(\mathbf{y})+\dim(\mathbf{u}))}$, and $\mathbf{\Psi} \in \Re^{6\times\rho\cdot(\dim(\mathbf{y})+\dim(\mathbf{u}))}$ is the coefficient matrix composed of the autoregression term and the exogenous input term, and it is optimized by solving the least square problem.

Table 4 summarizes the reconstruction error and data size for second-, third-, and fourth-order ARX models. This result verifies the superior performance of the SSM and N2SID compared to the ARX model. From the viewpoint of visual appearance, ARX models cannot maintain jiggly movements; they only cause subtle and slowly converging jiggles (see the supplemental video). This defect might be derived from the general property of the ARX model, as it is excessively sensitive to redundancy and noise in the input/output signals. Existing methods therefore extend the input variables of ARX models by adding joint rotation velocities and accelerations, and use a dimensionality reduction technique to identify a low-dimensional subspace of deformation dynamics. In contrast, our method works without such extensions, i.e. only skeleton pose $\mathbf{u}$ is required to control the helper bones, because the state vector $\mathbf{z}$ implicitly holds the minimal subset of the internal dynamic state such as momentum and inertia. In addition, nuclear norm optimization further minimizes the redundancy of state variables and system matrices.

5.3 Heterogeneous soft tissue

We here conduct a test on a practical example of a human character whose heterogeneous muscle structure was made from different materials. The human model was composed of 11356 vertices, 19 primary bones, and 328 virtual muscles of various elasticities as shown in Figure 7(a). We approximated this muscle rig using 1, 2, and 4 helper bones, and each construction time was 124, 205, and 362 seconds, for investigating which body parts were preferentially compensated for by adding helper bones.

Table 5 summarizes the statistics of the learning result. The skinning weights accumulated over all helper bones are visualized in Figure 7(b), (c) and (d). The first helper bone covered the wide area around shoulders and fatty parts like upper body and hip (Figure 7(b)), neither deformation which could be reconstructed using naive LBS. The additional second helper bone refined the distribution of skinning weights to improve the deformation quality around the same area (Figure 7(c)). The case of four helper bones exhibited more detailed weight distribution especially on shoulders, and also distributed around the legs (Figure 7(d)). This result indicates that our learning algorithm of the bone controller successfully detected the heterogeneous distribution of soft tissues.

Figure 8 visualizes the error of the vertex position obtained with 4, 6 and 8 helper bones at the frame of the largest RMSE in the training sequence. As the heat maps indicate, the reconstruction errors were not always consistently decreased according to the number of helper bones. Our method minimizes the global error over all vertices and over the entire training sequence; however, it often causes an increase in the error of local regions. Although this defect is negligible in most cases, an alternative controller learning method might be explored to ensure both global and local optimality.

Table 5: *Performance for various numbers of helper bones.*

# of helper bones	RMSE (cm)	Execution time (μs/frame)	avg. $\dim(\mathbf{z})$	Data size (KB)
1	0.41	17.8	3.17	9.6
2	0.40	22.5	3.83	21.5
4	0.39	27.8	3.44	36.3

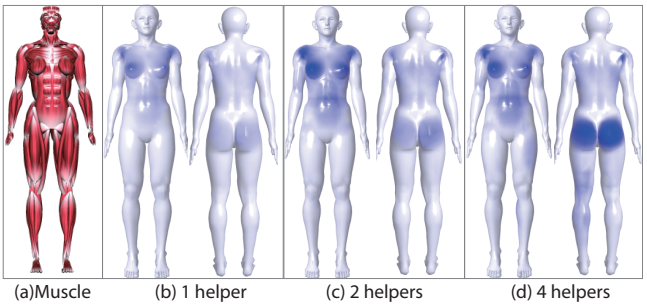

Figure 7: *(a) Muscle rig of human-like character. (b)-(d) Accumulated skinning weight of all helper bones where the area of larger weights is indicated by a darker blue.*

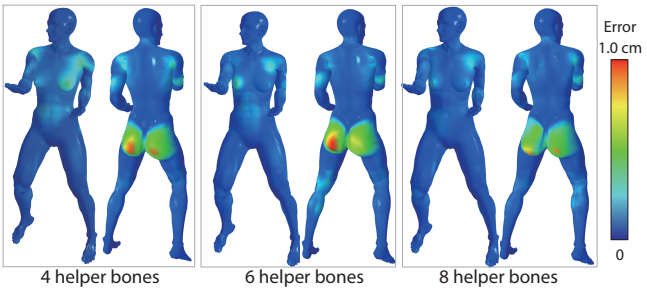

Figure 8: *The heat map shows the error from the ground truth where blue and red mean* 0 *and* $\geq$ 1 *cm, respectively.*

5.4 Exaggeration of dynamic deformations

We can exaggerate certain components of the dynamic transformation of a helper bone by simply scaling the corresponding component of $\mathbf{y}$. Our helper bone controller allows designers to intuitively control the effect of soft-tissue dynamics. This operation does not affect the stability and efficiency of the bone controller because no feedback is supplied from the output $\mathbf{y}$ to the internal state $\mathbf{z}$. On the other hand, exaggeration for static deformation component $\mathbf{x}$ is often difficult due to its nonlinear relation to the primary skeleton pose $\mathbf{u}$.

We evaluated these suppressing or exaggerating effects for the monster leg model with 4 helper bones by uniformly scaling all dynamic components $\mathbf{y}$ by 0.5, 1.0, 2.0, 4.0 and 6.0. As demonstrated in the supplemental video, this simple technique can provide intuitive results without any extra computations. Excessive increase of the scaling ratio, however, caused collapse of skin surface, as shown in Figure 9, because our purely signal-based approach neglects the physical phenomenon of deformed shapes. Despite such a drawback, this signal-based technique is practical for creating a variety of skin stiffness from a single set of training data.

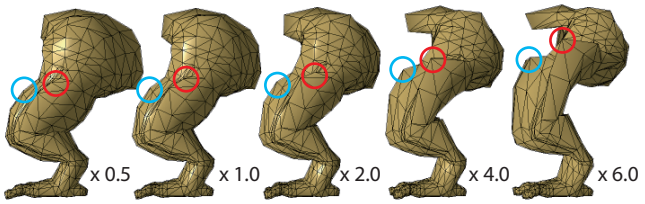

Figure 9: *Deformation exaggeration by scaling dynamic controller output where each of the blue and red circles encloses the vertex of the same index for comparing their movements according to the scales of exaggeration. Dynamic deformation around the thigh was generated by sharp downswing of the character and the inertia of soft tissues. Excessive exaggeration yields the collapse of skin mesh.*

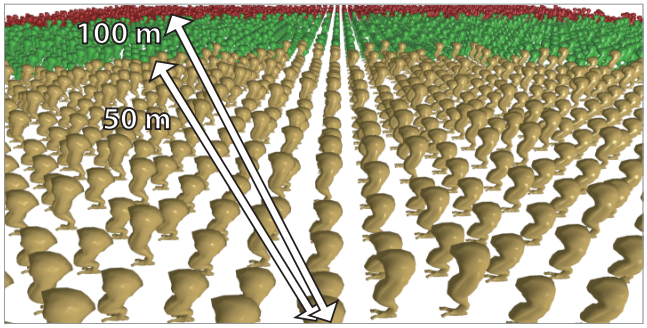

Figure 10: *A massive crowd scene synthesis with LOD control on skin deformation. Yellow, green and red models activated both dynamic and static controller, only static controller, and neither controller, respectively.*

5.5 Deformation level-of-detail controls

The level of-detail (or LOD) control of skin deformation can be easily implemented by simply switching on/off the bone controllers. The dynamic control of the helper bone is disabled in higher priority if its visual appearance is unperceivable on the screen. Static control is disabled next if the character is distant from view. This three-level control was applied to a massive crowd scene including 4900 monster leg models. Characters were arranged on the regular grid of 2 meters on each side. The static controller was activated to the character in the view frustum within 100 meters radius from the viewpoint, and the dynamic controller was combined for characters within 50 meters. Both controllers were disabled for the other distant characters. We measured execution time per frame in synthesizing the local transformation of the helper bones of all characters.

The average execution time was 1.72 ms for 700 fine-level characters and 1700 middle-level ones, as shown in Figure 10. The 900 fine-level visible characters required 6.9 ms without LOD controls as a worst case, and the computation time was reduced to 2.11 ms with the LOD control. This shows that redundant computation for skin deformation was effectively reduced by our LOD control technique.

6 Discussion

We have proposed a practical method for synthesizing plausible skin deformation including the effects of soft-tissue dynamics. Our dynamic skinning with the helper bone controller has the following advantageous properties:

Efficiency Low-rank approximation of a linearly structured SSM with nuclear norm optimization allows simpler and faster runtime computations.

Stability Stability of the bone controller is guaranteed by introducing a divergence-free system matrix through heuristic modifications.

Simplicity SSM and N2SID methods require few parameters to be tuned, whereas the ARX model requires a careful decision of the model order and selection of input variables.

Flexibility The controller learning is intrinsically signal-based, which allows various dynamical behaviors to be flexibly imitated without any extension of the controllers.

Portability Skin deformation can be efficiently implemented on GPU using the standard LBS procedure, and the general matrix-vector formulation of the SSM ensures portability.

To the best of our knowledge, our method is the first to introduce the nuclear norm optimization method to data-driven animations. This method has the potential to be applied to a wide variety of regression-based animation syntheses for minimizing model complexity. For instance, our rank-reduced, SSM-based dynamic controller would be a better alternative to traditional ARX models used in corrective shape-based techniques [de Aguiar et al. 2010; Loper et al. 2015; Pons-Moll et al. 2015].

Our signal-based learning approach has the potential for imitating physically-invalid deformations hand-crafted by animators, although the training data were generated using physics-based simulations in all experiments. However, estimating the expressive space reproducible by our controller model is an open question.

Since the helper bone is a general-purpose rig, our method can be applied to various types of skeleton-driven characters including non-articulated animals and creatures. Also, our example-based learning of helper bone controllers can be applied to a substitute for other types of computationally-expensive rigs. We experimentally confirmed that the behaviors of muscle rigs with heterogeneous structure can be successfully imitated by a lighter-weight helper bone rig. It is challenging to simultaneously construct the skeleton and helper bone rigs from a sequence of simulated or captured shape animations [Neumann et al. 2013; Pons-Moll et al. 2015] in a fully-automated way.

Limitations and future work Our helper bone has limitations on the interactions between the skin and other objects or against external forces because it is controlled only by the primary skeleton. For example, our helper bone is not applicable to the deformation of feet caused by ground contact. Furthermore, our method restricts each helper bone to be bound to a single primary bone as a parent, i.e., a helper bone is not allowed to be a child of another helper bone. This precondition cannot allow the dynamics propagating along a long thin model such as an animal's tail.

Our method does not ensure global optimality for the skinning weights and the helper bone controller. We have found that an increase in the number of helper bones often degrades the reconstruction, because numerical errors are separately accumulated when solving optimizations for the skinning decomposition and bone controller learning. A more sophisticated algorithm should be investigated to simultaneously solve these problems.

Our SSM-based controller can be learned only from a single continuous sequence of skeleton motions and corresponding skin deformations because N2SID cannot simultaneously analyze different sequences. This limitation requires a training sequence of a length that exponentially proliferates for the number of primary bones.

Our system identification should be extended to handle multiple sequences. Moreover, our method currently lacks the generalization ability for deformations of different skin materials. This limitation might be overcome by using a mixture of multiple SSMs learned from sample data of various materials, like mixture of ARX models [Xia et al. 2015].

A helper bone is usually affected by spatially neighboring a few primary bones. Such a locality could be automatically detected by the l_1-norm optimization in learning the static bone controller. However, low-rank approximation of the N2SID method might not fully reflect the intrinsic sparsity of the linear system. This defect could be overcome by introducing sparsity constraints into the least square estimate of the system matrices.

Although most helper bone rigs have been adopted for the LBS-based technique, popular nonlinear blend skinning techniques such as dual quaternion skinning are worth investigating, and will be included in our future work.

Acknowledgements

We thank TAISO, Renpoo for making the human character model available at http://www.behind-universe.org/ and for many helpful comments. This work was supported by JSPS KAKENHI Grant Number 15K16110, 15H02704.

References

ANDERSEN, M. S., DAHL, J., AND VANDENBERGHE, L. CVXOPT: A python package for convex optimization. http://cvxopt.org.

ANGELIDIS, A., AND SINGH, K. 2007. Kinodynamic skinning using volume-preserving deformations. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2007*, 129–140.

DE AGUIAR, E., SIGAL, L., TREUILLE, A., AND HODGINS, J. K. 2010. Stable spaces for real-time clothing. *ACM Transactions on Graphics 29*, 4, 106:1–106:9.

FAN, Y., LITVEN, J., AND PAI, D. K. 2014. Active volumetric musculoskeletal systems. *ACM Transactions on Graphics 33*, 4, 152:1–152:9.

HAHN, F., MARTIN, S., THOMASZEWSKI, B., SUMNER, R., COROS, S., AND GROSS, M. 2012. Rig-space physics. *ACM Transactions on Graphics 31*, 4, 72:1–72:8.

HAHN, F., THOMASZEWSKI, B., COROS, S., SUMNER, R., AND MARKUS. 2013. Efficient simulation of secondary motion in rig-space. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2013*, 165–171.

HSU, E., PULLI, K., AND POPOVIĆ, J. 2005. Style translation for human motion. *ACM Transactions on Graphics 24*, 3, 1082–1089.

JAMES, D. L., AND PAI, D. K. 2002. Dyrt: Dynamic response textures for real time deformation simulation with graphics hardware. *ACM Transactions on Graphics 21*, 3, 582–585.

KAVAN, L., AND SORKINE, O. 2012. Elasticity-inspired deformers for character articulation. *ACM Transactions on Graphics 31*, 6, 196:1–196:8.

KAVAN, L., COLLINS, S., ZARA, J., AND O'SULLIVAN, C. 2007. Skinning with dual quaternions. In *Proceedings of ACM SIGGRAPH Symposium on Interactive 3D Graphics 2007*, 39–46.

KIM, J., AND KIM, C.-H. 2011. Implementation and application of the real-time helper-joint system. In *Game Developers Conference 2011*.

KRY, P. G., JAMES, D. L., AND PAI, D. K. 2002. Eigenskin: Real time large deformation character skinning in hardware. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2002*, 153–159.

LARIMORE, W. 1990. Canonical variate analysis in identification, filtering and adaptive control. In *Proceedings of Control & Decision Conference*, 596–604.

LE, B. H., AND DENG, Z. 2012. Smooth skinning decomposition with rigid bones. *ACM Transactions on Graphics 31*, 6, 199:1–199:10.

LE, B. H., AND DENG, Z. 2014. Robust and accurate skeletal rigging from mesh sequences. *ACM Transactions on Graphics 33*, 4, 84.

LEWIS, J. P., CORDNER, M., AND FONG, N. 2000. Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *Proceedings of SIGGRAPH 2000*, 165–172.

LI, D., SUEDA, S., NEOG, D. R., AND PAI, D. K. 2013. Thin skin elastodynamics. *ACM Transactions on Graphics 32*, 4, 49:1–49:9.

LIU, Z., AND VANDENBERGHE, L. 2009. Interior-point method for nuclear norm approximation with application to system identification. *SIAM Journal on Matrix Analysis and Application 31*, 3, 1235–1256.

LIU, Z., HANSSON, A., AND VANDENBERGHE, L. 2013. Nuclear norm system identification with missing inputs and outputs. *Systems & Control Letters 62*, 8, 605–612.

LIU, Z. 2009. *Structured Semidefinite Programs in System Identification and Control*. PhD thesis, University of California, Los Angeles.

LJUNG, L. 1999. *System Identification: Theory for the User*. Prentice Hall.

LOPER, M., AND BLACK, N. M. M. J. 2014. Motion and shape capture from sparse markers. *ACM Transactions on Graphics 33*, 6, 220:1–220:13.

LOPER, M., MAHMOOD, N., ROMERO, J., PONS-MOLL, G., AND BLACK, M. J. 2015. SMPL: A skinned multi-person linear model. *ACM Transactions on Graphics 34*, 6, 248:1–248:16.

MAGNENAT-THALMANN, N., LAPERRIÈRE, R., AND THALMANN, D. 1988. Joint-dependent local deformations for hand animation and object grasping. In *Proceedings on Graphics Interface '88*, 26–33.

MARI, J., STOICA, P., AND MCKELVEY, T. 2000. Vector ARMA estimation: a reliable subspace approach. *IEEE Transaction on Signal Processing 48*, 7, 2092–2104.

MOHR, A., AND GLEICHER, M. 2003. Building efficient, accurate character skins from examples. *ACM Transactions on Graphics 22*, 3, 562–568.

MOOR, B. D., MARC MOONEN AND, L. V., AND VANDEWALLE, J. 1988. A geometrical approach for the identification of state space models with singular value decomposition. In *International Conference on Acoustics, Speech, and Signal Processing*, vol. 4, 2244–2247.

MUKAI, T. 2015. Building helper bone rigs from examples. In *Proceedings of ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games 2015*, 77–84.

NEUMANN, T., VARANASI, K., HASLER, N., WACKER, M., MAGNOR, M., AND THEOBALT, C. 2013. Capture and statistical modeling of arm-muscle deformations. *Computer Graphics Forum 32*, 2, 285–294.

PARK, S. I., AND HODGINS, J. K. 2008. Data-driven modeling of skin and muscle deformation. *ACM Transactions on Graphics 27*, 3, 96:1–96:6.

PARKS, J. 2005. Helper joints: Advanced deformations on runtime characters. In *Game Developers Conference 2005*.

PONS-MOLL, G., ROMERO, J., MAHMOOD, N., AND BLACK, M. J. 2015. Dyna: A model of dynamic human shape in motion. *ACM Transactions on Graphics 33*, 4, 120:1–120:10.

RECHT, B., FAZEL, M., AND PARRILO, P. A. 2010. Guaranteed minimum-rank solutions of linear matrix equations via nuclear norm minimization. *SIAM Review 52*, 3, 471–501.

RUMMAN, N. A., AND FRATARCANGELI, M. 2015. Position-based skinning for soft articulated characters. *Computer Graphics Forum 34*, 6, 240–250.

SHI, X., ZHOU, K., TONG, Y., DESBRUN, M., BAO, H., AND GUO, B. 2008. Example-based dynamic skinning in real time. *ACM Transactions on Graphics 27*, 3, 29:1–29:8.

TIBSHIRANI, R. 2011. Regression shrinkage and selection via the lasso: A retrospective. *Journal of the Royal Statistical Society: Series B (Statistical Methodology) 73*, 3, 273–282.

VERHAEGEN, M. 1994. Identification of the deterministic part of MIMO state space models given in innovations form from input-output data. *Automatica 30*, 1, 61–74.

VIBERG, M. 1995. On subspace-based methods for the identification of linear time-invariant systems. *Automatica 31*, 12, 1835–1852.

WANG, X. C., AND PHILLIPS, C. 2002. Multi-weight enveloping: Least-squares approximation techniques for skin animation. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 129–138.

WANG, R. Y., PULLI, K., AND POPOVIĆ, J. 2007. Real-time enveloping with rotational regression. *ACM Transactions on Graphics 26*, 3, 73:1–73:10.

XIA, S., WANG, C., CHAI, J., AND HODGINS, J. 2015. Realtime style transfer for unlabeled heterogeneous human motion. *ACM Transactions on Graphics 34*, 4, 119:1–119:10.

A. Subspace method for system identification

We here briefly explain the subspace method for system identification [Liu et al. 2013]. The purpose of system identification is to estimate the SSM (Equation 10) that best describes the relation between observed input $\mathbf{u}_\tau$, i.e., the training sequence of the skeleton pose vector, and the output signals $\mathbf{y}_\tau$, i.e., the dynamic transformation of helper bone.

First, b-block Hankel matrices for input $\mathbf{U}_{(i)}$ and output $\mathbf{Y}_{(i)}$ are constructed from $\mathbf{u}_\tau$ and $\mathbf{y}_\tau$, $\tau \in \mathcal{T}$, respectively. Because both matrices are computed in a similar way, we here only provide the definition of $\mathbf{U}_{(i)}$ as

$$\mathbf{U}_{(i)} = \begin{bmatrix} \mathbf{u}_{i+1} & \mathbf{u}_{i+2} & \cdots & \mathbf{u}_{i+N} \\ \mathbf{u}_{i+2} & \mathbf{u}_{i+3} & \cdots & \mathbf{u}_{i+N+1} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{u}_{i+b} & \mathbf{u}_{i+b+1} & \cdots & \mathbf{u}_{i+N+b-1} \end{bmatrix}, \tag{A1}$$

where $\mathbf{U}_{(i)} \in \Re^{b\cdot\dim(\mathbf{u})\times N}$, N is the size of the sampling window, and b is the number of blocks that limits the maximum dimension of the internal state vector $\mathbf{z}_\tau$ as $\dim(\mathbf{z}_\tau) \leq b \cdot \dim(\mathbf{y})$, both of which are manually specified to satisfy $N \geq 2b\,(\dim(\mathbf{u}) + \dim(\mathbf{y}))$ and $N + 2b \leq |\mathcal{T}|$. Specifically, the number of blocks b has to be larger than the expected order of the SSM, and the larger size of the sampling window N usually provides more accurate estimation. We used $b = 10$ and $N = |\mathcal{T}| - 2b$ in all of our experiments.

The following matrix $\mathbf{\Theta}$ is a key component of the subspace method for estimating optimal SSM from input/output signals, which is given by

$$\mathbf{\Theta} = \mathbf{W}_1 \mathbf{Y}_{(b)} \mathbf{U}_{(b)}^\dagger \left[\mathbf{U}_{(0)}^T,\ \mathbf{Y}_{(0)}^T\right] \mathbf{W}_2\ , \tag{A2}$$

where $\mathbf{\Theta} \in \Re^{b\cdot\dim(\mathbf{y})\times b\cdot(\dim(\mathbf{u})+\dim(\mathbf{y}))}$. The matrix $\mathbf{U}_{(b)}^\dagger \in \Re^{N\times N}$ is the nullspace projection satisfying $\mathbf{U}_{(b)}\mathbf{U}_{(b)}^\dagger = \mathbf{0}$, which serves to cancel the effect of input signals from the output, and the concatenated matrix $\left[\mathbf{U}_{(0)}^T,\ \mathbf{Y}_{(0)}^T\right]$ is added to achieve more consistent estimation. Two square weight matrices $\mathbf{W}_1$ and $\mathbf{W}_2$ serve to improve estimation accuracy, and various algorithms for determining these values have been proposed [Larimore 1990; Verhaegen 1994; Viberg 1995]. We experimentally confirmed that instrumental variable algorithm [Viberg 1995] achieved better accuracy in most cases, which are defined as

$$\begin{aligned} \mathbf{W}_1 &= \left(\mathbf{Y}_{(b)}\mathbf{U}_{(b)}^\dagger\mathbf{Y}_{(b)}^T\right)^{-1/2}\ , \\ \mathbf{W}_2 &= \left(\begin{bmatrix} \mathbf{U}_{(0)} \\ \mathbf{Y}_{(0)} \end{bmatrix} \begin{bmatrix} \mathbf{U}_{(0)} \\ \mathbf{Y}_{(0)} \end{bmatrix}^T\right)^{-1/2}\ . \end{aligned} \tag{A3}$$

Note that, in the N2SID method, the matrix $\mathbf{Y}_{(b)}$ is constructed from the optimized variables $\mathbf{y}_\tau^*$ for providing a structure-preserving low-rank approximation of $\mathbf{\Theta}$, whereas the measured output $\mathbf{y}_\tau$ is used for computing $\mathbf{W}_1$, $\mathbf{W}_2$ and $\mathbf{Y}_{(0)}$.

The system matrices are estimated via TSVD $\mathbf{\Theta} \approx \mathbf{\Phi\Sigma\Omega}$, where $\mathbf{\Sigma} \in \Re^{k\times k}$ is composed of the k largest singular values of $\mathbf{\Theta}$. Given the singular values $\sigma_{\{i=1,\cdots,b\cdot\dim(\mathbf{y})\}}$ of $\mathbf{\Theta}$ in descending order, the matrix order k is determined by computing the cumulative contribution ratio $\lambda_k = \sum_{i=1}^k \sigma_i / \sum_{j=1}^{b\cdot\dim(\mathbf{y})} \sigma_j$ and finding the minimum k satisfying $\lambda_k \geq \lambda_{\min}$ for a given cutoff threshold $\lambda_{\min}$, where $\lambda_{\min} = 0.95$ was used for all of our experiments.

The system matrices $\mathbf{A}$ and $\mathbf{C}$ are then estimated using the product of the right singular vectors and the weight matrix $\mathbf{W}_1^{-1}\mathbf{\Phi}$, because it approximates $\left[\mathbf{C}^T,\ (\mathbf{CA})^T,\ (\mathbf{CA}^2)^T, \cdots,\ (\mathbf{CA}^{b-1})^T\right]^T$. Let $\mathbf{V} := \mathbf{W}_1^{-1}\mathbf{\Phi}$ and the partition $\mathbf{V}$ in b block rows $\mathbf{V}_1, \cdots, \mathbf{V}_b$ of size $\dim(\mathbf{y}) \times k$, then the optimal $\mathbf{C}^*$ is estimated to $\mathbf{C}^* = \mathbf{V}_1$, and $\mathbf{A}^*$ is obtained by solving the following least square problem,

$$\mathbf{A}^* = \underset{\mathbf{A}}{\operatorname{argmin}} \quad \sum_{i=2}^{b} |\mathbf{V}_i - \mathbf{V}_{i-1}\mathbf{A}|_{\mathcal{F}}^2\ , \tag{A4}$$

where $|\cdot|_{\mathcal{F}}$ denotes the Frobenius norm. The rest system matrices $\mathbf{B}$ and $\mathbf{D}$ and initial state vector $\mathbf{z}_0$ are estimated by solving the following least square problem:

$$\min_{\mathbf{B},\mathbf{D},\mathbf{z}_0} \sum_{i=1}^{N+b-1} \left| \mathbf{C}^*(\mathbf{A}^*)^i\mathbf{z}_0 + \sum_{j=0}^{i-1} \mathbf{C}^*(\mathbf{A}^*)^{i-j}\mathbf{B}\mathbf{u}_i + \mathbf{D}\mathbf{u}_i - \mathbf{y}_i \right|_2^2 .$$