Geostatistical Motion Interpolation

Tomohiko Mukai* Shigeru Kuriyama†
Toyohashi University of Technology

Figure 1: Animations synthesized by our motion interpolation in a 5D parametric space. One parameter changes the style of motion from *rough* to *delicate* as shown by the bar indicator. The other four parameters are the heights and widths of two successive steps of stairs for gait motions, and the 2D start and end locations of the box for lifting motions. None of the motions required post-cleaning of foot- or hand-sliding.

Abstract

A common motion interpolation technique for realistic human animation is to blend similar motion samples with weighting functions whose parameters are embedded in an abstract space. Existing methods, however, are insensitive to statistical properties, such as correlations between motions. In addition, they lack the capability to quantitatively evaluate the reliability of synthesized motions. This paper proposes a method that treats motion interpolations as statistical predictions of missing data in an arbitrarily definable parametric space. A practical technique of geostatistics, called universal kriging, is then introduced for statistically estimating the correlations between the dissimilarity of motions and the distance in the parametric space. Our method statistically optimizes interpolation kernels for given parameters at each frame, using a pose distance metric to efficiently analyze the correlation. Motions are accurately predicted for the spatial constraints represented in the parametric space, and they therefore have few undesirable artifacts, if any. This property alleviates the problem of spatial inconsistencies, such as foot-sliding, that are associated with many existing methods. Moreover, numerical estimates for the reliability of predictions enable motions to be adaptively sampled. Since the interpolation kernels are computed with a linear system in real-time, motions can be interactively edited using various spatial controls.

CR Categories: I.3.7 [Computer Graphics]: Three-dimensional Graphics and Realism—Animation G.3 [Probability and Statistics]: Correlation and Regression Analysis

Keywords: motion interpolation, humanoid animation, motion dataset, geostatistics, statistical prediction, kriging, variogram

*e-mail: mukai@val.ics.tut.ac.jp
†e-mail: kuriyama@ics.tut.ac.jp

1 Introduction

Motion editing via interactive posing is commonly used in creating character animations, which often requires flexible controls of various geometric constraints. However, interactive applications such as video games pre-determine motion control parameters (or types of constraints) to simplify the user's operations, and require very quick and robust techniques for motion generation. Direct blending of entire motions is essentially suited to these requirements, and particularly, interpolation of sampled motions is a convenient technique when a large motion dataset is available.

Motion interpolation parameterizes all motion samples in the abstract space defined for controlling motions with kinematical, physical, or emotional attributes, and multiple motions are then smoothly blended with kernel functions. Formally speaking, given the vector of parameters $\mathbf{c}$ for controlling a motion in a multidimensional real abstract space, motion interpolation generates a new motion $\mathbf{M}(\mathbf{c})$ by the weighted average of the motion samples $\mathbf{M}_i$ that have corresponding parameter $\mathbf{c}_i$ with kernel functions, denoted by $\mathbf{b}(\mathbf{c}) = \{b_1(\mathbf{c}), b_2(\mathbf{c}), \ldots, b_N(\mathbf{c})\}$, as follows:

$$\mathbf{M}(\mathbf{c}) = \sum_{i=1}^{N} b_i(\mathbf{c})\,\mathbf{M}_i$$

where N is the number of motion samples. The key issue is the design of the kernel functions whose values decrease according to the distance between $\mathbf{c}$ and $\mathbf{c}_i$, for example, using the inverse of the distance, spline bases, or Gaussian distributions. In all cases, the kernel functions should also be normalized to always sum to unity. Our contribution is to introduce geostatistics for creating alternative kernel functions that have some advantageous properties.

Most existing methods neglect the effects on the kernels caused by changes in the type of motion, the properties of parameters, and the passage of time. These defects easily cause inconsistent motions, for example, geometric errors in foot positions on floors or hand positions on handled objects. Such errors are mostly corrected by adjusting free skeletal parameters, for example, using inverse kinematics [Kovar and Gleicher 2002]. However, the plausibility of synthesized motions is not ensured. Other corrective techniques use

heuristics for adjusting the parameters of the kernels so that the resulting motion satisfies given constraints. Such strategies, however, force users to select adequate kernels and fine-tuned parameters, and their automatic optimization often requires vast computations [Rose et al. 2001]. Moreover, proposed motion interpolations utilize few statistical properties, and analyze only the variations of motion samples in a parametric space while ignoring correlations and temporal coherence.

Natural human motions essentially contain random variations, usually vary continuously with respect to relevant parameters, and their distributions have strong coherence. This property is well-suited to spatial statistics [Cressie 1993], which is a theory of statistical analysis concerning data that have spatial or temporal structures. We therefore propose motion interpolation based on spatial statistics by regarding all motion clips as spatial samples distributed in a multidimensional abstract space. We call this parametric space for controlling motion interpolation a *control space*. Parameters embedded in this control space are called *control parameters*, by which arbitrary spatial constraints are continuously represented. With this strategy, existing problems with spatial inconsistency, such as foot-sliding, can be directly eliminated by accurately predicting motions in this space. In addition, the control space can involve style parameters, and integrating spatial constraints with such parameters can enhance the expressive richness of synthesized motions. Figure 1 demonstrates our motion interpolation method with an example of stylized walking on irregularly-shaped stairs and the lifting of a box. Four dimensional spatial constraints and one style parameter are together embedded in a 5D control space. This example shows our ability to control motions using heterogeneous parameters in a high-dimensional control space.

This paper introduces a statistical approach to motion prediction, by which the correlations among sampled motions are considered. More concretely, our method estimates a statistical model that explains the correlation between the control parameters and motion samples in order to compute optimal interpolation kernels. The features of proposed method are outlined as follows:

Accuracy and robustness Motion samples parameterized in the control space are interpolated with kernels that are automatically and robustly optimized in accordance with statistical observations. Synthesized motions are therefore statistically accurate even for irregularly-distributed samples.

Interactive manipulation Most parts of the motion prediction can be preprocessed, and motions are interpolated using a simple linear system in real-time. This property enables interactive motion manipulations even in a high-dimensional control space.

Reliability estimate Reliability of interpolations can be estimated through predictive error variance, and can be used to reduce redundant samples and to assess the validity of regions in the control space.

In the following section, we explain related work and propose our method for motion interpolation in Section 3. Section 4 explains kernel computation, which is the core of our method. Experimental results are shown in Section 5, comparing the performance with radial basis function interpolations, and we discuss our conclusions in Section 6.

2 Related Work

Several approaches to motion interpolation have been proposed involving parameterization of motion samples in the frequency domain, using Fourier coefficients [Unuma et al. 1995] and hierarchical filtering [Bruderlin and Williams 1995]. These methods provide simple and intuitive control of motion blending. They are, however, ill-suited to non-cyclic short motion clips since frequency analysis is problematic on such data. A linear interpolation technique with stochastic sampling [Wiley and Hahn 1997] can manage general motions, but the computational cost increases exponentially with the number of motion samples.

Our method is closely related to the verbs and adverbs system [Rose et al. 1998] that introduced radial basis functions (RBFs) for interpolating motions with multiple parameters. This method was extended to solve inverse kinematics problems [Rose et al. 2001] and to interactively generate locomotion [Park et al. 2004]. RBFs can smoothly interpolate scattered motion samples, and the number, types, and shapes of the basis functions are optimized with the samples. These methods, however, neglect the variations of the functions with the passage of time, which often causes unexpected distortions in the synthesized motions. Such distortions can be reduced by naturally proliferating similar motions; for example, by using a heuristic algorithm to adaptively add pseudo-examples [Rose et al. 2001] or stochastic sampling to interpolate the K-nearest neighboring motions [Kovar and Gleicher 2004]. Another method of proliferation parameterizes motions in a dynamic space [Abe et al. 2004] to physically transform them. These methods have no inverse mapping from constrained conditions to a pose space except the locations of given samples. For this reason, they need to generate many similar motions by adaptively sampling control parameters to cover the continuous space of constraints. This random sampling approach, however, requires vast computation and storage to satisfy constrained conditions with reasonable accuracy, especially for a high-dimensional parametric space. In contrast, our statistical prediction directly determines the optimal values for kernel functions from the constraints given as control parameters.

Our method has some relation to multivariate analysis, which is introduced to intelligently classify motions to efficiently construct a state-transition graph. For example, K-means clustering [Tanco and Hilton 2000] and expectation maximization [Lee et al. 2002] are used to automatically group similar motions. Another method annotates motion clips [Arikan et al. 2003] using a support vector machine, and a similar strategy is later applied to the discrimination of natural motions [Ikemoto and Forsyth 2004]. The other usage of multivariate analysis is feature extraction. Principal component analysis (PCA) extracts essential bases of motion curves, and is applied to data reduction [Alexa and Muller 2000] and to dynamic optimization in a low-dimensional space [Safonova et al. 2004]. PCA, however, analyzes constituent poses independently of their temporal structures, and such limitation is common with pose-based map organization [Sakamoto and Kuriyama 2004]. On the other hand, ST-Isomap, which has been used to nonlinearly compute a lower-dimensional structure for cartoon data [de Juan and Bodenheimer 2004], can retain spatial and temporal coherences. These dimensionality reduction methods implicitly and indirectly determine control parameters along each axis of the reduction space, and thus constrained conditions are difficult to impose explicitly.

Recently, the scaled Gaussian process latent variable model (SGPLVM) has been proposed [Grochow et al. 2004] for interpolating poses from given constraints in a low-dimensional space, using a simple kernel estimator. This method can efficiently search for plausible poses from constrained curves such as positional trajectories of end-effectors. Our method has a similarity to the SGPLVM in giving constrained conditions with embedded parameters and in eliminating defects in RBF interpolation, but differs in the way parameters are embedded. The SGPLVM gives parameters at each pose, velocity, and acceleration, for considering spatial and local temporal dependencies, and it provides an abstract control space through learning. On the other hand, our method gives control parameters for each entire motion, allowing the correlation and time-coherence of motion samples to be fully exploited, and a control

space is explicitly provided by the user. SGPLVM optimizes interpolation kernels through the estimation of the probability distribution function representing similarity between samples, while our method optimizes them by estimating their dissimilarity. Moreover, our method automatically selects the optimal model for kernels whereas SGPLVM uses only Gaussian kernels.

3 Motion Interpolation

3.1 Theoretical background

Our method uses the practical application of spatial statistics, which is called *geostatistics* [Wackernagel 2003], that predicts continuous distributions of interpolating variables from the samples given with corresponding parameters. Geostatistics was originally invented by a French school of mines for estimating amounts of mineral resources, and has been widely used for spatial prediction of various phenomena in geographic information systems, environmental science, fisheries oceanography, forestry, and so on. It assumes that the correlation between samples is determined from the spatial distance (or Euclidean norm) of their control parameters alone, and uniquely estimates a statistical model, called *variogram function*, that explains the relation between the distance of parameters and the dissimilarity of corresponding samples. The dissimilarity is defined using the squared difference between two samples, and it is assumed to monotonically increase with respect to the distance. The details of the variogram function is later explained in Section 4.2.

Kriging, named for a pioneer D.G. Krige, is a generic term for spatial prediction based on geostatistics, and ordinary kriging is a best linear unbiased prediction of random function. It is also considered as a form of Gaussian process regression, similar to SGPLVM [Grochow et al. 2004], because it assumes that sampled data contain a certain degree of noise that is independently and identically distributed. Ordinary kriging requires offline adjustments through user observation, and mostly manages a 2D space, whereas Gaussian processes in machine learning regularly manage much higher dimensional spaces. Our contribution is the adaptation of kriging to motion interpolations that require online computations in the control space of arbitrary dimensions.

Let $\mathbf{c} = [c_1, c_2, \ldots, c_D]$ be the vector of control parameters in $\mathfrak{C}$, where $\mathfrak{C}$ represents a D-dimensional real abstract space, and let $\mathbf{c}_i$ be the value corresponding to the i-th sample. In the probabilistic model, a variable $s(\mathbf{c})$ is considered to be a realization of a random function $S(\mathbf{c})$ (i. e. an infinite family of random variables constructed at all points $\mathbf{c} \in \mathfrak{C}$). Kriging estimates a distribution of the random function from the noisy samples, and ordinary kriging requires the following condition, called *intrinsic stationarity*, to be satisfied in the random function [Cressie 1993].

Intrinsic stationarity : For arbitrary pairs of $\mathbf{c}_i, \mathbf{c}_j \in \mathfrak{C}$

$$E\{S(\mathbf{c}_i) - S(\mathbf{c}_j)\} = 0,\ Var\{S(\mathbf{c}_i) - S(\mathbf{c}_j)\} = \gamma(\mathbf{c}_i - \mathbf{c}_j)$$

hold, where $E\{\cdot\}$ denotes an expectation, $Var\{\cdot\}$ is a variance, and γ represents a variogram function.

The intrinsic stationarity therefore assumes the stationarity of the first and second moments of the difference of $S(\mathbf{c}_i)$ and $S(\mathbf{c}_j)$. These conditions are seldom satisfied for motion data; for example, the average of leg-joint rotations varies according to the length of stride, which breaks the first stationarity, and knee rotations are changed by the positional changes of feet more in the vertical than the horizontal direction, which breaks the second stationarity. To overcome these disagreements, ordinary kriging is extended by assuming a component of variations that are unrelated to random function, and such analysis is called *universal kriging* [Huijbregts and Matheron 1971].

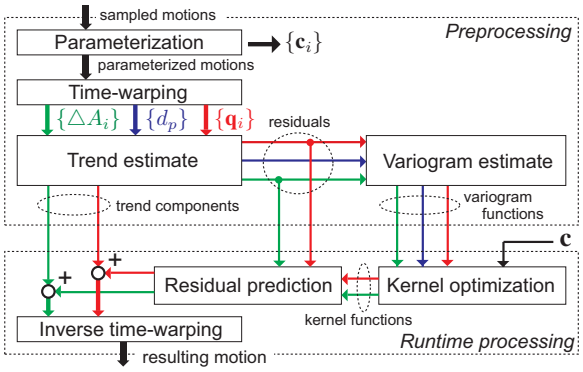


Figure 2: Diagram of motion interpolation.

Universal kriging divides each sample s_i at $\mathbf{c}_i$ into a trend component $m(\mathbf{c}_i)$ and its residual $r(\mathbf{c}_i)$ as

$$s_i = m(\mathbf{c}_i) + r(\mathbf{c}_i)$$

where $m(\mathbf{c}_i)$ is a deterministic component and $r(\mathbf{c}_i)$ is a random component that includes white noise. In other words, this assumes that the trend $m(\mathbf{c}_i)$ is deterministically computed from the control parameter $\mathbf{c}_i$ and that the residual $r(\mathbf{c}_i)$ is a realization of a random function satisfying intrinsic stationarity. Consequently, the computation of universal kriging proceeds in two steps: the first step estimates the trend, and the second step executes kriging on the residuals.

The predictive residual $r^*(\mathbf{c})$ is computed by blending the residuals of the samples $r_i = s_i - m(\mathbf{c}_i)$ with kernel functions $\mathbf{b}(\mathbf{c})$ whose values are a partition of unity:

$$r^*(\mathbf{c}) = \sum_{i=1}^{N} b_i(\mathbf{c})\, r_i\ ,\quad \sum_{i=1}^{N} b_i(\mathbf{c}) = 1$$

and the predictive value $s^*(\mathbf{c})$ is thus given by $s^*(\mathbf{c}) = m(\mathbf{c}) + r^*(\mathbf{c})$.

Figure 2 shows the diagram of our motion interpolation. The process is divided into two phases: preprocessing and runtime processing. In the preprocessing phase, all motion samples are parameterized in a control space and are time-aligned with time-warp functions. Trend components and variogram functions are then estimated for time-warped samples $\{\mathbf{q}_i\}$ and incremental time-warping $\{\triangle A_i\}$ with the corresponding control parameters $\{\mathbf{c}_i\}$, where pose distance metrics $\{d_p\}$ are also computed for increasing efficiency, as explained in Section 3.4. The runtime processing computes kernel functions with the given control parameters $\mathbf{c}$ and predicts residual components. The predicted motion and time-warp functions are obtained by adding their residuals to the trend, and the motion is finally re-transformed with the estimated inverse time-warp function.

3.2 Per-element interpolation

Each skeletal pose of a humanoid is formally represented by a pose vector $\mathbf{q}(t) = \{q_1(t), q_2(t), \ldots, q_{\dim(\mathbf{q})}(t)\}$ with a discrete time (or frame) t, where $\dim(\mathbf{q})$ denotes the total degrees of freedom of the humanoid skeleton. The elements of $\mathbf{q}(t)$ contain the rotation of all joints and a global 3D position and orientation of the root node that is located near the abdomen, and all rotational elements are represented by an exponential map. Each sampled motion $\mathbf{M}_i$ is then represented as a time series of the pose vector; $\mathbf{M}_i = \{\mathbf{q}_i(0), \mathbf{q}_i(1), \ldots, \mathbf{q}_i(T_i)\}$, where T_i denotes the duration of the i-th motion.

Since kriging basically predicts a scalar value, each element of the pose vector must be separately predicted by assuming independence of the control space. In addition, we consider that the kernel values should change according to the passage of time. For these reasons, our basic approach computes the kernel for each element of the pose vector at each frame.

Let $q_i^k(t)$ be the k-th element of the i-th motion sample $\mathbf{q}_i(t)$, and let $\mathbf{c}_i$ be the corresponding control parameters. In universal kriging, the k-th element of the synthesized pose $q^k(t, \mathbf{c})$ is represented as:

$$q^k(t, \mathbf{c}) = m^k(t, \mathbf{c}) + \sum_{i=1}^{N} b_i(\mathbf{c})\, r_i^k(t) \tag{1}$$

where $m^k(t, \mathbf{c})$ represents the trend component and $r_i^k(t) = q_i^k(t) - m^k(t, \mathbf{c}_i)$ is the k-th residual. Kernel functions $\mathbf{b}(\mathbf{c})$ are composed with the set of residuals $\mathbf{R}^k(t) = \{r_1^k(t), r_2^k(t), \ldots, r_N^k(t)\}$ and the set of the control parameters $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_N\}$ for N motion samples. This means that the continuous distribution of samples in $\mathfrak{C}$ is estimated through the statistical observations of $\mathbf{R}^k(t)$ and $\mathbf{C}$.

3.3 Trend surface

The trend component often can be represented by polynomials. High-dimensional polynomials are suited to fitting complicated trends, but are redundant for the samples whose variations are small. Motion data usually vary continuously according to the change of control parameters, and we therefore estimate the trend with a D-dimensional hyperplane

$$m^k(t, \mathbf{c}) = l_0 + l_1 c_1 + \cdots + l_D c_D \ .$$

The variables l_i of the trend hyperplane are uniquely determined by solving minimization problems of the form:

$$\{l_0, l_1, \ldots, l_D\} = \arg \min_{l_0, l_1, \ldots, l_D} \left[\sum_{i=1}^{N} \left(m^k(t, \mathbf{c}_i) - q_i^k(t) \right)^2 \right] \tag{2}$$

using a least squares approximation with a pseudo-inverse matrix.

3.4 Per-pose interpolation

The above per-element approach may predict motion more accurately than using an identical kernel for all variables, because it can analyze detailed individual distributions of each element. However, this neglects the correlations between elements, and the cost of computation and storage linearly increases according to $\dim(\mathbf{q})$. We therefore introduce kriging prediction based on the pose distance metric $d_p(\mathbf{q}_i, \mathbf{q}_j)$ [Kovar et al. 2002] that represents the dissimilarity between the two poses $\mathbf{q}_i$ and $\mathbf{q}_j$, by which their visual variation is quantified. This strategy relies on the correlation between the distance of the control parameter and the visual appearance of the pose. This estimates an identical kernel that interpolates all joint rotations through the prediction of a distribution of d_p among motion samples, instead of kriging every element separately. All joints are therefore assumed to have a common variogram function.

The variogram function is estimated using the residuals of the pose distance metric between sampled poses $\mathbf{q}_i(t)$ and the initial pose $\mathbf{q}_1(0)$ of the reference motion $\mathbf{M}_1$. The residuals, however, should be separately computed for each element by estimating their trend components with the assumption of independence of their randomness. Therefore, the k-th element $q^k(t, \mathbf{c})$ of the resulting pose is given by computing the kernel functions $\mathbf{b}(\mathbf{c})$ in Equation 1 with the set of the residuals of pose distances $\mathbf{R}^d(t) = \{r_1^d(t), r_2^d(t), \ldots, r_N^d(t)\}$, $r_i^d(t) = d_p(\mathbf{q}_i(t), \mathbf{q}_1(0)) - m^d(t, \mathbf{c}_i)$ and the set of the control parameters $\mathbf{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_N\}$. Notice that the trend $m^d(t, \mathbf{c})$ is computed by replacing $q_i^k(t)$ with $d_p(\mathbf{q}_i(t), \mathbf{q}_1(0))$ in Equation 2, and the trend $m^k(t, \mathbf{c})$ and residuals $r_i^k(t)$ are computed in the same way as the abovementioned per-element interpolation.

The pose distance metric, however, does not include the state of the root node consisting of a global 3D position and orientation along the vertical axis. These four variables are therefore predicted independently by computing $\mathbf{b}(\mathbf{c})$ in the same way as the per-element interpolation. This strategy can reduce the computational and storage costs in optimizing kernels by a ratio of $5/\dim(\mathbf{q})$ compared to the simple per-element interpolation.

3.5 Time warping

Sampled motions require normalizing in the time domain in order to ensure temporal correspondence among them. We assume that the control parameters $\mathbf{c}$ also alter the temporal variables such as speed, key-timing, and duration whose consistency must be ensured for reliable interpolations. All motion samples are therefore time-aligned to establish their temporal correspondence using a time-warp function $A_{i \to j}(t_i) = t_j$ that is determined by manually selecting corresponding poses $\mathbf{q}_i(t_i)$, $\mathbf{q}_j(t_j)$ of two motion samples $\mathbf{M}_i$, $\mathbf{M}_j$. The motion $\mathbf{M}_i$ is then time-aligned to be consistent with $\mathbf{M}_j$.

We use a coordinate-invariant time-warping function [Kovar and Gleicher 2003] that searches for an optimal function $A_{i \to j}$ via dynamic time-warping so as to minimize the sum of the pose distance metric during a motion sequence. Notice that the inverse of the time-warp function $A_{i \to j}^{-1} = A_{j \to i}$ is also computed in a similar way (see [Kovar and Gleicher 2003; Kovar and Gleicher 2004] for details). Our method first manually selects a reference motion $\mathbf{M}_1$ and next computes time-warp functions $A_{i \to 1}$ for time-aligning each motion $\mathbf{M}_{i \neq 1}$ to make all key-timing consistent.

After all poses are interpolated for each frame of $\mathbf{M}_1$, the composed motion is re-transformed in the time domain by using the inverse time-warp function to obtain appropriate time sequence. The inverse time-warp is also represented as a function with respect to $\mathbf{c}$, and thus the parameterized inverse time-warp function $A_{1 \to c}$ is estimated similarly by kriging. This estimate, however, is made for the increments of the inverse time-warp functions, where the i-th value is denoted by $\triangle A_i(t) = A_{1 \to i}(t) - A_{1 \to i}(t-1)$, to ensure a monotonically increasing sequence [Park et al. 2004]. Let $m^a(t, \mathbf{c})$ be the trend of $\triangle A_i(t)$, then an incremental inverse time-warp function $\triangle A_c(t, \mathbf{c})$ for synthesized motion is given by

$$\triangle A_c(t, \mathbf{c}) = m^a(t, \mathbf{c}) + \sum_{i=2}^{N} b_i(\mathbf{c})\, r_i^a(t)$$

where the kernel functions $\mathbf{b}(\mathbf{c})$ are computed with the set of the residuals $\mathbf{R}^a(t) = \{r_2^a(t), r_3^a(t), \ldots, r_N^a(t)\}$, $r_i^a(t) = \triangle A_i(t) - m^a(t, \mathbf{c}_i)$ and the set of the control parameters $\mathbf{C}$. Note that the trend $m^a(t, \mathbf{c})$ is similarly computed by replacing $q_i^k(t)$ with $\triangle A_i(t)$ in Equation 2, and the kernels $\mathbf{b}(\mathbf{c})$ are optimized in a similar way to per-element interpolation.

4 Kernel Computation

Kernel functions are optimized by estimating a variogram function from the set of residuals $\mathbf{R}(t)$ that are computed by extracting a trend component from samples, with the assumption that their distribution satisfies intrinsic stationarity. The algorithm explained in this section is used in the per-element, per-pose, and inverse time-warp interpolations.

4.1 Experimental variogram

The variogram function of motion data is difficult to determine because the distribution of the residuals is highly random, as shown in

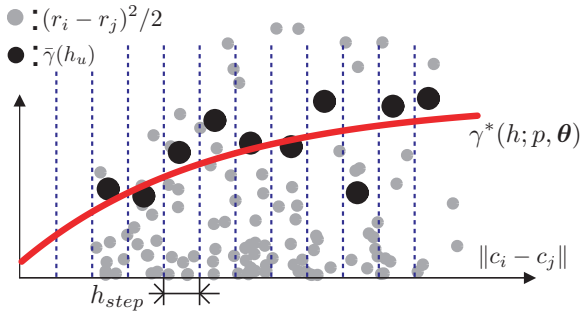


Figure 3: Experimental variogram and predictive variogram function. Small dots indicate the variations of residuals r_i and large dots indicate their averages $\bar{\gamma}(h_u)$ at each interval I_u. The curve $\gamma^*(h; p, \boldsymbol{\theta})$ represents the predictive variogram function fitted to the averages. This shows actual data of the pose distance metric for the reaching motion used in Section 5.1.

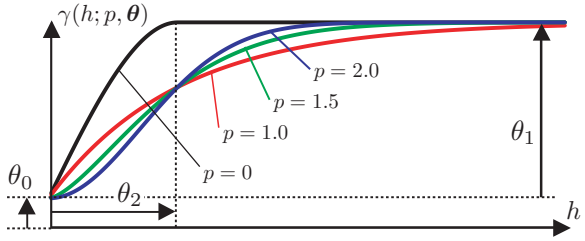


Figure 4: Theoretical variogram model: p determines the type of a variogram function, $\theta_0 \geq 0$ denotes the intensity of white noise, $\theta_1 \geq 0$ is the dissimilarity excluding noise at an infinite distance $h = \infty$, and $\theta_2 > 0$ is the distance at which the correlation vanishes.

Figure 3. We therefore compute *experimental variogram* $\bar{\gamma}$ by averaging the residuals within the uniformly divided regions, by which the effect of the white noise can be cancelled out. Universal kriging assumes the isotropic (i. e. rotation invariant) behavior of the underlying variogram, and estimates the variogram using the distance (or norm) $h = \|\mathbf{c}_i - \mathbf{c}_j\|$ between control parameters. The range of distance h is divided by each h_{step} into N_I intervals $I_u (u = 1, 2, \ldots, N_I)$ whose center is denoted by $h_u = h_{step}(u - 1/2)$, and the average of variations $\bar{\gamma}(h_u)$ is calculated in each I_u. The simple arithmetic average, however, lacks accuracy due to the effects of noisy error, and we therefore use a robust estimate proposed by Cressie [1980], which can minimize the effect of noise,

$$\bar{\gamma}(h_u) = \frac{1}{2} \frac{\left(\frac{1}{n(I_u)} \sum_{n(I_u)} |r_i - r_j|^{1/2}\right)^4}{0.457 + 0.494/n(I_u) + 0.045/n(I_u)^2}$$

where r_i is a residual at $\mathbf{c}_i$ and $n(I_u)$ is the number of the pairs of samples included in I_u.

The distribution of this experimental variogram is expected to have saturation curves similar to those in Figure 4. The actual samples, however, often generate unstable shapes in the large distance regions, and this property decreases estimate accuracy. We therefore use only the lower half range $[0, h_{\max}/2]$ for estimation [Wackernagel 2003], where $h_{\max}$ is the maximum distance among all samples.

4.2 Estimate of variogram function

The dissimilarity between samples is estimated from the distance h between corresponding control parameters, through representative theoretical variogram functions formulated as (see Figure 4):

$$\gamma(h; p, \boldsymbol{\theta}) = \begin{cases} \theta_0 + \frac{\theta_1}{2}\left\{\frac{3h}{\theta_2} - \left(\frac{h}{\theta_2}\right)^3\right\}, & 0 < h \leq \theta_2 \\ \theta_0 + \theta_1, & h > \theta_2 \end{cases} \quad \text{for } p = 0\ ,$$

$$\gamma(h; p, \boldsymbol{\theta}) = \theta_0 + \theta_1\left\{1 - \exp\left(-\left(\frac{h}{\theta_2}\right)^p\right)\right\},\ h > 0 \text{ for } 0 < p \leq 2$$

where $\gamma(0; p, \boldsymbol{\theta}) = 0$ always holds by definition.

Here we estimate a predictive variogram function, denoted by $\gamma^*(h; p, \boldsymbol{\theta})$, by optimizing the variables of the theoretical variogram functions to best approximate the experimental variogram $\bar{\gamma}(h_u)$, as shown in Figure 3. The optimal value of p is often manually selected through the observation of the shape of $\bar{\gamma}(h_u)$; we, however, compute it automatically by using cross validation [Wackernagel 2003]. Since the computational cost increases with the number of sampling frames, we assume that the type of a variogram function is time-invariant and compute the cross validation only at a representative frame. The value of p is therefore determined by manually selecting the frame at which the pose is relatively distinctive. The values of $\boldsymbol{\theta}$, however, are optimized for each frame because they are assumed to vary according to the passage of time. See Appendix A for details of this estimate.

The estimate of the predictive variogram function requires an appropriate value of h_{step}. A smaller h_{step} decreases the stability in optimization due to the fitting to many samples, and a larger h_{step} often misses important spatial variations. Journel and Huijbregts [1978] suggested that h_{step} should be set so that the number of pairs satisfies $n(I_u) \geq 30$ for each interval. However, it is not an easy task to capture the motion samples of so many variations. Our current implementation therefore divides the usable range of distance into $N_I = 10$ intervals, which can attain reasonable estimate accuracy.

4.3 Kernel optimization

Kriging optimizes kernel functions $\mathbf{b}(\mathbf{c}) = [b_1(\mathbf{c}), b_2(\mathbf{c}), \ldots, b_N(\mathbf{c})]$ to minimize the expectation of predictive error variance denoted by $\sigma_E^2(\mathbf{c})$ under the condition that $\sum_{i=1}^N b_i(\mathbf{c}) = 1.0$. This constrained optimization is efficiently solved using Lagrange multipliers λ. The objective function is then augmented as,

$$f(\mathbf{b}, \lambda) = \sigma_E^2(\mathbf{c}) - 2\lambda\left(\sum_{i=1}^N b_i(\mathbf{c}) - 1\right).$$

The value of $\sigma_E^2(\mathbf{c})$ is given by the variogram function $\gamma^*(h; p, \boldsymbol{\theta})$ under intrinsic stationarity as [Wackernagel 2003],

$$\sigma_E^2(\mathbf{c}) = 2\sum_{i=1}^N b_i(\mathbf{c})\gamma^*_{p,\theta}(\|\mathbf{c}_i - \mathbf{c}\|) - \sum_{i=1}^N\sum_{j=1}^N b_i(\mathbf{c})b_j(\mathbf{c})\gamma^*_{p,\theta}(\|\mathbf{c}_i - \mathbf{c}_j\|)$$

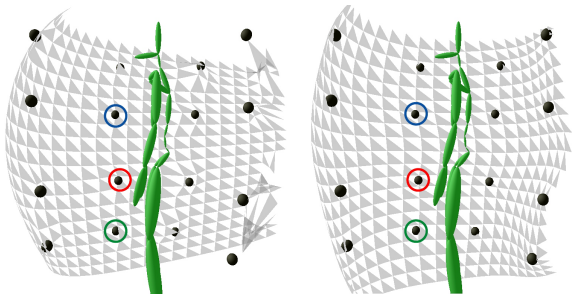
where $\gamma^*_{p,\theta}(h)$ is an abridged notation of $\gamma^*(h; p, \boldsymbol{\theta})$ The objective function $f(\mathbf{b}, \lambda)$ is a positive quadratic with respect to b_i and λ, and its minimum value can be computed by imposing $\partial f(\mathbf{b}, \lambda)/\partial b_i = \partial f(\mathbf{b}, \lambda)/\partial \lambda = 0$.

From these equations, optimal kernels are calculated as follows:

$$\begin{bmatrix} \mathbf{b}(\mathbf{c})^T \\ \lambda \end{bmatrix} = \begin{bmatrix} \boldsymbol{\Gamma} & \mathbf{1}^T \\ \mathbf{1} & 0 \end{bmatrix}^{-1} \begin{bmatrix} \boldsymbol{\gamma}^*(\mathbf{c})^T \\ 1 \end{bmatrix}, \quad \boldsymbol{\Gamma} = \{\gamma^*_{p,\theta}(\|\mathbf{c}_i - \mathbf{c}_j\|)\}_{ij}\ , \qquad (3)$$

$$\boldsymbol{\gamma}^*(\mathbf{c}) = [\gamma^*_{p,\theta}(\|\mathbf{c}_1 - \mathbf{c}\|), \gamma^*_{p,\theta}(\|\mathbf{c}_2 - \mathbf{c}\|), \ldots, \gamma^*_{p,\theta}(\|\mathbf{c}_N - \mathbf{c}\|)]$$

where $[\ ]^{-1}$ denotes an inverse matrix, $[\ ]^T$ denotes transpose, and $\mathbf{1} = [\overbrace{1, \ldots, 1}^{N}]$. Notice that the vector $\boldsymbol{\gamma}^*(\mathbf{c})$ coincides with the j-th column of the matrix $\boldsymbol{\Gamma}$ by assigning $\mathbf{c} = \mathbf{c}_j$. Since $\boldsymbol{\Gamma}$ is a non-singular matrix, this leads to the condition of interpolation $b_j(\mathbf{c}_j) = 1,\ b_{i\neq j}(\mathbf{c}_j) = 0$.



(a) RBF (b) Per-pose

Figure 5: Target positions of synthesized reaching motions. Black spheres represent the targets of the sampled motions and vertices on gray meshes denote those of the synthesized motions. The mesh in (a) generated by RBF interpolation shows large distortions near the samples (black spheres) and behind the body. In contrast, as shown in (b), per-pose interpolation based on kriging generates a continuous and smooth distribution even in the regions of extrapolation.

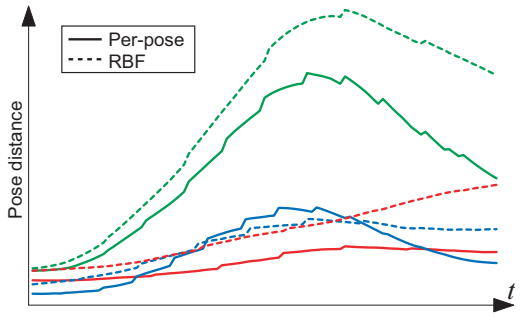


Figure 6: Change of the pose distance metric between motion sample and synthesized motion via cross validation. Each color represents the comparison at each target circled with the same color in Figure 5. Dotted curves show the errors of RBF interpolations, and solid curves show those of our per-pose interpolations.

Table 1: Comparison of prediction accuracy and time spent in runtime and preprocessing (msec/frame) measured on a dual 3.0 GHz Xeon CPU. The time spent in time-warping is omitted from the preprocessing because it is required for all methods.

Reliability threshold	RBF		Per-element		Per-pose	
	MAX	RMSE	MAX	RMSE	MAX	RMSE
$\rho \geq 0.9$	13.96	4.12	6.65	2.11	6.53	1.96
$\rho \geq 0.8$	36.64	10.56	14.52	3.29	14.99	3.31
$\rho \geq 0.7$	50.13	14.15	25.44	5.43	24.80	5.63
Runtime	0.28		8.39		0.78	
Preprocess	0		146.56		13.40	

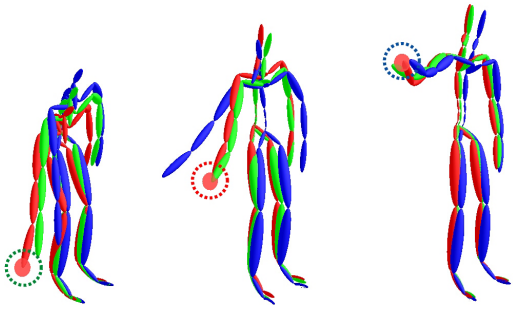


Figure 7: Comparison of sampled and synthesized motions via cross validation. Skeletal figures colored red show motion samples, while blue and green figures represent the synthesized motions with RBF and per-pose interpolations, respectively.

The computation of the inverse matrix in Equation 3 can be performed as a preprocess because all elements are computed without using the free control parameters $\mathbf{c}$. Consequently, at runtime this linear system only requires the calculation of $\boldsymbol{\gamma}^*(\mathbf{c})$ and its multiplication with the precomputed inverse matrix, which can usually be done in real-time.

5 Results and Discussion

This section explains the application of our geostatistical motion interpolation, and compares the accuracy and efficiency of prediction with the existing RBF interpolation techniques [Rose et al. 2001]. We employed no pseudo-examples, and used motion data sampled at 120 Hz with DOFs $\dim(\mathbf{q}) = 75$.

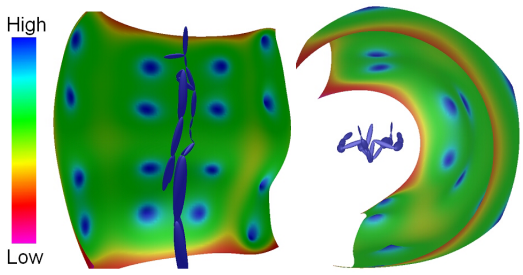
5.1 Performance comparison with RBF interpolation

We took 16 samples of various reaching motions for comparing the accuracy and efficiency of three methods: (1) RBF interpolation [Rose et al. 2001], (2) per-element interpolation proposed in Section 3.2, and (3) per-pose interpolation proposed in Section 3.4. Each sample was captured by asking a performer to initially take a standing pose and to finish each reaching movement when their right hand touched any constrained location indicated by the black spheres in Figure 5.

All motion samples were parameterized in the 2D control space $\mathbf{c} = \{c_1, c_2\} \in \mathfrak{C}$ with the turning angle c_1[deg] from the forward direction and the height c_2[cm] of a corresponding target position. The prediction accuracy of each method is evaluated by computing the maximum error, $MAX := \max_i [\|\mathbf{c}_i - \hat{\mathbf{c}}_i\|]$, and the root mean square error, $RMSE := \left(\sum_{i=1}^{S} \|\mathbf{c}_i - \hat{\mathbf{c}}_i\|^2 / S\right)^{1/2}$ within the parameters of $0 \leq c_1 \leq 220$, $0 \leq c_2 \leq 200$, where S represents the number of sampled locations, and $\hat{\mathbf{c}}_i$ denotes the control parameters corresponding to the final pose generated from $\mathbf{c}_i$, that is, $\hat{\mathbf{c}}_i$ represents the final angle and height of the right hand in the synthesized motion with $\mathbf{c}_i$. About five-hundred of control parameters, $S = 500$, are uniformly sampled for estimating prediction errors.

Table 1 shows the MAX and RMSE measured in the sampling region whose reliability, explained in the next section, exceeds given thresholds, and shows the average computation time per frame for runtime and preprocessing. This shows that our geostatistical interpolations have lower MAX and RMSE than RBF interpolations. The per-element method shows no significant improvement over the per-pose method, so the large cost in runtime and preprocessing for per-element is not justified. Figure 5 illustrates the estimated target hand positions in reaching motions, which shows the advantage of our geostatistical prediction in terms of smoothness. From these observations, we can conclude that per-pose interpolation, based on the kriging with pose distance metrics, is the best method for ensuring both accuracy and efficiency.

The above evaluation only considers the difference of control parameters; it only ensures the accuracy in the target position, not the accuracy of entire motions. For this reason, we experimented with



(a) side view (b) top view

Figure 8: Reliability map with statistical estimates of prediction error. The surface represents the reachable space of final positions of right hand, and blue and red areas indicate regions of high and low reliability, respectively. Target positions of reaching motion samples approximately coincide with the centers of blue areas.

using cross validation for evaluating the prediction accuracy of entire motions. After excluding a sample $\mathbf{M}_x$ from a learning dataset, we interpolate (or predict) a motion at $\mathbf{c}_x$, and compare the resulting motion with the original $\mathbf{M}_x$. Figure 6 shows the change of errors using the pose distance metric, and Figure 7 shows some snapshots comparing the sampled and synthesized motions. The results demonstrate the superiority of our geostatistical interpolations over RBF interpolations in predicting entire motions, especially during the last phase. The decrease of accuracy is noticeable at the peripheral target circled with green because of the lack of surrounding motion samples.

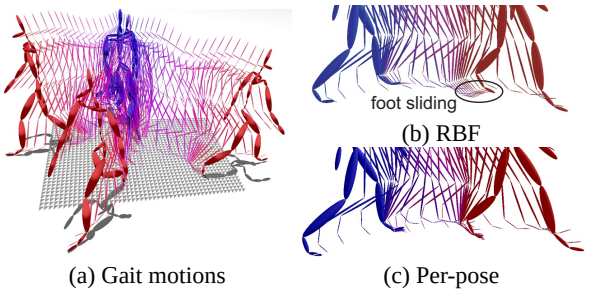
5.2 Reliability of prediction

Our statistical technique can compute the reliability $0 \leq \rho(\mathbf{c}) \leq 1$ of the predicted value (see Appendix B for the derivation). For example, Figure 8 shows a map with reliability of reachable space, indicated with colors. We experimentally confirmed that the amount of error $\|\mathbf{c} - \hat{\mathbf{c}}\|$ has a strong correlation with the reliability level. Pearson's product moment correlation coefficient becomes -0.70064 for the example in Figure 8, which proves that they have a strong negative correlation. This property enables the identification of highly reliable regions of the control space to ensure the plausibility of motions. Note that this reliability estimate is based on the same theoretical background as Gaussian processes [Grochow et al. 2004].

The reliability $\rho(\mathbf{c})$ can also be used as a quantitative indicator for reducing redundant samples. We employed a simple greedy algorithm for data reduction. It computes the reliability at each control parameter $\mathbf{c}_i$ by excluding the corresponding motion sample $\mathbf{M}_i$, and then removes the sample with the highest reliability, $\max_i [\rho(\mathbf{c}_i)]$. This procedure is iterated until the highest reliability falls below a given threshold τ. The number of gait motion samples used in Figure 9, for instance, could be reduced from 69 to 29 for $\tau = 0.85$ without introducing noticeable distortion.

5.3 Applications for interactive motion manipulation

Figure 9 shows the gait motions generated with the control parameters in a 2D space. Various curving gaits are automatically generated by giving only the relative 2D displacement of the root position after two strides. This shows that our method can generate plausible motions without using any corrective techniques such as inverse kinematics. This advantageous property is derived from the features of the kernel functions that are statistically optimized for each frame. Figure 10 shows a time-series of variogram variables for pose distance metrics in this example, which indicates the time-dependency of the correlation between motions.



(a) Gait motions (b) RBF (c) Per-pose

Figure 9: Interpolations of gait motions. Control parameters are given by the 2D displacement of a root node projected on the ground. The blue and red colored figures denote initial and final poses, respectively. The motions in (a) and (c) are generated with our per-pose interpolations, and the motion in (b) is generated with RBF interpolation. Foot-sliding is clearly detected in RBF interpolation but is negligible in our method.

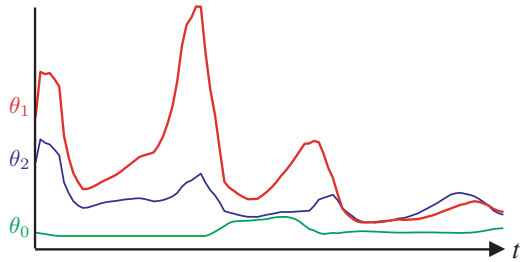


Figure 10: Time-dependency of predictive variogram variables for pose distance metrics in Figure 9.

Figure 11 demonstrates synthesized carrying motions with 4D control parameters given by the combination of start and end positions of the moved object. Our system successfully interpolates or extrapolates motions interactively even with such a high-dimensional control space. However, this example exhibits slight inconsistencies in foot positions due to an insufficient number of samples (only 13). In fact, this example requires the correction of the foot positions to avoid sliding; nevertheless the magnitude of errors is so small compared with existing motion interpolations.

Figure 12 shows controlling gait motions using the footprints of both feet as 4D control parameters. This example also demonstrates the accuracy of the foot positions, and separate control of these positions enhances the flexibility in fine control of gait styles. Figure 13 shows stair climing motions on irregularly-shaped stairs. The shape of each step is separately changed by giving height and width, and the parameters for successive pairs of steps form a 4D control space. These examples also generate skate-free gaits without the need for corrective calculations.

The above examples can be extended to synthesize motions from many footprints or steps by increasing the dimension of the control space. However, the increase of the dimension rapidly degrades the resulting motions due to insufficient samples. The practical solution is to blend the motions synthesized from all successive pairs. Strictly speaking, this blend sacrifices the reliability, but we experimentally confirmed that such ill effects are negligible. Figure 1 demonstrates the gaits on many steps generated by blending the second half period of the synthesized motion with the first half period of the following motion using ease-in/out functions.

Table 2 shows the number of samples, and the computational costs for runtime and preprocessing. This demonstrates that our

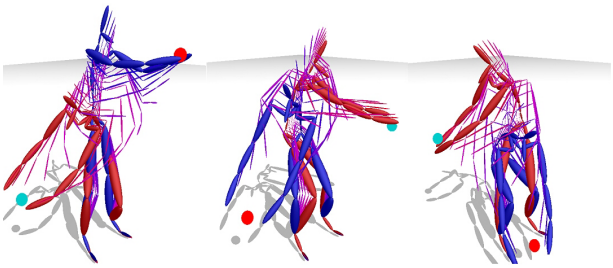


Figure 11: Carrying motions with two positional constraints. Each position consists of the turning angle around the vertical axis and the height from the ground.

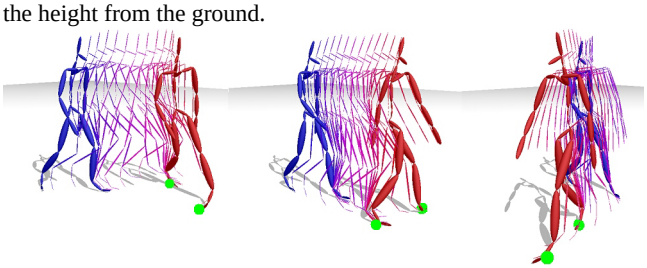


Figure 12: Gait controls with two footprints. Constrained 2D displacements are imposed on both right and left feet.

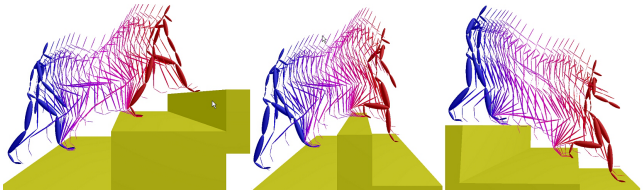


Figure 13: Gait controls with two step shapes. The heights and widths of two successive steps are given as control parameters.

interpolation can generate motions in real-time, and that the preprocessing time is tolerable for interactive motion editing. Notice that the computational cost of the runtime is roughly proportional to the number of samples.

6 Conclusions

We have proposed a novel method of motion interpolation based on geostatistics. The contributions of our method may be summarized as follows:

- Our interpolation predicts motions at given control parameters more accurately than RBF interpolation. This property can reduce the cost of post-cleaning for positional inconsistencies in end-effectors.
- The computational algorithm of our method employs no manual tuning parameters by relying on the theory of spatial statistics, which allows easy implementation.
- Our motion control based on spatial constraints is well-suited to not only interactive motion editing but also on-line motion manipulations, such as in video games, through the handling of various geometric targets.

Table 2: Number of samples and time spent in runtime and preprocessing (msec/frame) measured on a dual 3.0 GHz Xeon CPU.

Motion	Samples	Runtime	Preprocess
Figure 9	69	3.53	325.5
Figure 11	13	0.56	9.1
Figure 12	69	3.56	338.1
Figure 13	29	1.31	40.9

- The plausibility of the resulting motion can be quantitatively evaluated using estimates of prediction reliability, which allows adaptive reduction or population of motion samples.

The accuracy of prediction relies on a sufficient number of sampled motions. However, the number of necessary samples usually increases exponentially in the dimension of a control space. In particular, estimating an appropriate variogram function requires several samples distributed around the center of the control space. Nevertheless, the number of required samples is far less in our method than that required in simple linear interpolations. In addition, our method supplies a tool for adequately generating new samples through observation of a reliability map. Prediction accuracy also deeply depends on the choice of control parameters, and all parameters must be normalized so that the control space ensures intrinsic stationarity. Although this paper introduced a trend estimate for solving this problem, anisotropy analysis [Wackernagel 2003] is regarded as another possible solution.

Geostatistical interpolation could be utilized for retargeting motions by selecting bodily variables such as height and weight as control parameters. Conversion of image features to control parameters may enable motion prediction from 2D image sequences of human motions. Moreover, our numerical method may supply a useful tool for analyzing higher-level human motions that have corresponding spatial parameters; for example, the relational analysis between the position of obstacles and avoiding movements. Our future work includes the investigation of the validity of these applications.

Acknowledgements

We would like to thank Ken Anjyo for his valuable comments, William Baxter for proof-reading and video narration, Toyohisa Kaneko for supervising, and the first reviewer for theoretically deep comments. We also thank to Kazuhiko Mino and the staff of Links DigiWorks Inc. for the support of motion capturing. This work was supported by the 21st century COE program (Intelligent human sensing) and grants-in-aid for scientific research from Ministry of Education, Culture, Sports, Science and Technology.

References

Abe, Y., Liu, C. K., and Popović, Z. 2004. Momentum-based parameterization of dynamic character motion. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2004*, 173–182.

Alexa, M., and Muller, W. 2000. Representing animations by principal components. *Computer Graphics Forum 19*, 3, 411–418.

Arikan, O., Forsyth, D. A., and O'Brien, J. F. 2003. Motion synthesis from annotations. *ACM Transactions on Graphics 22*, 3, 402–408.

BRUDERLIN, A., AND WILLIAMS, L. 1995. Motion signal processing. In *Proc. of SIGGRAPH 95*, 97–104.

CRESSIE, N. A., AND HAWKINS, D. M. 1980. Robust estimation of the variogram. *Mathematical Geology 12*, 115–125.

CRESSIE, N. A. 1985. Fitting variogram models by weighted least squares. *Mathematical Geology 17*, 563–586.

CRESSIE, N. A. 1993. *Statistics for Spatial Data*. Wiley-Interscience.

DE JUAN, C., AND BODENHEIMER, B. 2004. Cartoon textures. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 267–276.

GROCHOW, K., MARTIN, S. L., HERTZMANN, A., AND POPOVIĆ, Z. 2004. Style-based inverse kinematics. *ACM Transactions on Graphics 23*, 3, 522–531.

HUIJBREGTS, C. J., AND MATHERON, G. 1971. Universal kriging. In *Proc. of International Symposium on Techniques for Decision-Making in Mineral Industry*, 159–169.

IKEMOTO, L., AND FORSYTH, D. A. 2004. Enriching a motion collection by transplanting limbs. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2004*, 99–108.

JOURNEL, A. G., AND HUIJBREGTS, C. J. 1978. *Mining Geostatistics*. Academic Press.

KOVAR, L., AND GLEICHER, M. 2002. Motion graphs. *ACM Transactions on Graphics 21*, 3, 473–482.

KOVAR, L., AND GLEICHER, M. 2003. Flexible automatic motion blending with registration curves. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 214–224.

KOVAR, L., AND GLEICHER, M. 2004. Automated extraction and parameterization of motions in large data sets. *ACM Transactions on Graphics 23*, 3, 559–568.

KOVAR, L., SCHREINER, J., AND GLEICHER, M. 2002. Footskate cleanup for motion capture editing. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 97–194.

LEE, J., CHAI, J., REITSMA, P. S. A., HODGINS, J. K., AND POLLARD, N. S. 2002. Interactive control of avatars animated with human motion data. *ACM Transaction on Graphics 21*, 3, 491–500.

PARK, S. I., SHIN, H. J., KIM, T. H., AND SHIN, S. Y. 2004. On-line motion blending for real-time locomotion generation. *Computer Animation and Virtual Worlds 15*, 3-4, 125–138.

ROSE, C., BODENHEIMER, B., AND COHEN, M. F. 1998. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications 18*, 5, 32–40.

ROSE, C. F., SLOAN, P.-P. J., AND COHEN, M. F. 2001. Artist directed inverse-kinematics using radial basis function interpolation. *Computer Graphics Forum 20*, 3, 239–250.

SAFONOVA, A., HODGINS, J. K., AND POLLARD, N. S. 2004. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Transactions on Graphics 23*, 3, 514–521.

SAKAMOTO, Y., AND KURIYAMA, S. 2004. Motion map: Image-based retrieval and segmentation of motion data. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 259–266.

TANCO, L. M., AND HILTON, A. 2000. Realistic synthesis of novel human movements from a database of motion capture examples. In *Proc. of Workshop on Human Motion*, 137–142.

UNUMA, M., ANJYO, K., AND TAKEUCHI, R. 1995. Fourier principles for emotion-based human figure animation. In *Proc. of SIGGRAPH 95*, 91–96.

WACKERNAGEL, H. 2003. *Multivariate Geostatistics*. Springer-Verlag.

WILEY, D. J., AND HAHN, J. K. 1997. Interpolation synthesis of articulated figure motion. *IEEE Computer Graphics and Applications 17*, 6, 39–45.

A. Cross validation for estimating variogram function

Cross validation is the simplest method for estimating a variogram function. This method is employed for finding the variable p that best explains the spatial variation of N samples r_i. After excluding a sample r_x from a dataset, we compute $r^*(\mathbf{c}_x)$ predicted from the remainder $r_{i\neq x}$ of the dataset. This process is repeated for each sample in order to compute the mean square error of prediction $MSE := \frac{1}{N}\sum_{x=1}^{N} |r_x - r^*(\mathbf{c}_x)|^2$. The predictive variogram function is then automatically determined so as to minimize MSE. In searching for the optimal p, the variables $\boldsymbol{\theta}$ are computed so as to satisfy the following minimizing criterion [Cressie 1985]:

$$\boldsymbol{\theta} = \arg\min_{\boldsymbol{\theta}} \left[\sum_{u=1}^{N_I} \frac{n(I_u)}{\gamma^*(h_u; p, \boldsymbol{\theta})^2} \left(\bar{\gamma}(h_u) - \gamma^*(h_u; p, \boldsymbol{\theta})\right)^2 \right] . \quad (4)$$

Our current implementation solves this nonlinear weighted least squares problem by using a public numerical computing library GSL that is available at http://www.gnu.org/software/gsl. Notice that the effect of $\boldsymbol{\theta}$ on the fitting is emphasized in the interval I_u of small h_u because the weights $n(I_u)/\gamma^*(h_u; p, \boldsymbol{\theta})^2$ monotonically decrease with respect to h_u for $n(I_u) \approx const$.

Since the variation of p produces very little effect on estimate accuracy compared with the effects of $\boldsymbol{\theta}$, Equation 4 is computed with constant step length $p = 0.0, 0.2, 0.4, \ldots, 2.0$, once at a selected frame. The most effective value of p is then selected among the discrete samples, and next the variables $\boldsymbol{\theta}$ are optimized with Equation 4 for all frames by fixing p at the selected value.

B. Reliability of prediction

The variance of the prediction error $\sigma^2(\mathbf{c})$ is estimated with the predictive variogram function of pose distance metrics as,

$$\sigma^2(\mathbf{c}) = \sum_{i=1}^{N} b_i \gamma^*(\|\mathbf{c} - \mathbf{c}_i\|; p, \boldsymbol{\theta}) .$$

This value vanishes at the sample location $\mathbf{c}_i$ because of the relations $b_i(\mathbf{c}_i) = 1$, $b_i(\mathbf{c}_{j\neq i}) = 0$ and $\sigma^2(\mathbf{c}_i) = \gamma^*(0; p, \boldsymbol{\theta}) = 0$. The estimate increases according to the distance from $\mathbf{c}_i$, and takes the maximum value of $\gamma^*(\infty; p, \boldsymbol{\theta}) = \theta_0 + \theta_1$ at the locations that have no spatial correlations with the sampled locations. We then define the reliability of prediction $\rho(\mathbf{c})$ as a normalization of $\sigma^2(\mathbf{c})$:

$$\rho(\mathbf{c}) = 1.0 - \sigma^2(\mathbf{c})/(\theta_0 + \theta_1)$$

where $\rho(\mathbf{c}_i) = 1.0$ and $\lim_{\|\mathbf{c}\|\to\infty} \rho(\mathbf{c}) = 0.0$ hold.