DOI: 10.1111/cgf.13893

COMPUTER GRAPHICS *forum*
Volume 39 (2020), number 1 pp. 595–606

Synthesizing Character Animation with Smoothly Decomposed Motion Layers

Haegwang Eom[1,*], Byungkuk Choi[2,*], Kyungmin Cho[1], Sunjin Jung[1], Seokpyo Hong[1] and Junyong Noh[1]

[1]Visual Media Lab, KAIST, Daejeon, South Korea
{haegonggun, cluemaker, sunjin225, tjrvy8578, junyongnoh}@kaist.ac.kr
[2]Weta Digital, Wellington, New Zealand
litlpoet@gmail.com

Abstract

The processing of captured motion is an essential task for undertaking the synthesis of high-quality character animation. The motion decomposition techniques investigated in prior work extract meaningful motion primitives that help to facilitate this process. Carefully selected motion primitives can play a major role in various motion-synthesis tasks, such as interpolation, blending, warping, editing or the generation of new motions. Unfortunately, for a complex character motion, finding generic motion primitives by decomposition is an intractable problem due to the compound nature of the behaviours of such characters. Additionally, decomposed motion primitives tend to be too limited for the chosen model to cover a broad range of motion-synthesis tasks. To address these challenges, we propose a generative motion decomposition framework in which the decomposed motion primitives are applicable to a wide range of motion-synthesis tasks. Technically, the input motion is smoothly decomposed into three motion layers. These are base-level motion, a layer with controllable motion displacements and a layer with high-frequency residuals. The final motion can easily be synthesized simply by changing a single user parameter that is linked to the layer of controllable motion displacements or by imposing suitable temporal correspondences to the decomposition framework. Our experiments show that this decomposition provides a great deal of flexibility in several motion synthesis scenarios: denoising, style modulation, upsampling and time warping.

Keywords: animation systems, animation

ACM CCS: •Computing methodologies → Computer graphics, Animation, Motion processing

1. Introduction

The capturing and processing of human motion are very common processes at present with well-established pipeline techniques that generate realistic character animation. However, many of these techniques, such as the retargeting, warping, blending and editing of input motions, depend strongly on the original quality of the captured data. Thus, the initial motion must be suitably processed, before it can be used in the subsequent high-level operations so as to increase the quality of the final motion.

Because human motion can be viewed as a type of high-dimensional signal that varies over time, a variety of methods have been developed by generalizing conventional signal processing and geometric techniques for problems that arise from various motion-synthesis tasks. In particular, many researchers have employed motion decomposition techniques that extract meaningful motion primitives from the input motion. Then, based on the decomposed motion primitives, new motions are easily synthesized to remove high-frequency noise [XFJ*15], to compress dense data [LS01, ZSD12], to change the style of motions [UAT95, RCB98, KPS03, SCF06] or to recognize and retrieve a motion [Vas02, ZSD12].

Unfortunately, finding generic motion primitives by decomposition is inherently difficult due to the complex nature of human motion. Hence, existing decomposition techniques have mainly attempted to search for meaningful primitives that are especially designed for a specific motion synthesis scenario, which in turn has complicated the generalization of these processes to different

*Both the authors contributed equally to this work.

scenarios. For instance, it is often difficult to apply a technique that is mainly designed to remove high-frequency noise to a different context in which the primitives need to be amplified or time-warped in a systematic manner.

As an alternative to motion primitives, the direct processing or modulation of raw motion signals has been widely adapted to achieve a general solution for various types of motions. Due to its simplicity and computational advantages, a parametric model is used most often for the decomposition of a single motion [RCB98, LS01, BSH15]. Though the parametric model is well suited for use with spline-based interpolation techniques, it is also associated with limited expressiveness that leads to re-parameterizations, corrections or sophisticated manual interventions for further extensions.

To address these issues, we propose a novel generative motion decomposition technique for which the input motion is separated into three motion layers corresponding to base-level motions, a layer with controllable motion displacements and a layer with high-frequency residuals. The key innovation of our approach is its flexible generation of smoothly decomposed motion layers, which makes it suitable for a range of useful motion applications to synthesize new character motions. As we opt for a non-parametric generative method to extract each motion layer, our decomposition technique is more robust against various motion signals compared to existing parametric or spline-based methods [LS99, LS01]. In addition, the resulting synthesized motions from the combinations of layers are always smooth, which is vital for the subsequent high-level motion processing.

Technically, the decomposition of smooth motion layers of articulated characters entails careful considerations of certain motion properties, such as the proper choice of orientation / rotation representation and preservation of kinematic constraints, such as foot contacts and joint limits. For the base-level motion, we propose to extend linear Gaussian systems (LGSs) directly to joint orientations - unit quaternions – with a strong smoothness prior. For the layer of controllable motion displacements (rotations), the same technique is utilized but with a weak smoothness prior. To preserve contextual information of the given motion and to finalize the motion synthesis process, a spacetime optimization method imposed with automatically detected environmental constraints is used.

This paper is organized as follows. First, related studies and applications are introduced in Section 2. The application of an LGS to compute reliable motion signals from given motion data is described in Section 3. Next, explanations of how the LGS can be extended to decompose a given motion into three motion layers by carefully considering the motion properties of a human are discussed in Section 4. Section 5 demonstrates the impact of smoothly decomposed motion layers in several motion synthesis scenarios: denoising, style modulation, upsampling and temporal warping of the given motion. Finally, we conclude with a brief discussion of the limitations and future directions in Sections 6 and 7.

2. Related Work

Motion signal processing Early studies in the area of motion signal processing mainly focused on generalizing conventional signal processing and geometric techniques to manipulate motion data. Some of the pioneering studies, fundamental theories of motion signal processing by Bruderlin and Williams [BW95] and the time warping of motion by Witkin and Popović [WP95], have been widely used for decades. As the processing of orientation and rotation signals is known to be especially challenging, a significant body of research has been compiled thus far in an effort to obtain stable orientation and rotation signals during the processing steps [Sho85, KKS95, RB97, Gra98, FHK*98, JS02, Lee08, Bou13, LDH14]. In these studies, depending on the particular purpose of the application, a single representation, either a unit quaternion or an exponential map, was considered for simplicity. Our work employed unit quaternions for base-level motions to achieve stable decomposition of the orientations and took advantage of rotation vectors with an exponential map for a layer of controllable motion displacements, which in turn provided a reasonable level of controllability.

Motion denoising As part of an animation pipeline, motion denoising is often required for cleaning up noisy input motion data. Many research outcomes have been reported in this area over the last two decades. Early studies of motion denoising mainly focused on geometric processing, such as translational or rotational smoothing [LS96, LS01, LS02, FHK*98]. Data-driven methods have been proposed to preserve the spatio-temporal features of highly coordinated human motions [HJ10, FJX*14, XFJ*15]. Recently, deep learning frameworks have shown a high-quality performance in the correction of motion data [HSKJ15, HSK16, Hol18, MLCC17, LZZ*19]. While the direction that the deep learning framework pursues is promising, its applicability in practice can be limited as it requires training priors, such as noisy or corrupted motion data; the resulting system is inevitably specialized to handle the situations defined by the given data. In contrast, our approach effectively denoises the motion of any character, such as a human or a quadruped by decomposing it into layers without any preprocessing or dependency on pre-selected situations.

Motion decomposition and synthesis To accomplish various synthesizing demands, many researchers investigated a rational choice of motion primitives that were often decomposed or extracted from input motions. The motion primitives were then utilized as bases to generate or synthesize new motions. Selecting sparse key postures from a given dense motion has proven to be feasible for use with spline-based interpolation techniques [LT01, BSH15]. The extraction of sparse motion bases in the quaternion space [ZSD12] or the extraction of high-level motion bases from large motion databases [XFJ*15] has also been explored. Decomposing a motion into several important components, especially to synthesize a new motion, has been another direction pursued since the work of Rose *et al.* [RCB98]. To incorporate style variations on top of a base motion, Unuma *et al.* [UAT95] split a motion into several frequency bands by Fourier expansions, and Shapiro *et al.* [SCF04, SCF06] decomposed a motion through an independent component analysis. Among all related decomposition techniques, Lee's work [LS99, LS01], in which decomposition was achieved by a multiresolution analysis, is most relevant to ours. Instead of using a spline-based representation, we utilized a generative statistical model that provided stable estimates of orientation / rotation values as well as considerable flexibility with various motion synthesis tasks using decomposed motion layers.

Recently, Holden *et al.* [HSK16] introduced a deep learning-based approach that made it possible to synthesize and edit character motion. Their approach works well for tasks such as following root paths for the purpose of locomotion synthesis. In terms of applicability to various situations, however, our approach has noteworthy advantages. For example, while Holden *et al.* [HSK16] require training for each specific motion with corresponding data, such as dancing and kicking, our method is not learning based and therefore our system naturally allows input motion with any joint-structure or motion type. Furthermore, in their method, the network has to be redesigned for different tasks (style transfer vs. style modulation). In contrast, our method can handle various synthesis scenarios, such as style modulation, resampling, time warping and so on, in a unified framework.

Generative models for motion synthesis There has been strong renewed interest in generative models for motion synthesis recently due to the increasing power of computations and advances in solving relatively large systems rapidly. The Gaussian process latent variable model has been used to model style-based inverse kinematics [GMHP04], to model smooth dynamics in human motion [WFH08] and to control character motions [LWH*12]. To achieve style variations from given motion databases, researchers employed a Bayesian network in several studies [LBJK09, MXH*10]. Finally, Gaussian mixture models and Gaussian processes (GPs) have also been adopted to compute smooth motion paths [TL08], to edit and control a given motion [IAF09, UK12] and to synthesize style variations [ZSSL14]. In this study, we extended the LGS in a motion decomposition framework, with a focus on estimating smoothly decomposed motion layers in a non-parametric manner.

3. Motion Smoothing Using Linear Gaussian Systems

The key idea of our motion decomposition framework is to generate, rather than interpolate, smooth functions from a given motion using an LGS. An LGS is a non-parametric generative regression technique based on a Bayesian assumption pertaining to the underlying functions. Thus, when relying on model priors with regard to the uncertainty, an LGS has the benefits of naturally avoiding the over- and under-fitting problems that commonly arise when parametric regression methods are used. The LGS has been widely adopted in various domains, and we briefly explain the main concepts while focusing on character motion. For more detailed explanations, the reader can refer to Calvetti and Somersalo [CS07] and Murphy [Mur12].

Given a character motion, we assume that each joint channel contains N_d noisy observations $\mathbf{y} = [y_1^{\mathrm{T}}\ y_2^{\mathrm{T}} \dots\ y_i^{\mathrm{T}}\ \dots\ y_{N_d}^{\mathrm{T}}]^{\mathrm{T}}$ that correspond to N_f hidden variables $\mathbf{x} = [x_1^{\mathrm{T}}\ x_2^{\mathrm{T}}\ \dots\ x_j^{\mathrm{T}}\ \dots\ x_{N_f}^{\mathrm{T}}]^{\mathrm{T}}$. Without loss of generality, notation $\mathbf{x}$ and $\mathbf{y}$ can represent any entity describing each degree of freedom of joints. For example, translation can be represented as $\mathbf{x}, \mathbf{y} \in \mathbb{R}^3$, while a hinge joint can be thought of as $\mathbf{x}, \mathbf{y} \in \mathbb{R}$. We can then model the likelihood $p(\mathbf{y}|\mathbf{x})$ using the LGS as follows:

$$\mathbf{y} = \mathrm{A}\mathbf{x} + \boldsymbol{\epsilon}_{\mathbf{y}}, \tag{1}$$

where $\boldsymbol{\epsilon}_{\mathbf{y}} \sim \mathcal{N}(0, \sigma^2\mathrm{I})$, σ^2 is the observation noise and A is a $N_d \times N_f$ projection matrix that selects the observed data from the entire frame range N_f. For example, given three data points ($N_d = 3$) corresponding to the first, second and fifth frames to estimate x_j for five frames ($N_f = 5$), the following projection matrix can be constructed:

$$\mathrm{A} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2}$$

Note that the number of data points N_d is fixed, as it is identical to the number of sampled postures of a given motion in our setting. However, N_f is the number of target frames to estimate (hidden in terms of the LGS), which can be used to control temporal variations together with the projection matrix A. We describe how to design A specifically in Section 5.3.

A smoothness prior can also be encoded by assuming that x_j is the average of its neighbours, x_{j-1} and x_{j+1}, plus some Gaussian noise:

$$x_j = \frac{1}{2}(x_{j-1} + x_{j+1}) + \epsilon_j, (2 \le j \le N_j - 1), \tag{3}$$

where $\boldsymbol{\epsilon}_j \sim \mathcal{N}(0, \frac{1}{\lambda}\mathrm{I})$. In vector form, this equation can be written as:

$$\mathrm{L}\mathbf{x} = \boldsymbol{\epsilon}, \tag{4}$$

where L is the $(N_f - 2) \times N_f$ second-order finite difference matrix. The corresponding prior has the following form:

$$p(\mathbf{x}) = \left(\mathbf{x}|0, (\lambda^2\mathrm{L}^{\mathrm{T}}\mathrm{L})^{-1}\right). \tag{5}$$

At this stage, the posterior mean can be computed using a Bayes rule for LGS as follows:

$$\begin{aligned} p(\mathbf{x}|\mathbf{y}, \lambda, \sigma) &= \mathcal{N}(\mathbf{x}|\boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}}, \Sigma_{\mathbf{x}|\mathbf{y}}), \\ \Sigma_{\mathbf{x}|\mathbf{y}}^{-1} &= \Sigma_{\mathbf{x}}^{-1} + \mathrm{A}^{\mathrm{T}}\Sigma_{\mathbf{y}}^{-1}\mathrm{A} = \lambda^2\mathrm{L}^{\mathrm{T}}\mathrm{L} + \frac{1}{\sigma^2}\mathrm{A}^{\mathrm{T}}\mathrm{A}, \\ \boldsymbol{\mu}_{\mathbf{x}|\mathbf{y}} &= \Sigma_{\mathbf{x}|\mathbf{y}}\left[\mathrm{A}^{\mathrm{T}}\Sigma_{\mathbf{y}}^{-1}\mathbf{y} + \Sigma_{\mathbf{x}}^{-1}\boldsymbol{\mu}_{\mathbf{x}}\right] \\ &= \frac{1}{\sigma^2}\left(\lambda^2\mathrm{L}^{\mathrm{T}}\mathrm{L} + \frac{1}{\sigma^2}\mathrm{A}^{\mathrm{T}}\mathrm{A}\right)^{-1}\mathrm{A}^{\mathrm{T}}\mathbf{y}. \end{aligned} \tag{6}$$

Here, the posterior mean μ represents the resulting estimate for a given signal. Note that the prior precision λ and the variance of the observation noise σ can be used as smoothness parameters that affect the posterior mean μ. In particular, for a strong prior (large λ), the estimate is very smooth, while for a weak prior (small λ), the estimate is close to the given motion. Although the observation noise σ also affects the posterior mean, the prior precision λ will contribute more to the overall smoothness, which is very useful because it effectively extracts a base-level (coarse) signal from the given data. The posterior mean can be computed equivalently by solving the following optimization problem:

$$\begin{aligned} \min_{\mathbf{x}}\ & \frac{1}{2\sigma^2}\sum_{i=1}^{N_d}(x_{i\mapsto j} - y_i)^2 \\ & + \frac{\lambda}{2}\sum_{j=2}^{N_f-1}\left\{(x_j - x_{j-1})^2 + (x_j - x_{j+1})^2\right\}, \end{aligned} \tag{7}$$

Figure 1: *The y values of the translation channel of the root joint in the original motion (blue) and their corresponding y values from three different types of synthesis: base-level extraction (red), denoising (green) and stylize modulation (yellow).*

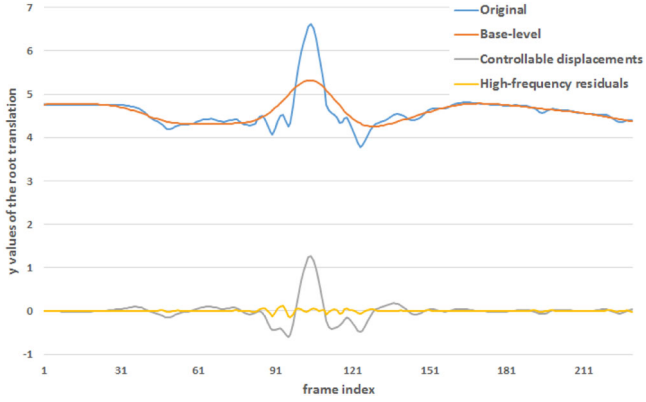


Figure 2: *One of the translational channels of the original motion (blue) is decomposed into three layers: the base-level movement (red), a layer with controllable displacements (grey) and a layer with high-frequency residuals (yellow). Note that the same geometric interpretation is possible for orientations and rotations as for translations. However, it is difficult to effectively visualize their values altogether, as the rotation vectors in two layers (controllable and residual) should be converted to quaternions to have an identical geometric meaning.*

where $x_{i\mapsto j}$ represents the mapping from the index i of the data to the corresponding frame index j, which is analogous to the role of the design matrix A in Equation (2). This can be recognized as a discrete approximation of the Tikhonov regularization, where the first term fits the data and the second term penalizes estimated values that vary too widely. These two Equations (6) and (7) are used interchangeably throughout this paper.

4. Multi-level Motion Decomposition

In this section, we explain how LGS can be efficiently utilized in the context of a character motion. We treat a motion as a discretized high-dimensional vector varying over time. Let $\mathcal{M}(t) = (\mathbf{p}_0(t), \mathbf{q}_0(t), \ldots, \mathbf{q}_{N_j-1}(t))$, $(1 \le t \le N_f)$ denote a given motion, where $\mathbf{p}_0(t) \in \mathbb{R}^3$ and $\mathbf{q}_i(t) \in \mathbb{S}^3$, $(0 \le i < N_j)$ describe the translational motion of the root segment and the rotational motion of the ith joint at time t, respectively. N_j is the number of joints.

The key idea of our method is to decompose a given motion $\mathcal{M}(t)$ that is noisy and difficult to manipulate within a high-dimensional vector space of a single layer. The decomposition proceeds as follows. First, we define base-level motion $\mathcal{M}^b(t)$, as the invariant primitive of input motion $\mathcal{M}(t)$, at time t. The base-level motino is obtained by applying the LGS as described in Section 3. We then apply the LGS once again to the residual between $\mathcal{M}(t)$ and $\mathcal{M}^b(t)$ in order to separate it into a controllable motion displacement vector $\boldsymbol{\delta}^c(t)$ and the residual motion displacement $\boldsymbol{\delta}^r(t)$ that consists of high-frequency noise (Figures 1 and 2). To this end, the given motion $\mathcal{M}(t)$ is decomposed into three levels as follows:

$$\mathcal{M}(t) = (\mathcal{M}^b(t) \oplus \boldsymbol{\delta}^c(t)) \oplus \boldsymbol{\delta}^r(t). \tag{8}$$

Here, the operator $\oplus$ represents a displacement mapping; simple vector addition for the translational part and quaternion multiplication for the rotational part after exponentiating the 3D rotation vector [KKS95, LS99, Lee08]. We chose quaternion multiplication for stable composition of a wide range of rotations. Other representations, such as an exponential map, produced interpolation artefacts on extreme poses due to their singularity. The comparison result between the choices is shown in Section 5.2.

In what follows, we explain in detail how to apply the LGS described in Section 3 to extract the base-level motion $\mathcal{M}^b(t)$ in quaternion space and a controllable motion displacement vector in Sections 4.1 and 4.2, respectively. Section 4.3 explains a contact handling method as we applied it to the decomposed motion layers.

4.1. Base-level motion

To extract a useful base-level motion, we smooth out a given motion by optimizing the LGS using a strong smoothness prior together with the assumption of a significant observation noise in the input motion. The strong smoothness prior can be enforced by assigning relatively large values to λ and σ in Equation (6). The choices of λ and σ depend on the type of given motions; however, we can find practical values for them using a simple heuristic that is detailed in Section 4.2.

For a translational joint, the base-level trajectory can be directly extracted by computing the posterior mean $p(\mathrm{X}_t|\mathrm{Y}_t, \lambda^b, \sigma^b)$ using Equation (6). Here, X_t and Y_t are corresponding $N_f \times 3$ and $N_d \times 3$ matrices created by stacking every translational vector along the rows.

Because orientations (SO3) and their associated operations are generally well defined in quaternion space, the direct approximation of smooth orientations using Equation (6) may be ill-posed. As Equation (6) is specialized for entities in Euclidean space, we are required to solve a nonlinear optimization problem over quaternion

space for the stable estimation of joint orientations by extending Equation (7) as follows:

$$\min_{\mathbf{q}^b_{\mathbf{x}}} \frac{1}{2\sigma^2}\sum_{t=1}^{N_d}\left\|\log\left(\mathbf{q}^b_{\mathbf{x}}(t)^{\mathrm{T}}\mathbf{q}_{\mathbf{y}}(t)\right)\right\|^2 + \frac{\lambda}{2}\sum_{t=2}^{N_f-1}\left\{\left\|\log\left(\mathbf{q}^b_{\mathbf{x}}(t-1)^{\mathrm{T}}\mathbf{q}^b_{\mathbf{x}}(t)\right)\right\|^2 + \left\|\log\left(\mathbf{q}^b_{\mathbf{x}}(t+1)^{\mathrm{T}}\mathbf{q}^b_{\mathbf{x}}(t)\right)\right\|^2\right\}. \tag{9}$$

Here, $\log(\mathbf{q})$ is the quaternion log map that converts a quaternion into a rotation vector. As $\log(\mathbf{q}_a^{\mathrm{T}}\mathbf{q}_b)$ represents a rotational difference between two orientations, each term is well defined. However, it is difficult to differentiate the quaternion log map. Moreover, optimizing the nonlinear equation can be costly for a large number of frames. With the observation that, for two close orientations $\mathbf{q}_a, \mathbf{q}_b \in \mathbb{S}^3$, the geodesic distance and the chordal distance are approximately equal [Bou13, LDH14],

$$g(\mathbf{q}_a, \mathbf{q}_b) = \left\|\log\left(\mathbf{q}_a^{\mathrm{T}}\mathbf{q}_b\right)\right\| \approx \|\mathbf{q}_b - \mathbf{q}_a\|, \tag{10}$$

where $g(\mathbf{q}_a, \mathbf{q}_b)$ computes the geodesic distance between two different orientations, and where the cost function (Equation 9) can be simplified for joint orientations without a quaternion log map as follows:

$$\min_{\mathbf{q}^b_{\mathbf{x}}} \frac{1}{2\sigma^2}\sum_{t=1}^{N_d}\left\|\mathbf{q}^b_{\mathbf{x}}(t) - \mathbf{q}_{\mathbf{y}}(t)\right\|^2 + \frac{\lambda}{2}\sum_{t=2}^{N_f-1}\left\{\left\|\mathbf{q}^b_{\mathbf{x}}(t) - \mathbf{q}^b_{\mathbf{x}}(t-1)\right\|^2 + \left\|\mathbf{q}^b_{\mathbf{x}}(t) - \mathbf{q}_{\mathbf{x}}{}^b(t+1)\right\|^2\right\}. \tag{11}$$

This equation is well suited for the setting of the LGS; therefore, solving this equation is equivalent to computing the posterior mean of $p(\mathrm{X}_{\mathbf{q}}|\mathrm{Y}_{\mathbf{q}}, \lambda^b, \sigma^b)$ using Equation (6), where $\mathrm{X}_{\mathbf{q}}$ and $\mathrm{Y}_{\mathbf{q}}$ are correspondingly $N_f \times 4$ and $N_d \times 4$ matrices created by stacking every quaternion $\mathbf{q}$ as a four-dimensional homogeneous vector along the rows. Note that the resulting posterior mean of the orientations must be renormalized to ensure that the quaternions retain the unit length.

As mentioned in Section 3, the prior precision λ and the variance of the observation noise σ in Equation (11) are essential parameters concerning the smoothness of a base-level motion. If the base-level motion is extracted with a weak prior (small λ and σ), a layer of controllable motion displacements eventually has a small difference from the base-level motion that cannot provide sufficient controllability. On the other hand, rotation vectors in the layer of controllable motion displacements may be ill-defined if the base-level motion is extracted with a strong prior (large λ and σ). We explain in the subsequent section how we find proper parameter values for both λ and σ.

4.2. Controllable motion displacements

After the base-level motion is obtained, we can factor out motion displacements $\boldsymbol{\delta}(t)$ between the given motion and the base-level motion as follows:

$$\boldsymbol{\delta}(t) = \mathcal{M}(t) \ominus \mathcal{M}^b(t). \tag{12}$$

Specifically,

$$\boldsymbol{\delta}(t) = \begin{pmatrix}\mathbf{p}_0(t)\\ \mathbf{q}_0(t)\\ \vdots\\ \mathbf{q}_{N_j-1}(t)\end{pmatrix} \ominus \begin{pmatrix}\mathbf{p}^b_0(t)\\ \mathbf{q}^b_0(t)\\ \vdots\\ \mathbf{q}^b_{N_j-1}(t)\end{pmatrix} = \begin{pmatrix}\mathbf{p}_0(t) - \mathbf{p}^b_0(t)\\ \log(\mathbf{q}^b_0(t)^{\mathrm{T}}\mathbf{q}_0(t))\\ \vdots\\ \log(\mathbf{q}^b_{N_j-1}(t)^{\mathrm{T}}\mathbf{q}_{N_j-1}(t))\end{pmatrix} = \begin{pmatrix}\mathbf{u}_0(t)\\ \mathbf{v}_0(t)\\ \vdots\\ \mathbf{v}_{N_j-1}(t)\end{pmatrix}, \tag{13}$$

where $\mathbf{u}_0(t) \in \mathbb{R}^3$ denotes a translation vector, and $\mathbf{v}_i(t) = \theta\hat{\mathbf{v}}_i(t) \in \mathbb{R}^3$ is a rotation vector that maps $\mathbf{v}_i(t)$ to a unit quaternion representing a rotation of $\theta = \|\mathbf{v}_i(t)\|$ about the axis $\hat{\mathbf{v}}_i(t) = \mathbf{v}_i(t)/\|\mathbf{v}_i(t)\|$ by the quaternion exponentiation $\exp(\mathbf{v}_i(t))$ [Lee08]. Note that the operator $\ominus$ is equivalent to the inverse of the displacement map described in Equation (8).

Practical base-level motion In general, quaternion exponentiation is a many-to-one mapping. Consequently, the domain is usually limited to $\|\mathbf{v}_i(t)\| < \pi$ in order to define an inverse log map in Equation (13). Therefore, in our decomposition setting for the base-level motion, we need to design $\mathbf{q}^b_i(t)$ to cover a sufficient range $||\log(\mathbf{q}^b_i(t))|| \lesssim \pi$ of a given orientation by smoothing the original orientation $\mathbf{q}_i(t)$. As a result, the exponential map of $\mathbf{v}_i(t)$ guarantees a one-to-one mapping and its inverse log map becomes well defined within the limited domain $\|\mathbf{v}_i(t)\| < \pi$.

We employ a simple heuristic to find the best parameter values for the smoothness prior explained in Section 4.1 that will lead to a reasonably good and practical base-level motion. Starting from weak prior values ($\lambda = 1.0$ and $\sigma = 0.1$), we iteratively apply Equation (11) to the given motion while increasing both λ and σ until the inverse log map of motion displacements (Equation 13) becomes higher than a certain threshold. Specifically, we stop the iteration if

$$\max \|\mathbf{v}_i(t)\| > \gamma. \tag{14}$$

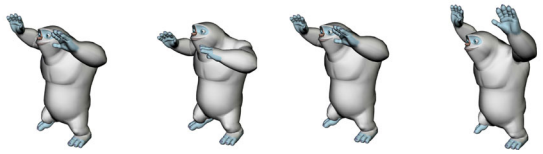
Because of varying degrees of movement of a given motion, some motions require greater maximum displacements than others. Instead of manually setting λ and σ for each case, the user can easily extract a base-level motion from various motions according to an intuitive criterion γ ($\pi/2 \le \gamma < \pi$), which determines the range of control.

Residual motion displacements Because the base-level motion $\mathcal{M}^b(t)$ is extracted as smoothly as possible while reasonably approximating the given motion, the decomposed displacements $\boldsymbol{\delta}(t)$ may contain high-frequency noise from the original motion. In order to remove this noise, we apply the LGS once again and separate high-frequency residuals from the displacements as follows:

$$\begin{aligned}\boldsymbol{\delta}^c(t) &= \boldsymbol{\mu}_{\mathrm{X}_\delta|\mathrm{Y}_\delta,\lambda^c,\sigma^c}(t),\\ \boldsymbol{\delta}^r(t) &= \widetilde{\exp}(\boldsymbol{\delta}(t)) \ominus \widetilde{\exp}(\boldsymbol{\delta}^c(t)),\end{aligned} \tag{15}$$

where $\widetilde{\exp}(\boldsymbol{\delta}) = (\mathbf{u}_0(t), \exp(\mathbf{v}_0(t)), \ldots, \exp(\mathbf{v}_{N_j-1}(t)))$ is an exponential map generalized to the displacement vectors, and where the smoothed displacement vector $\boldsymbol{\delta}^c(t)$ is the posterior mean of

(a) The *Malcolm* model.



(b) The *Gump* model.

Figure 3: *(From left to right) The original, a base-level, a denoised ($\alpha = 1.0$) and a style amplified motion ($\alpha = 3.0$ for the* Malcolm *model and $\alpha = 2.0$ for the* Gump *model).*

$p(\mathrm{X}_\delta|\mathrm{Y}_\delta, \lambda^c, \sigma^c)$ at time t. Note that $\boldsymbol{\delta}^r(t)$ contains only the noisy residuals from the original motion, which it is always preferable to remove.

Controllability From a geometric point of view, $\boldsymbol{\delta}^c(t)$ can be interpreted as the translation $\mathbf{u}_0(t)$ of the root and rotations $\{\mathbf{v}_i(t)|0 \le i < N_j\}$ of every joint with respect to the base-level posture $\mathcal{M}^b(t) = (\mathbf{p}_0^b(t), \mathbf{q}_0^b(t), \ldots, \mathbf{q}_{N_j-1}^b(t))$. Furthermore, as the scalar multiplication of the rotation vector $\alpha\mathbf{v} \in \mathbb{R}^3$ is well defined, which represents a spinning motion around the same axis $\hat{\mathbf{v}}$ but where the magnitude of its rotation angle $\|\mathbf{v}\|$ is scaled by a factor of α [Gra98, Lee08], we utilize $\boldsymbol{\delta}^c(t)$ to control the resulting synthesized motion as follows:

$$\mathcal{M}'(t) = \mathcal{M}^b(t) \oplus \mathrm{W}(t)\boldsymbol{\delta}^c(t). \tag{16}$$

Here, $\mathcal{M}'(t)$ denotes the synthesized new motion and $\mathrm{W}(t)$ represents a block diagonal matrix that controls the scale of a displacement vector for each joint at time t, as follows:

$$\mathrm{W} = \begin{bmatrix} \alpha_0^{\mathbf{u}}\mathrm{I}_3 & 0 & \cdots & 0 \\ 0 & \alpha_0^{\mathbf{v}}\mathrm{I}_3 & & \vdots \\ \vdots & & \ddots & 0 \\ 0 & \cdots & 0 & \alpha_{N_j-1}^{\mathbf{v}}\mathrm{I}_3 \end{bmatrix}. \tag{17}$$

In this equation, $\alpha_0^{\mathbf{u}}\mathrm{I}_3$ and $\alpha_i^{\mathbf{v}}\mathrm{I}_3$ are weight factors for the root translation and the rotation of the ith joint, respectively. Explanation of how to design the control matrix $\mathrm{W}(t)$ for denoising and style modulation is provided in Section 5.2.

4.3. Contact handling

Because our decomposition technique utilizes joint space signals, the resulting joint trajectories in the operation space, i.e. a joint path in the world space, may deviate from the original trajectories after synthesizing a new motion. For closed-chain joints, such as a leg that has rich foot contacts, this leads to undesirable artefacts, such as foot sliding. To avoid these artefacts, every environmental contact in the original motion is initially pre-computed using the method by Le Callennec and Boulic [LCB06], and this is followed by solving a spacetime optimization to synthesize a motion with a set of contact constraints. The final motion accounting for environmental contacts can be computed as follows:

$$\mathcal{M}^*(t) = \mathcal{M}'(t) \oplus \boldsymbol{\delta}^e(t) = \left(\mathcal{M}^b(t) \oplus \mathrm{W}(t)\boldsymbol{\delta}^c(t)\right) \oplus \boldsymbol{\delta}^e(t), \tag{18}$$

where $\mathcal{M}'(t)$ is a synthesized motion by utilizing Equation (16) without considering contacts, and $\boldsymbol{\delta}^e(t)$ is a displacement vector at time t computed by the spacetime optimization given the original contact points as positional constraints [LS99].

5. Results

We now demonstrate how the proposed decomposition technique can be easily applied to various motion-synthesis tasks. First, we detail the experimental settings used to produce all of the results in this paper. Next, we present the techniques for handling spatial variations for noise removal and style modulation. Finally, we provide explanations of how the LGS can be used to modulate temporal variations for upsampling and dynamic time warping.

5.1. Experimental settings

For all of the experimental results, the motion data were captured via a Vicon optical system with eight cameras at a rate of 120 fps and then sub-sampled to 24 fps. We demonstrated our method with three different characters. First, the *Malcolm* model had a skeleton structure with 65 bones and 57 DoFs, while the *Gump* model had a skeletal structure identical to that of the *Malcolm* model but with different proportions (Figure 3). Finally, the *Dog* model had a quadruped skeleton structure with 49 bones and 151 DoFs. All of the redundant DoFs in the joints were excluded from the computation. Please refer to the Supplementary Video for these models in action.

All experiments were conducted on a desktop computer using a single thread with an Intel Core i7-5930k@3.50GHz and 32GB RAM. We implemented our framework as a stand-alone application and exposed a few user parameters (α and N_f) with tunable sliders; their values as used in each experiment are explained in the following sections. We also set the parameters related to the decomposition process: $\gamma = \pi/2$ for Equation (14), $\lambda^c = 1$ and $\sigma^c = 1$ for Equation (15).

Since our system uses an LGS, which is essentially a GP, the computation time grows exponentially when the number of data points increases [IAF09]. As a remedy, we adapted a window-based optimization technique by repeatedly optimizing the window of 120 frames at intervals of 60 frames. To prevent discontinuity between the two windows, we interpolated overlapping frames with weights computed from a sigmoid function. With this approach, our system achieves a scalable computation time regardless of the size of the input data. A breakdown of the computational costs for the primary operations is summarized in Table 1. Note that our window-based optimization is simply applied to all of our synthesizing scenarios. In contrast, it is difficult to apply the same principle to counterpart methods, such as a dynamic time warping

Table 1: *A breakdown of the computational costs required for various types of motions.*

	Motion	N_d	N_f	Total(s)	
				entire frame	window-based optimization
Denoising	Body building	2526	2526	77 839.9	5.1
	Sword	330	330	9.5	2.0
	Dog	200	200	1.3	0.9
Style modulation	Kick	230	230	3.2	0.6
	Parkour	200	200	1.0	0.3
	Dance	560	560	77.9	2.7
	Throwing	360	360	17.7	1.7
Re-sampling	Airplane	8	40	0.001	0.001
	Ballet	34	680	6.0	6.0
Automatic time-warping	Kick	230	286	3.9	–
	Diving catch	211	261	3.0	–
	Jump	79	139	0.09	–

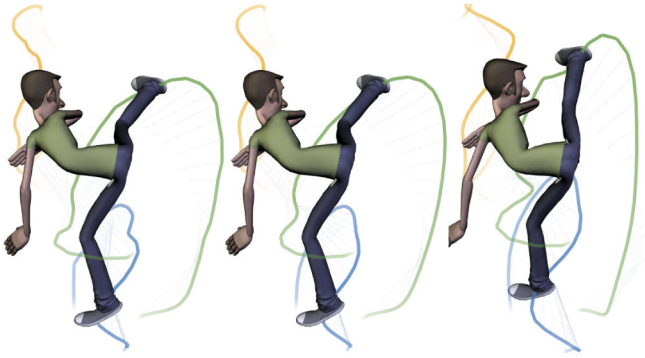


Figure 4: *(From left to right) Trajectories of the original, a denoised and a style amplified motion.*

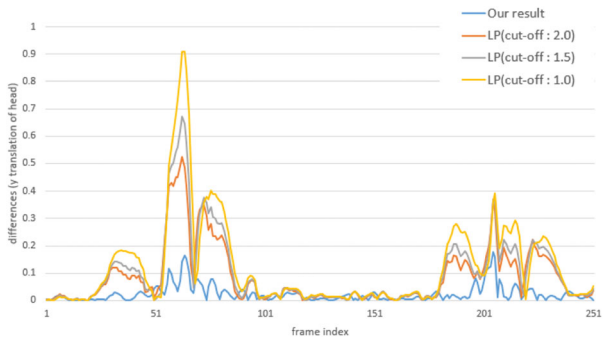


Figure 5: *The differences in denoising results from the ground truth data (measured in the Euclidean distance). Our results (blue) generally have a smaller positional error than those from applying a low-pass filter with various cut-off values.*

because the correspondence of the frames between the observed data and the desired result is not linear.

5.2. Denoising and style modulation

To synthesize spatial variations, with smoothly decomposed motion layers, we directly leverage the controllable motion displacements by adjusting the α value in Equation (17).

Denoising Removing noise from a captured motion is an indispensable pre-processing task that must be performed to achieve a high-quality motion. As described in Section 4.2, our method removes high-frequency noise $\boldsymbol{\delta}^r(t)$ automatically during the decomposition process. Specifically, denoised motion can be obtained by simply setting $\alpha = 1$, $\mathrm{W}(t) = \mathrm{I}$. Note that our method effectively removes every instance of high-frequency noise in the joint space such that the signals in the operation space of the joints are naturally denoised (Figure 4). Smooth signals in the operation space are particularly important when spacetime constraints need to be imposed for further processing of the resulting motion (Section 4.3).

We compare our denoising results to those produced by applying a low-pass filter with various cut-off parameters. We first generate synthetic noisy input data by adding random translation and rotation values to each pose along the time. Then, we denoise the data to reconstruct the ground truth motion. Figure 5 shows the positional errors of the denoised results from the ground truth data. Our method reproduces the ground truth data reasonably well with a small error over most of the time, while the application of a simple low-pass filter with various cut-off parameters fails to smooth the motion in certain ranges. See the Supplementary Video for a visual comparison.

We also compare the results produced using the quaternion representation and an exponential map. Figure 6 shows that rotation multiplication becomes unstable near the singularity point($\pi/2$) when the exponential map representation is employed, as explained in Section 4.

Style modulation By changing α values continuously instead of setting $\alpha = 1$, controllable motion displacements can be modulated with respect to the base-level motion. Because base-level motions can be considered as motions that reveal only global behaviour, we can produce style variations ranging from flat ($\alpha = 0$) to dynamic ($\alpha > 1$) compared to the original motion, which is similar

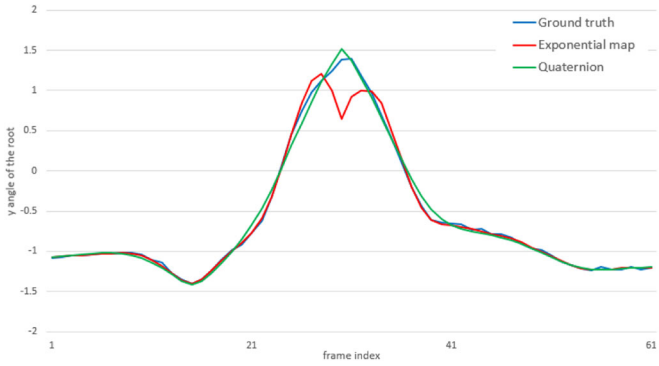


Figure 6: *Denoising operation using different rotation representations. While the quaternion (green) produces stable rotation values close to the ground truth data (blue), the exponential map (red) suffers at the singularity point around* $\pi/2$.

Figure 7: *The original motion (left) is exaggerated by the MultiRes method (middle) and our method (right). Our method modulates the original motion more consistently than the MultiRes method when the original motion has substantial rotational changes.*

to the work of Wang *et al.* [WDAC06]. A more useful extension can be achieved by designing a control matrix W(t) that affects a set of joints separately, which is similar to the work of Zhou *et al.* [ZSSL14] and Xiao *et al.* [XFJ*15]. Specifically, instead of using a uniform α for all joints in Equation (17), our method modulates local styles by controlling only the partial set of α_i^v that corresponds to the joints of a selected body part.

Comparisons to spline-based techniques We compare our method with a multi-resolution motion analysis technique (MultiRes) [LS01] that has a concept similar to ours in generating a final motion in the context of style modulation. The MultiRes method approximates a given or an edited motion with multiple displacement layers in a coarse-to-fine manner. To perform a same style modulation task, we first exaggerate or attenuate a given motion using our method and then mimic the same style using the MultiRes method. As the MultiRes requires editing of given postures to embrace style variations, we just use some of the key postures chosen from the final result of our method. Because it is difficult to measure the difference between the two methods quantitatively, we instead evaluated a visual quality and its interpolation performance in the synthesized motion. Figure 7 shows that both methods generate a similar motion style exaggerated from the original kicking motion. However, our method produces more visually consistent interpolation compared to the MultiRes method for a highly dynamic motion. Though a spline-based motion processing technique, such as the MultiRes method, can modulate style variations in a wide range by properly imposing spatial constraints, for simple style modulation tasks, our method produces smooth and consistent motion variations without imposing any manual constraints. Please see the Supplementary Video for more examples.

5.3. Resampling and time warping

To synthesize temporal variations prior to the decomposition process, we change the number of target frames N_f and map corresponding frame indices from y_i to x_j properly in Equation (7). Specifically, these changes are encoded into the projection matrix of the LGS, which in turn, generates retimed motions from the original motion.

Resampling By simply mapping the corresponding frame indices from y_i to x_j uniformly, globally resampled motion can be easily achieved. Specifically, our method estimates the orientations more stably than does a spline-based technique, such as a quaternion-based spherical linear interpolation (*SLERP*) (Figure 8). In an upsampling case, while the quaternion *SLERP* operation is apt to choose an incorrect path such that the in-between rotations produce flipping artefacts, our method always explicitly generates stable in-between rotations by considering neighbouring rotations with the help of the smoothness prior.

Time warping Instead of using uniform correspondences between y_i and x_j, retimed motions are synthesized by dynamically varying the correspondences over time, which is similar to the work of Witkin and Popovic [WP95] and Hsu *et al.* [HdSP07]. For instance, highlighting a portion of a dramatic action sequence by gradually easing the motion in and out can simply be achieved by automatically changing the correspondences between y_i to x_j dynamically based on the amounts of displacements between adjacent frames (Figure 9). For time warping examples, please see the Supplementary Video.

6. Discussion

6.1. Limitations

As detailed in Table 1, the computational complexity of the proposed decomposition technique scales with the number of frames to estimate owing to the need to invert the $N_f \times N_f$ matrix in Equation (6). This can be further improved by employing more advanced approximation techniques, such as that suggested by Quinonero-Candela *et al.* [QCRW07]. A further speed-up can also be achieved by parallelizing our decomposition process, because the process can easily be applied to each joint independently.

The current implementation deals with kinematic constraints automatically. In certain cases, a smooth transition between constraints is necessary. For instance, if a given motion has dense constraints, such as a foot contact during a stationary motion, smoothness in the final motion can be broken due to the density of the imposed constraints. We plan to incorporate a smooth transition between constraints to remedy this type of artefact.

Through controllable motion displacements, style variations can be synthesized by changing the α values in Equation (17). However, depending on the given motions, large values of $|\alpha|$ may produce

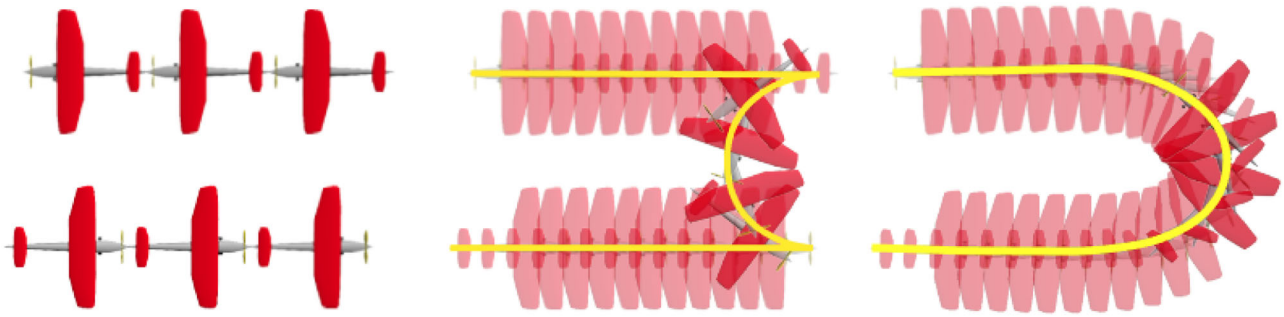


Figure 8: *A schematic experiment with sparse keyframes that include large rotational change (≥180°) between frames 3 and 4 (left: frames 1–6 counterclockwise from the bottom row). While* SLERP *fails to interpolate natural changes (middle), our method generates a correct path by considering the neighbouring rotations.*

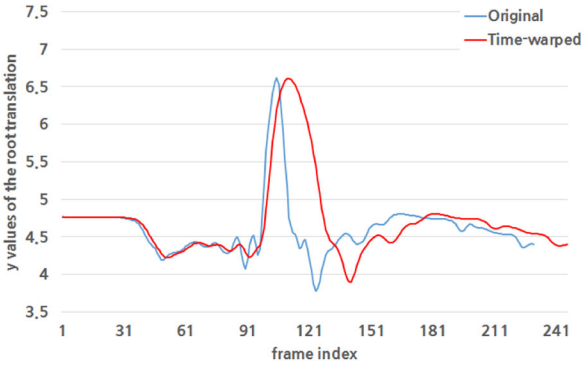


Figure 9: *The y values of the root translation of the original motion (blue) and their corresponding y values after the automatic time warping are applied (red). Note that the range with large motion displacements in the original motion (a time window around frame 110) is naturally highlighted with the effect of noticeable slowing down compared to the neighbouring time intervals.*

penetration artefacts as well as visually unnatural exaggeration in the final motion. One simple means of solving this problem in the current implementation is to impose joint limits during the synthesis process, although complex situations, such as self-penetration, would not be solved perfectly. A more sophisticated solution would be to consider spatio-temporal relationships during the synthesis process, which is similar to Ho *et al.* [HKT10].

6.2. Future work

While computing a posterior mean using the LGS, the posterior variance is also computed, as expressed by Equation (6). Though not used explicitly in the current implementation, this information can be used for a number of interesting applications. Specifically, for resampling and time warping scenarios, a system may be designed to ask for more data intelligently by resorting to a posterior variance similar to the concept of active learning. The animation quality for regions with a high degree of uncertainty can be improved by these additional data.

An interesting future direction would be to automatically find the values of relevant user parameters, such as λ for each motion layer, according to the type of motion. This would allow a consistent decomposition of the base-level and the controllable motion displacements for similar behaviours automatically, which may be vital to providing a large number of motion databases for data-driven techniques. In addition, a well-designed range of α values in Equation (17) could be used to sample various style variations automatically. We believe that such a systematic sampling strategy would provide major benefits to the data augmentation process, which has attracted growing interest from the deep-learning research community.

Our contact handling is still prone to minor foot sliding when dramatic style modulation is intended. Preserving essential motion properties to ensure physical realism will be an exciting and challenging direction to pursue. For example, it will be useful to predict new contact points for the modulated motion in order to automatically prevent penetration artefacts. Furthermore, adopting a physics-based motion controller [HNJ*14, HEN*16] would help produce both visually and physically correct motion that preserves physical properties, such as velocity and acceleration after the style modulation.

While we focused on estimations of decomposed motion layers that were as smooth as possible, we plan to explore possible new choices for the priors. By modelling different or additional priors for each layer, we expect that interesting motions can be procedurally synthesized. As an extension to modelling various priors, it will also be interesting to design flexible user interfaces to handle various aspects of our system, an example being compositing the final motion by modulating local prior weights via easy-to-use user interfaces similar to the work of Choi *et al.* [CBiRL*16].

7. Conclusion

We formulated our decomposition model with a purpose in mind, indeed inspired by motion editors' everyday workflow as well as the recent advances in machine learning techniques. All of the suggested applications were motivated by the typical motion pre-processing stage that must be undertaken before further high-level edits.

Technically, the approximation of stable rotations is the primary concern motivated by the difficulty of achieving them using existing solutions. To this end, we presented a general and flexible decomposition framework for motion synthesis, which provides a smoothly synthesized final motion that is essential for high-quality character animations and subsequent high-level motion processing. Instead of interpolating a given motion, we estimated new postures using linear Gaussian systems. As a result, all of the synthesized motions in our experiments demonstrated stable and smooth approximations of orientations and rotations. We also established the benefits of our approach when it is used to synthesize spatial and temporal variations of a given motion, which in turn, provides useful applications for denoising, style modulation, resampling and time warping in one unified framework.

Acknowledgements

We thank the anonymous reviewers for their valuable comments. This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (NRF-2017R1A2A1A05000979).

References

[Bou13] Boumal N.: Interpolation and regression of rotation matrices. In *Geometric Science of Information*, vol. 8085 LNCS of *Lecture Notes in Computer Science*. Springer Berlin Heidelberg (2013), pp. 345–352.

[BSH15] Brandt C., Seidel H.-P., Hildebrandt K.: Optimal spline approximation via ℓ 0-minimization. *Computer Graphics Forum 34*, 2 (22 May 2015), 617–626.

[BW95] Bruderlin A., Williams L.: Motion signal processing. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1995), SIGGRAPH '95, ACM, pp. 97–104.

[CBiRL*16] Choi B., Blanco i Ribera R., Lewis J., Seol Y., Hong S., Eom H., Jung S., Noh J.: Sketchimo: Sketch-based motion editing for articulated characters. *ACM Transactions on Graphics (TOG) 35*, 4 (2016), 146.

[CS07] Calvetti D., Somersalo E.: *An Introduction to Bayesian Scientific Computing: Ten Lectures on Subjective Computing*, vol. 2. Springer Science & Business Media, 2007.

[FHK*98] Fangt Y. C., Hsieh C. C., Kim M. J., Chang J. J., Woo T. C.: Real time motion fairing with unit quaternions. *Computer Aided Design and Applications 30*, 3 (Mar. 1998), 191–198.

[FJX*14] Feng Y., Ji M., Xiao J., Yang X., Zhang J. J., Zhuang Y., Li X.: Mining spatial-temporal patterns and structural sparsity for human motion data denoising. *IEEE Transactions on Cybernetics 45*, 12 (2014), 2693–2706.

[GMHP04] Grochow K., Martin S. L., Hertzmann A., Popović Z.: Style-based inverse kinematics. *ACM Transactions on Graphics (TOG) 23* (2004), 522–531.

[Gra98] Grassia F. S.: Practical parameterization of rotations using the exponential map. *Journal of Graphics Tools 3*, 3 (1998), 29–48.

[HdSP07] Hsu E., da Silva M., Popović J.: Guided time warping for motion editing. In *Proceedings of the 2007 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2007), Eurographics Association, pp. 45–52.

[HEN*16] Han D., Eom H., Noh J., Shin J. S.: Data-guided model predictive control based on smoothed contact dynamics. *Computer Graphics Forum 35* (2016), 533–543.

[HJ10] Hui L., Jinxiang C.: Example-based human motion denoising. *IEEE Transactions on Visualization and Computer Graphics 16*, 5 (Sept. 2010), 870–879.

[HKT10] Ho E. S., Komura T., Tai C.-L.: Spatial relationship preserving character motion adaptation. *ACM Transactions on Graphics (TOG) 29* (2010), 33.

[HNJ*14] Han D., Noh J., Jin X., S Shin J., Y Shin S.: On-line real-time physics-based predictive motion control with balance recovery. *Computer Graphics Forum 33* (2014), 245–254.

[Hol18] Holden D.: Robust solving of optical motion capture data by denoising. *ACM Transactions on Graphics (TOG) 37*, 4 (2018), 165.

[HSK16] Holden D., Saito J., Komura T.: A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics (TOG) 35*, 4 (2016), 138.

[HSKJ15] Holden D., Saito J., Komura T., Joyce T.: Learning motion manifolds with convolutional autoencoders. In *SIGGRAPH Asia 2015 Technical Briefs* (2015), ACM, p. 18.

[IAF09] Ikemoto L., Arikan O., Forsyth D.: Generalizing motion edits with Gaussian processes. *ACM Transactions on Graphics 28*, 1 (1 Jan. 2009), 1–12.

[JS02] Jehee L., Sung Y. S.: General construction of time-domain filters for orientation data. *IEEE Transactions on Visualization and Computer Graphics 8*, 2 (2002), 119–128.

[KKS95] Kim M.-J., Kim M.-S., Shin S. Y.: A general construction scheme for unit quaternion curves with simple high order derivatives. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1995), SIGGRAPH '95, ACM, pp. 369–376.

[KPS03] Kim T.-H., Park S. I., Shin S. Y.: Rhythmic-motion synthesis based on motion-beat analysis. *ACM Transactions on Graphics (TOG) 22* (2003), 392–401.

[LBJK09] Lau M., Bar-Joseph Z., Kuffner J.: Modeling spatial and temporal variation in motion data. *ACM Transactions on Graphics 28*, 5 (Dec. 2009), 171:1–171:10.

[LCB06] Le Callennec B., Boulic R.: Robust kinematic constraint detection for motion data. In *Proceedings of the 2006*

ACM SIGGRAPH/Eurographics Symposium on Computer Animation (2006), Eurographics Association, pp. 281–290.

[LDH14] Lang M., Dunkley O., Hirche S.: Gaussian process kernels for rotations and 6D rigid body motions. In *2014 IEEE International Conference on Robotics and Automation (ICRA)* (May 2014), IEEE, pp. 5165–5170.

[Lee08] Lee J.: Representing rotations and orientations in geometric computing. *IEEE Computer Graphics and Applications 28*, 2 (Mar. 2008), 75–83.

[LS96] Lee J., Shin S. Y.: Motion fairing. In *Proceedings Computer Animation'96* (1996), IEEE, pp. 136–143.

[LS99] Lee J., Shin S. Y.: A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (1999), ACM Press/Addison-Wesley Publishing Co., pp. 39–48.

[LS01] Lee J., Shin S. Y.: A coordinate-invariant approach to multiresolution motion analysis. *Graphical Models 63*, 2 (2001), 87–105.

[LS02] Lee J., Shin S. Y.: General construction of time-domain filters for orientation data. *IEEE Transactions on Visualization and Computer Graphics 8*, 2 (2002), 119–128.

[LT01] Lim I. S., Thalmann D.: Key-posture extraction out of human motion data. In *Proceedings of the 23rd Annual International Conference of the IEEE Engineering in Medicine and Biology Society, 2001*. (2001), vol. 2, pp. 1167–1169.

[LWH*12] Levine S., Wang J. M., Haraux A., Popović Z., Koltun V.: Continuous character control with low-dimensional embeddings. *ACM Transactions on Graphics (TOG) 31*, 4 (2012), 28.

[LZZ*19] Li S., Zhou Y., Zhu H., Xie W., Zhao Y., Liu X.: Bidirectional recurrent autoencoder for 3D skeleton motion data refinement. *Computers & Graphics 81* (2019), 92–103.

[MLCC17] Mall U., Lal G. R., Chaudhuri S., Chaudhuri P.: A deep recurrent framework for cleaning motion capture data. *arXiv preprint arXiv:1712.03380* (2017).

[Mur12] Murphy K. P.: *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[MXH*10] Ma W., Xia S., Hodgins J. K., Yang X., Li C., Wang Z.: Modeling style and variation in human motion. In *Proceedings of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2010* (2010), pp. 21–30.

[QCRW07] Quinonero-Candela J., Rasmussen C. E., Williams C. K.: Approximation methods for Gaussian process regression. In *Large-scale Kernel Machines*. L. Bottou, O. Chapelle, D. DeCoste and J. Weston (Eds.). MIT Press (2007), pp. 203–223.

[RB97] Ramamoorthi R., Barr A. H.: Fast construction of accurate quaternion splines. In *Proceedings of the 24th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1997), SIGGRAPH '97, ACM Press/Addison-Wesley Publishing Co., pp. 287–292.

[RCB98] Rose C., Cohen M. F., Bodenheimer B.: Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications 18*, 5 (Sept. 1998), 32–40.

[SCF04] Shapiro A., Cao Y., Faloutsos P.: Interactive motion decomposition. In *ACM SIGGRAPH 2004 Sketches* (New York, New York, USA, 8 Aug. 2004), ACM, p. 30.

[SCF06] Shapiro A., Cao Y., Faloutsos P.: Style components. In *Proceedings of Graphics Interface 2006* (2006), Canadian Information Processing Society, pp. 33–39.

[Sho85] Shoemake K.: Animating rotation with quaternion curves. In *Proceedings of the 12th Annual Conference on Computer Graphics and Interactive Techniques* (1 July 1985), vol. 19, ACM, pp. 245–254.

[TL08] Tay M. K. C., Laugier C.: Modelling smooth paths using Gaussian processes. In Laugier, C., Siegwart, R. (eds.) *Field and Service Robotics*. Springer Tracts in Advanced Robotics, vol. 42, pp. 381–390. Springer, Heidelberg (2008).

[UAT95] Unuma M., Anjyo K., Takeuchi R.: Fourier principles for emotion-based human figure animation. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 1995), SIGGRAPH '95, ACM, pp. 91–96.

[UK12] Ukita N., Kanade T.: Gaussian process motion graph models for smooth transitions among multiple actions. *Computer Vision and Image Understanding 116*, 4 (Apr. 2012), 500–509.

[Vas02] Vasilescu M. A. O.: Human motion signatures: Analysis, synthesis, recognition. In *Proceedings of the 16th International Conference on Pattern Recognition, 2002*. (2002), vol. 3, pp. 456–460.

[WDAC06] Wang J., Drucker S. M., Agrawala M., Cohen M. F.: The cartoon animation filter. *ACM Transactions on Graphics (TOG) 25* (2006), 1169–1173.

[WFH08] Wang J. M., Fleet D. J., Hertzmann A.: Gaussian process dynamical models for human motion. *IEEE Transactions on Pattern Analysis and Machine Intelligence 30*, 2 (Feb. 2008), 283–298.

[WP95] Witkin A., Popovic Z.: Motion warping. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques - SIGGRAPH '95* (New York, New York, USA, 1995), ACM Press, pp. 105–108.

[XFJ*15] Xiao J., Feng Y., Ji M., Yang X., Zhang J. J., Zhuang Y.: Sparse motion bases selection for human motion denoising. *Signal Processing 110* (May 2015), 108–122.

[ZSD12] Zhu M., Sun H., Deng Z.: Quaternion space sparse decomposition for motion compression and retrieval. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, 2012), P. Kry and J. Lee (Eds.), SCA '12, Eurographics Association, pp. 183–192.

[ZSSL14] Zhou L., Shang L., Shum H. P. H., Leung H.: Human motion variation synthesis with multivariate Gaussian processes. *Computer Animation and Virtual Worlds 25*, 3–4 (May 2014), 301–309.

Supporting Information

Additional supporting information may be found online in the Supporting Information section at the end of the article.

Data video S1