DOI: 10.1111/cgf.13089

COMPUTER GRAPHICS *forum*
Volume 36 (2017), number 8 pp. 418–428

Multi-Variate Gaussian-Based Inverse Kinematics

Jing Huang[1,*], Qi Wang[2,*], Marco Fratarcangeli[3], Ke Yan[4,†] and Catherine Pelachaud[1]

[1]CNRS/LTCI Telecom Paristec, Paris, France
[2]LIF, Ecole Centrale de Marseille, Marseille, France
[3]Chalmers University of Technology, Göteborg, Sweden
[4]College of Information Engineering, China Jiliang University, Hangzhou, China
keddiyan@gmail.com

Abstract

Inverse kinematics (IK) equations are usually solved through approximated linearizations or heuristics. These methods lead to character animations that are unnatural looking or unstable because they do not consider both the motion coherence and limits of human joints. In this paper, we present a method based on the formulation of multi-variate Gaussian distribution models (MGDMs), which precisely specify the soft joint constraints of a kinematic skeleton. Each distribution model is described by a covariance matrix and a mean vector representing both the joint limits and the coherence of motion of different limbs. The MGDMs are automatically learned from the motion capture data in a fast and unsupervised process. When the character is animated or posed, a Gaussian process synthesizes a new MGDM for each different vector of target positions, and the corresponding objective function is solved with Jacobian-based IK. This makes our method practical to use and easy to insert into pre-existing animation pipelines. Compared with previous works, our method is more stable and more precise, while also satisfying the anatomical constraints of human limbs. Our method leads to natural and realistic results without sacrificing real-time performance.

Keywords: animation, clustering, guassian process, inverse kinematics, Jacobian

ACM CCS: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism, Animation

1. Introduction

Inverse kinematics (IK) is widely used in robotics and computer animation for generating human poses from a set of constraints. In general, the problem is under-determined, meaning that the resulting animation is not unique. Several strategies have been proposed to solve such a problem, for example, body optimization by centre mass position [BMT96], dynamical system optimization [LHP05] and the minimization of kinematic energy algorithm [HS87]. In recent years, machine learning models have been employed to simulate human motions in applications such as motion synthesis and style representation [UGB*04, Law04, BH00, WB99]. All these works use statistical methods to describe the poses of human body. Different from geometric methods, statistical methods usually generate natural motions by considering the likelihood of certain poses according to either the body geometry (including collisions between body parts) or motion styles. In this paper, we present a multi-variate Gaussian-based IK (MGIK) approach. The poses are computed by minimizing an objective function and maximizing multiple Gaussian distribution density functions that are learned from a motion capture data set. Each Gaussian distribution density function describes a group of similar poses. Multi-variate Gaussian distribution models (MGDMs) are constructed from the degrees of freedom (DOFs) of the skeletal joints. A run-time MGDM is obtained automatically from multiple MGDMs and evaluated using the Gaussian process (GP) in real time. By embedding the Gaussian distribution into the objective function of the Jacobian IK, we obtain a natural solution that satisfies all constraints. The time complexity of the proposed algorithm is similar to the damped least squares (DLS) IK solution, and our method can be easily integrated into existing animation frameworks.

The main contribution of the proposed MGDM-based IK algorithm is the introduction of an objective function over a probability

*The major work about modelling and derivative of formula was done by Jing Huang and Qi Wang.
†Corresponding author.

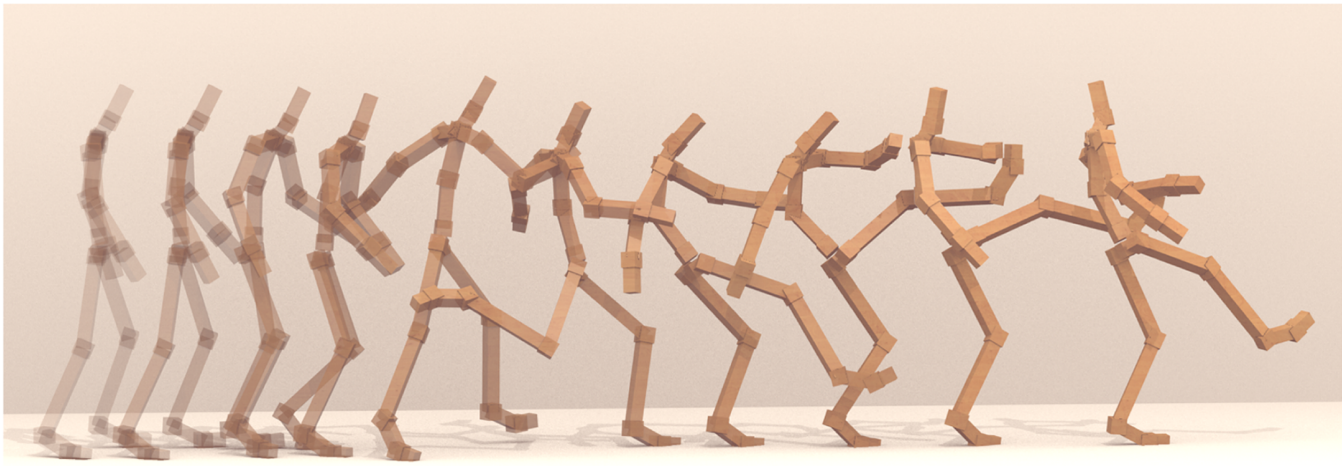

Figure 1: *Results of our MGIK.*

density function that describes coherent motion over DOFs in a local joint space. Our probability density function (PDF) is built upon MGDMs that are learned from a real data set. Our extended IK solver generates natural poses comparable to the motion capture method. The contributions introduced by our work are summarized as follows:

Accuracy: Our method perfectly solves the distance constraints between end-effectors and target positions while performing natural looking poses learned from real data.

Flexibility: Our method scales very well with the size of the learning data; and compared with other machine learning approaches, our method requires less data to narrow the precision of the constraints.

Performance: The resulting poses are generated in real time (1–3 ms per frame) which is faster that of other data-driven approaches (30–300 ms per frame) [GMHP04, WTR11, MLC10, WC11, HSCY13]. Our learning process is also very fast (less than 5 mins for 100 clusters of 30 000 poses using matlab [MAT14]).

Our solution is straightforward to implement, and can be easily integrated into existing animation pipelines by following the directions given in this paper.

2. Related Work

The objective of the basic IK is to find a chain configuration that satisfies the given constraints, which is a well-studied problem starting from 1980s [WE84, Wam86, NH86, Chi97, BB98, ZB94]. We also view the problem of the basic conventional IK as the process of reducing under-determined redundancies. However, in the basic IK, even if the objective function is solved and the resulting configuration satisfies all the given constraints, the displayed poses may not appear to be realistic and natural, especially for virtual character simulations. Many pieces of follow-up work have been conducted to improve the IK performance in terms of both quality and speed. In addition to the Gaussian-based IK solver, which is employed in this work, there are other IK methods, such as analytic methods [WSB78, TGB00, CCZB00], procedural numerical methods [Bai85, WC91, BK04, AL11] and example-based approaches [YKH04, KG04, RPpSC01].

Due to the complexity of the human model, these analytic solutions using multiple chain models [WSB78, TGB00, CCZB00] are much more easily implemented and thus often adopted in real-world applications (e.g. Alias Maya). Various extensions of the IK model design poses for body parts, such as the limbs and spine, for different morphologies [HRE*08, TGB00]. While the analytic solutions need to be modelled according to specific geometrical and morphological information, the procedural numerical methods can be seen as a general solution and can be adapted for various models. Certain procedural methods, such as heuristics [WC91, AL11], are often chosen due to their low computational overhead and flexibility in modelling. These methods define a custom rule that solves the targeting constraints iteratively. They are useful for robotic simulations or for simple constraint conditions, but they cannot be used to solve more specific problems such as modelling motion styles. Another group of procedural numerical solutions uses the Jacobian matrix. The basic conventional Jacobian solution has been extended to solve different constraint problems [BT92]. Some of the extensions [Bae01, Gle01] are well-designed editing tools for animation designers to specify sophisticated constraints on key poses. Liu *et al.* [LPL09] use independent feature subspace analysis (IFSA) to learn a low-dimensional motion style model. A constrained optimization is applied to the null space of the pseudo-inverse Jacobian solution. Thus, the posture can be edited in its parameter space.

Compared to these interactive editing tools, online applications usually require high computational speed and fewer constraints than offline applications, as handling these constraints requires more computational costs. The more straightforward way is to use motion capture data directly, which is called an example-based approach. One method [WP95] warps the sequence with motion parameter curves with defined key frames like constraints. The derived sequence is a smooth deformation on top of the original

key frame motion. Some other methods [AF02, KGP02, LCR*02, HG07, PP10] produce new sequences by interpolating or blending original sequences based on a certain similarity metric. Intuitively, all these methods focus on reusing motion data based on defined high-level constraints (e.g. reaching a target position or target key frame constraints), either concatenating or interpolating between selected sequences. They offer limited flexibility in creating new motion sequences; however, they cannot provide new specific poses in the sequence that are different from original data set. Similar to these motion generation methods, example-based IK systems have also been developed [YKH04, KG04, RPpSC01, FHKS12, FXS12]. These methods exploit the interpolation between example poses in the constraint space. Different interpolation methods are used in order to improve the precision of the constraints, such as Barycentric interpolation, the radial basis function (RBF)-based [RPpSC01] solution, the K-Nearest Neighbours (KNN) interpolation [KG04]. Unfortunately, these methods limit all referenced data using the same constraint type. The recording data should satisfy the specified constraint usage. The interpolation is not scalable given different dimension numbers of constraints. Moreover, as part of the example-based approaches, these methods can create poses only by interpolating or extrapolating referenced poses in the data; dissimilar poses cannot be synthesized. Thus, compared to procedural methods, they are less flexible.

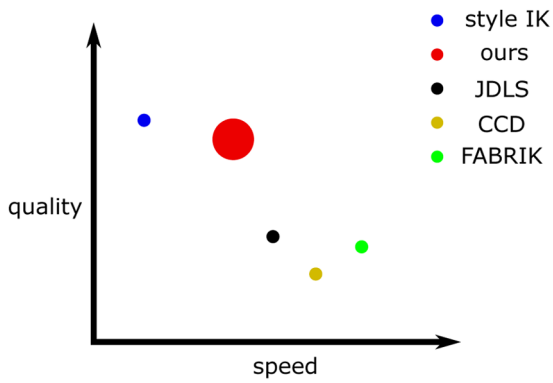

Figure 2: *The common comparison of different methods with the quality axis and the speed axis: style IK [GMHP04, WTR11] (24 ms per frame), JDLS [BK04] (0.6 ms per frame), CCD [WC91] (0.3 ms per frame) and FABRIK [AL11] (0.05 ms per frame). All performance records are obtained from indicated reference papers. The quality and speed of our algorithm vary depending on the chosen training data and the number of MGDM clusters.*

Recently, many machine learning approaches have been studied for the IK problem. Gaussian Process Latent Variable Models (GPLVMs) are some of the most successful models applied to motion capture data [GMHP04, Law04, WTR11]. Grochow *et al.* [GMHP04] model their style-based IK with a log domain PDF on the GPLVMs. Compared to a GP, GPLVMs can learn latent spaces from data sets during the training stage, which plays the role of a global non-linear dimensionality reduction technique. Learning latent spaces enables the GPLVMs to prevent over-fitting on small data sets. However, since the latent variable and the pose must be learned simultaneously in the synthesis process, GPLVMs are not as efficient computationally as the conventional IK methods, such as the Jacobian IK method. Especially, when the size of the training set is large, the dimensions of the kernel matrix in GPLVM will become too large, and the computational complexity increases drastically. Previous work is studied to address the problem of the high computational complexity of the GPLVM on a large database. Wu *et al.* [WTR11] use an adaptive clustering algorithm to select representative frames from a large motion-capture database, which is able to accelerate training and synthesize poses for the GPLVM. Lau *et al.* [LBJK09] use the Bayesian network model to capture the conditional independence of motion data, and a multi-variate probability distribution is used to model the variety of the generative model. Wei and Chai [WC11] use a mixture of factor analysers (MFAs) method to apply on a large data set; the prior information is used to optimize the poses. This method provides a maximum *a posteriori* (MAP) framework to deal with the interactive character-posing problem. The MAP framework guarantees that the generated posture is the most-likely posture that is similar to poses in the data set meeting the constraints. When optimizing the posteriori, Expectation Maximization is used to iteratively update the parameters. Although the computational complexity decreases comparing with that of the GPLVM, it still increases linearly with respect to the size of data set. Other works dealt with the unnaturalness of poses. Ho *et al.* [HSCY13] propose a Gauss Linking Integral model for preventing body penetration. Min *et al.* [MLC10] use multi-linear analysis techniques to construct a generative motion model for generating stylistic motion, including reaching the target motion. Some other works [TWC*09, LPLT11, HCMTH15] focus on compressing motion data by using data feature analysis to reduce the data's dimension, similar to IK minimization. These features can be spatial–temporal coherence or trajectory correlation. All the mentioned literature has shown that the machine learning approach has the ability to generate high-qualitative animations, but with less efficiency. The open challenge becomes to improve the speed performance of the computation.

Inspired by these methods, we develop our method as a conventional solution, while exploiting the PDF likelihood function with machine learning methods. Figure 2 illustrates a general comparison in terms of animation quality and efficiency base on our knowledge. Moreover, our solution can be easily integrated into conventional IK pipelines. We demonstrate that our solver can generate high visual quality natural poses with high real-time speed performance in the experimental result in Section 5.

3. Overview

Given a target trajectory for an end effector, the method of IK computes each posture frame of the corresponding end effector on or close to the target position. However, most geometric methods suffer from a loss of naturalness. In this paper, we propose a model called MGDM to address this problem. The main idea is to use a Gaussian probability distribution as a soft constraint and to combine the basic Jacobian IK to acquire a natural posture sequence for the skeletal character to reach the goal positions. The whole learning and synthesis process is outlined in Figure 3.

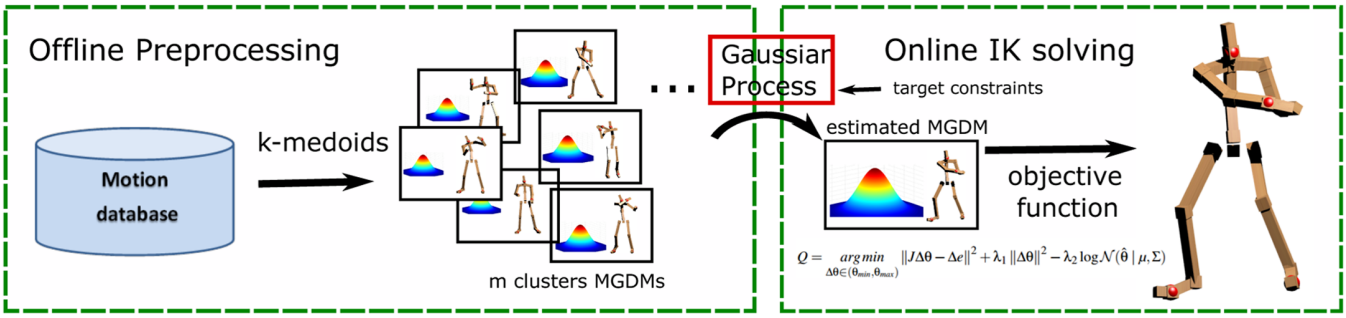

Figure 3: *The pipeline of our method.*

In data processing, we construct a pose data set $\{p_i\}$ using all the character poses extracted from the motion capture sequences. For each posture, we choose 60 joint DOFs as input features (excluding the root node). Though the Gaussian distribution model is capable to model different poses, it is impossible to model different poses in the data set using only one global distribution function. We choose the clustering method K-medoids to divide all the poses in terms of similarity into M clusters and construct M Gaussian distributions.

The number and the quality of the Gaussian distributions are extremely important for the final animation. If the Gaussian distributions are not proper or sufficient, the trajectory of the poses suffers from unnatural poses. If the distribution of these Gaussian functions is not dense enough, frames are skipped when doing a run-time selection from these Gaussian functions. To build a continuous space of Gaussian functions, we selected the GP method. A GP is defined as a stochastic process such that any finite sub-collection of random variables has a multi-variate Gaussian distribution. It predicts a new Gaussian distribution over the function space rather than directly searching through the M Gaussian distributions. The GP can be used to predict parameters by setting the positions of the target object as the inputs, which prevents the problem of skipping frames.

Our new objective function is derived by combining the basic Jacobian metric and the multi-variate Gaussian probability distribution (Figure 3).

4. Multi-Variate Gaussian-Based IK

The Gaussian distribution is embedded into the conventional Jacobian IK formula to bound the joint angle with both hard (joint DOF limits) and soft (joint DOF probability density distributions) constraints through a dynamically damped hyper-parameter λ. A new objective function is proposed, and an analytic solution of each iteration is derived. By optimizing the objective function, the desired posture trajectory is obtained with a coherence and naturalness for the skeletal character to reach its target position. The GP estimates the run-time Gaussian probability distribution with a relatively small set of examples. Since the Gaussian distribution is obtained from the GP according to the corresponding constrained joints, the estimated Gaussian distribution is more precise than when it is acquired directly from the data set. In addition, the interpolation made by the GP is suitable for various constraint dimensions, which means that new constraints can be added in the application without reprocessing the data.

4.1. The skeletal model

The virtual character is controlled by a skeletal model. We model the full body character with $n = 66$ DOFs including 3 DOFs for the translation vector $\mathbf{t}$ on the root joint and 63 DOFs for the rotation vector $\mathbf{r}$ in the whole skeleton. The DOF number is adjustable according to the data. The data set is loaded directly from the Bio-Vision format (BVH) animation files. The root joint is used for describing the global position (3 DOFs) and orientation (3 DOFs) in the global space (measured in metres). For a character pose, we consider a 66 DOFs vector, $\mathbf{x} = [\mathbf{t}, \mathbf{r}] = [t_1, t_2, t_3, r_1, r_2, \ldots, r_{63}]$. In the learning process, we omit the global position (t_1, t_2, t_3) and global orientation of the root joint (r_1, r_2, r_3) and keep only the joint angles $[r_3, r_4, \ldots, r_{63}]$ to explore the correlation of the DOFs in the local space.

4.2. Learning the multi-variate Gaussian distribution

The MGDM is a generalization of the uni-variate Gaussian distribution to higher dimensions. It describes the probability over the vector $\mathbf{x}$ with two parameters of μ and Σ which are the mean vector and covariance matrix, respectively (see Equation 1). Since the joint motions are dependent on each other, simple linearization usually fails to describe the correlation of all joint DOFs in the IK problem. μ and Σ are used to encode the motion correlation of the DOFs. The MGDM learns the correlation of the DOFs and constrains the generated poses of IK.

$$\mathbf{x} \sim \mathcal{N}_h(\boldsymbol{\mu}, \boldsymbol{\Sigma}), \tag{1}$$

where $\mathbf{x}$ is a h-dimensional random vector, $\mathbf{x} = [X_1, X_2, ???X_h]$, $\boldsymbol{\mu}$ is the h-dimensional mean vector with its value given by $\boldsymbol{\mu} = [\mathrm{E}[X_1], \mathrm{E}[X_2], \ldots, \mathrm{E}[X_h]]$, $\boldsymbol{\Sigma}$ is a $h \times h$ covariance matrix and $\boldsymbol{\Sigma} = [\mathrm{Cov}[X_i, X_j]], i = 1, 2, \ldots, h; j = 1, 2, \ldots, h$. We assume that all poses are sequentially sampled from a multi-variate Gaussian distribution, especially for similar poses. The probability of a given pose from such a multi-variate Gaussian distributed

data set can be computed using the normal distribution function as follows:

$$pdf(\mathbf{x}) = \frac{1}{\sqrt{(2\pi)^h |\mathbf{\Sigma}|}} \exp\left(-\frac{1}{2}(\mathbf{x}-\boldsymbol{\mu})^{\mathrm{T}} \mathbf{\Sigma}^{-1}(\mathbf{x}-\boldsymbol{\mu})\right), \quad (2)$$

where μ and Σ represent the mean vector and covariance matrix, respectively, and h is the number of dimensions of $x \in \mathbb{R}^h$.

The MGDM describes the variation of a chosen data set. However, one MGDM is inadequate to represent the whole data set on a suitable level. A posture is a high-dimensional random vector. The distribution of these vectors in the high-dimensional space can be sparse. Given one animation sequence, different poses may have large differences in terms of joint angles, which leads to great variation of DOFs. To solve this problem, we choose to simplify the data by clustering similar poses and computing the MGDM features separately in the pre-processing stage. Generally speaking, more clusters provide a more precise presentation of the data by the MGDMs. With respect to taking real poses for the mean vector, The Voronoi iteration K-medoids algorithm [PJ09] is employed to build m posture clusters in our system. This algorithm takes a real data sample (not the averaged mean value) as the centre of each cluster. The process is similar to K-means [JMF99], with the difference that the centre is assigned to the most centring point as evaluated by the Voronoi diagram.

After clustering, we compute the mean and the covariance matrix for each cluster. Suppose that we deal with the data of one group of n similar poses. The mean of the data set is computed by $\boldsymbol{\mu} = \sum \mathbf{x_i}/N$. The covariance matrix is defined as:

$$\Sigma_{sub}(i, j) = \{E[(x_i - \mu_i)(x_j - \mu_j)]\}, \quad (3)$$

where $\Sigma_{sub}(i, j)$ is an element of the matrix of Σ; i and j are the different indices of the DOFs. To make Σ invertible, a small noise β is added $\Sigma = \Sigma + \beta \times I$.

The final learned data structure is a list of MGDM parameters $(\boldsymbol{\mu}, \Sigma)$ that represent these clusters of similar poses. Since the MGDMs are computed by clustering methods, these MGDMs may not be well distributed. If two neighbouring trajectory (targeting) frames use two different MGDM clusters as their IK metric, the generated poses may have discontinuity problems (such as the skipping-frame problem). In contrast, the GP method is able to generate a continuous latent space for these MGDMs. To generate a continuous series of poses, the GP method generates the MGDM list and estimates the optimal MGDM for each target vector (constraints) during run time.

4.3. Online estimation of Gaussian distribution by the GP

Regarding regression in machine learning, we usually first assume what type of the objective function $f(\cdot)$ is to learn the optimal parameters. However, the function f cannot be defined by a simple Bayesian model because the prior assumption might be imprecise and affect the prediction result. The GP provides us a better way to marginalize over all possible choices for the function f. Both over-fitting and under-fitting are avoided to obtain a better prediction without any assumptions. The GP is an extension of the multivariate Gaussians from finite-sized to infinite-sized collections of real-valued variables [Ras06]. It can be thought of as a distribution over random functions [GMHP04]. Suppose that the input values $\mathbf{X} = (x_1, x_2, x_3, \ldots, x_m)$ are labelled by $\mathbf{Y} = (y_1, y_2, y_3, \ldots, y_m)$ in the training set. We wish to learn a mapping from the input x to output y: $y = f(x)$. From the definition of the GP, the collection of random variables $f(x) : x \in X$ is drawn from the GP and the probability of the set $(f(x_1), f(x_2), \ldots, f(x_m))$ is given by Equation (4).

$$\begin{bmatrix} f(x_1) \\ \vdots \\ f(x_m) \end{bmatrix} \sim \mathcal{N}(0,\ K), \quad (4)$$

where K is the covariance matrix and the entry $k(x_i, x_j)$ is defined by some specific kernel function.

In this work, we set the position of the constrained joints (multiple end effectors) as an input, which is a list of end effector targets: $T = t_1, t_2, \ldots t_i)$. The list T is scalable and configurable for different modelling objectives. We use the GP to predict the parameters $\{\mu, \Sigma\}$ of the Gaussian distribution, where μ is the mean and Σ is the covariant matrix. As a result, the GP usually provides a good prediction for the parameter set $\{\mu, \Sigma\}$.

During the learning phase, we have learned m clusters of MGDMs. In the synthesis phase, only one MGDM is needed, which represents best the goal posture. We assume that there exists a mapping between our target constraint (the end effector positional constraint) vector T and MGDM. The T_j is computed by updating the skeleton using its μ values. Generally, we require $T_1, T_2, \ldots, T_m$ to estimate the MGDM of current T_c. The m MGDMs corresponds m Gaussian functions, $f = (f_1, f_2, \ldots, f_m)^T$, with $\mathcal{N}(\mu, \Sigma) = (\mathcal{N}(\mu_1, \Sigma_1), \mathcal{N}(\mu_2, \Sigma_2), \ldots, \mathcal{N}(\mu_m, \Sigma_m))$. We define the estimated MGDM as f_*. The distribution of these functions is

$$\begin{bmatrix} f \\ f_* \end{bmatrix} \sim \mathcal{N}\left(0, \begin{bmatrix} k_{1,1} & \cdots & k_{1,m} & k_{1,*} \\ \vdots & \cdots & \vdots & \vdots \\ k_{m,1} & \cdots & k_{m,m} & k_{m,*} \\ k_{*,1} & \cdots & k_{*,m} & k_{*,*} \end{bmatrix}\right), \quad (5)$$

where $k_{i,j}$ is a kernel of the GP. To simplify the formula representation, the sub-matrix K_{sub} and $\mathbf{k}_*$ are defined as

$$K_{sub} = \begin{bmatrix} k_{1,1} & \cdots & k_{1,m} \\ \vdots & \ddots & \vdots \\ k_{m,1} & \cdots & k_{m,m} \end{bmatrix}, \quad (6)$$

$$\mathbf{k}_* = \begin{bmatrix} k_{1,*} \\ \vdots \\ k_{m,*} \end{bmatrix}. \quad (7)$$

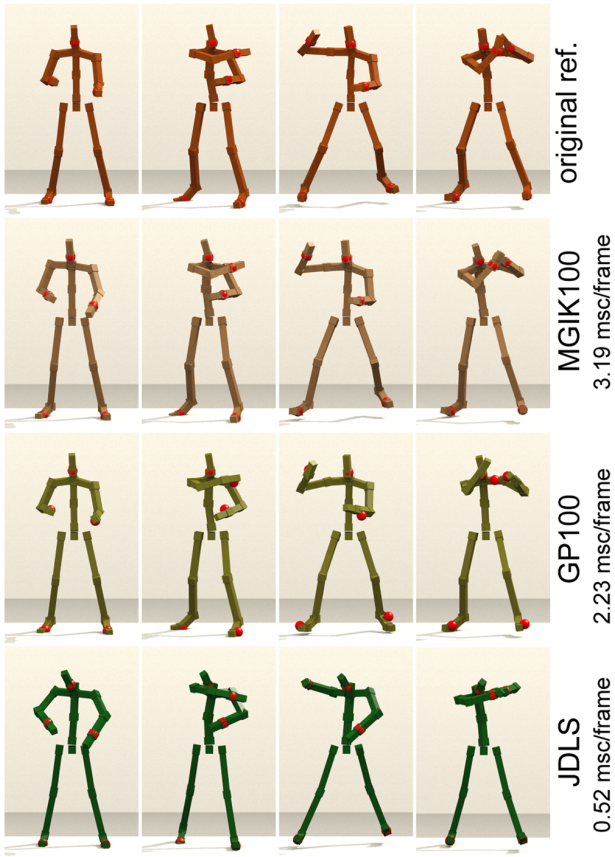

Figure 4: *The generated poses using four different solutions according to the given targets (red ball) in the tennis example, from top to bottom: original mocap data, our MGIK (100 cluster samples), GP (100 samples) [GMHP04] and JDLS (without constraints) [BK04].*

We use Laplacian kernel to compute each $k_{i,j}$ in the matrix:

$$k_{i,j} = \exp\left(-\frac{||T_i - T_j||}{\delta}\right), \tag{8}$$

where δ is a scale factor that describes how much the two points T_i and T_j can influence each other. The estimated MGDM is computed by

$$\mathcal{N}(\mu_*, \Sigma_*) = \mathbf{k}_*^\top K_{sub}^{-1} \mathcal{N}(\mu, \Sigma). \tag{9}$$

Given a vector of targets T_c in the local space of the root joint, the estimated MGDM is computed by the GP. We show the experimental results in Section 5.

4.4. Synthesis model

An estimated MGDM is computed using our GP method described in the previous section. It is used to complete our objective function for posture synthesis. In general IK applications, new poses are often generated using a Jacobian-based IK method [Bai85, BB04] by specifying the constraint hierarchy. Even though the constraints are solved, the generated poses may not look natural. The objective of our method is to find a posture configuration that maximizes a certain probability given a constraint vector. In our case, the probability is modelled by the MGDMs.

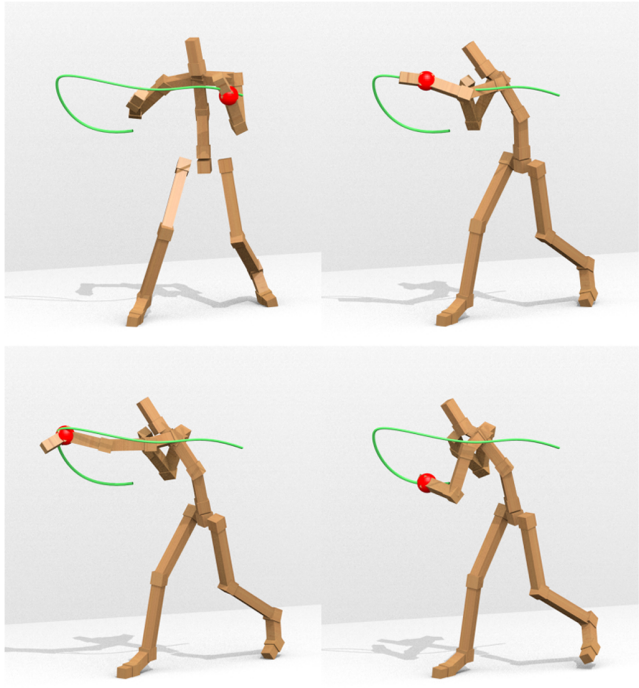

Figure 5: *The motion trajectory, which is used to generate or modify motion sequences.*

We assume that all the poses are sequentially sampled from our multi-variate Gaussian distribution (MGD). Our objective function is defined as

$$\begin{aligned} Q = \underset{\hat{\theta}\in(\theta_{min},\theta_{max})}{arg\,min} & (\|J\Delta\theta - \Delta e\|^2 + \lambda_1\|\Delta\theta\|^2 \\ & -\lambda_2 \log \mathcal{N}(\hat{\theta}|\mu, \Sigma)), \end{aligned} \tag{10}$$

where J is the Jacobian matrix, θ is the angle vector of DOFs, e is the vector of desired end effector positions. It is noted that $\hat{\theta} = \theta_0 + \Delta\theta$, where θ_0 is the original pose. The normal distribution is formulated as

$$\mathcal{N}(\hat{\theta} \mid \mu, \Sigma) = \frac{1}{(2\pi)^{N/2}\,|\Sigma|^{1/2}}\; e^{-\frac{1}{2}(\hat{\theta}-\mu)^\top \Sigma^{-1}(\hat{\theta}-\mu)}, \tag{11}$$

$$\log \mathcal{N}(\hat{\theta} \mid \mu, \Sigma) = \log\left[\frac{1}{(2\pi)^{N/2}\,|\Sigma|^{1/2}}\right] - \frac{1}{2}(\hat{\theta}-\mu)^\top \Sigma^{-1}(\hat{\theta}-\mu). \tag{12}$$

By decomposing the normal distribution function, our formula can be simplified to

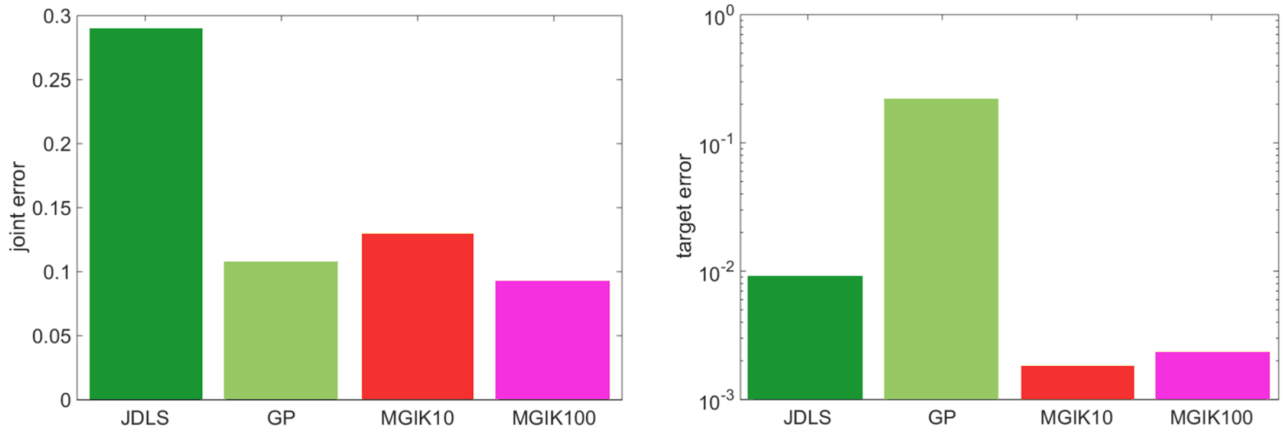

Figure 6: *The error metric (joint space MSE in radians and target distance MSE in metres) comparisons between our solution (MGIK) using 10 clusters and 100 clusters and other methods: JDLS [BK04] and GP [GMHP04].*

Table 1: *The performance comparison of our solution using 10 clusters and 100 clusters with other methods, that is, JDLS [BK04], SGPLVM [GMHP04, WTR11], GP [GMHP04] and Jacobian Transpose IK (JT) [AL09] with five target constraints. The threshold of the reaching targets is 0.01 m. The naturalness is judged by two simple subjective user evaluation standards: (1) physically possible and (2) natural looking.*

Methods	Average ms	Reaching targets	Posture
MGIK10	1.1	Yes	Natural
MGIK100	3.1	Yes	Natural
JDLS	0.55	Yes	Unnatural
SGPLVM	24.78	Yes	Natural
GP	2.31	No	Natural
JT	0.33	No	Unnatural

$$Q = \underset{\hat{\theta}\in(\theta_{min},\theta_{max})}{arg\,min} \left(\|J\Delta\theta - \Delta e\|^2 + \lambda_1 \|\Delta\theta\|^2 - \lambda_2 \left[C - \frac{1}{2}(\hat{\theta}-\mu)^\top \Sigma^{-1}(\hat{\theta}-\mu) \right] \right), \tag{13}$$

where $C = \log[\frac{1}{(2\pi)^{N/2}|\Sigma|^{1/2}}]$ is a constant. We can reformulate our objective function as

$$Q = \underset{\hat{\theta}\in(\theta_{min},\theta_{max})}{arg\,min} \left(\|J\Delta\theta - \Delta e\|^2 + \lambda_1 \|\Delta\theta\|^2 + \lambda_2 \left[\frac{1}{2}(\hat{\theta}-\mu)^\top \Sigma^{-1}(\hat{\theta}-\mu) \right] \right), \tag{14}$$

where $\hat{\theta} = \theta_0 + \Delta\theta$. We define three temporary variables, namely, A, B and C:

$$\begin{aligned} A &= \frac{\delta}{\delta\Delta\theta}\left[\left(\Delta\theta^\top J^\top - \Delta e^\top\right)(J\Delta\theta - \Delta e) \right] \\ &= \frac{\delta}{\delta\Delta\theta}\left[\Delta\theta^\top J^\top J\Delta\theta - \Delta\theta^\top J^\top \Delta e - \Delta e^\top J\Delta\theta + \Delta e^\top \Delta e \right] \\ &= 2J^\top J\Delta\theta - 2J^\top \Delta e, \end{aligned}$$

$$\begin{aligned} B &= \frac{\delta}{\delta\Delta\theta}\left[\lambda_1 \Delta\theta^2\right] = 2\lambda_1\Delta\theta, \\ C &= \frac{1}{2}\lambda_2 \frac{\delta}{\delta\Delta\theta}[(\hat{\theta}-\mu)^\top \Sigma^{-1}(\hat{\theta}-\mu)] \\ &= \frac{1}{2}\lambda_2 \frac{\delta}{\delta\Delta\theta}\left[\left(\theta_0^\top \Sigma^{-1} + \Delta\theta^\top \Sigma^{-1} - \mu^\top \Sigma^{-1}\right)(\theta_0 + \Delta\theta - \mu) \right] \\ &= \lambda_2 \left(\Sigma^{-1}\theta_0 + \Sigma^{-1}\Delta\theta - \Sigma^{-1}\mu \right), \end{aligned} \tag{15}$$

where Σ^{-1} is a symmetric matrix. The derivative of the function Q of $\Delta\theta$ can be evaluated as

$$\begin{aligned} \frac{\delta Q}{\delta\Delta\theta} &= A + B + C = 2J^\top J\Delta\theta - 2J^\top \Delta e + 2\lambda_1\Delta\theta \\ &\quad + \lambda_2 \left(\Sigma^{-1}\theta_0 + \Sigma^{-1}\Delta\theta - \Sigma^{-1}\mu \right). \end{aligned} \tag{16}$$

We set $\frac{\delta Q}{\delta\Delta\theta} = 0$ to minimize the objective function; and we solve $\Delta\theta$ by

$$\Delta\theta = \left(2J^\top J + 2\lambda_1 + \lambda_2\Sigma^{-1}\right)^{-1} \left(2J^\top \Delta e + \lambda_2\Sigma^{-1}\mu - \lambda_2\Sigma^{-1}\theta_0\right). \tag{17}$$

If the covariance is not taken into account, our function becomes the general DLS IK formulation [Bai85].

4.5. Optimization and objective balancing

Our system optimizes three aspects: first, the distance between the targets and end effectors: $\|J\Delta\theta - \Delta e\|$; second, $\Delta\theta$, which keeps the change of neighbouring frames as small as possible; and third, the probability of the DOFs configuration as well as the correlation between different DOFs: $\mathcal{N}(\hat{\theta} \mid \mu, \Sigma)$. It is crucial to set appropriate λ_1 and λ_2 to balance the variable optimizations. During each iteration, the distance value of $\|J\Delta\theta - \Delta e\|$ becomes smaller, while the whole objective function becomes more imbalanced. The reaching target may be unreachable if the second objective part becomes strongly dominant. To make our function more balanced through the whole iterations process, we weight our λ_2 with the

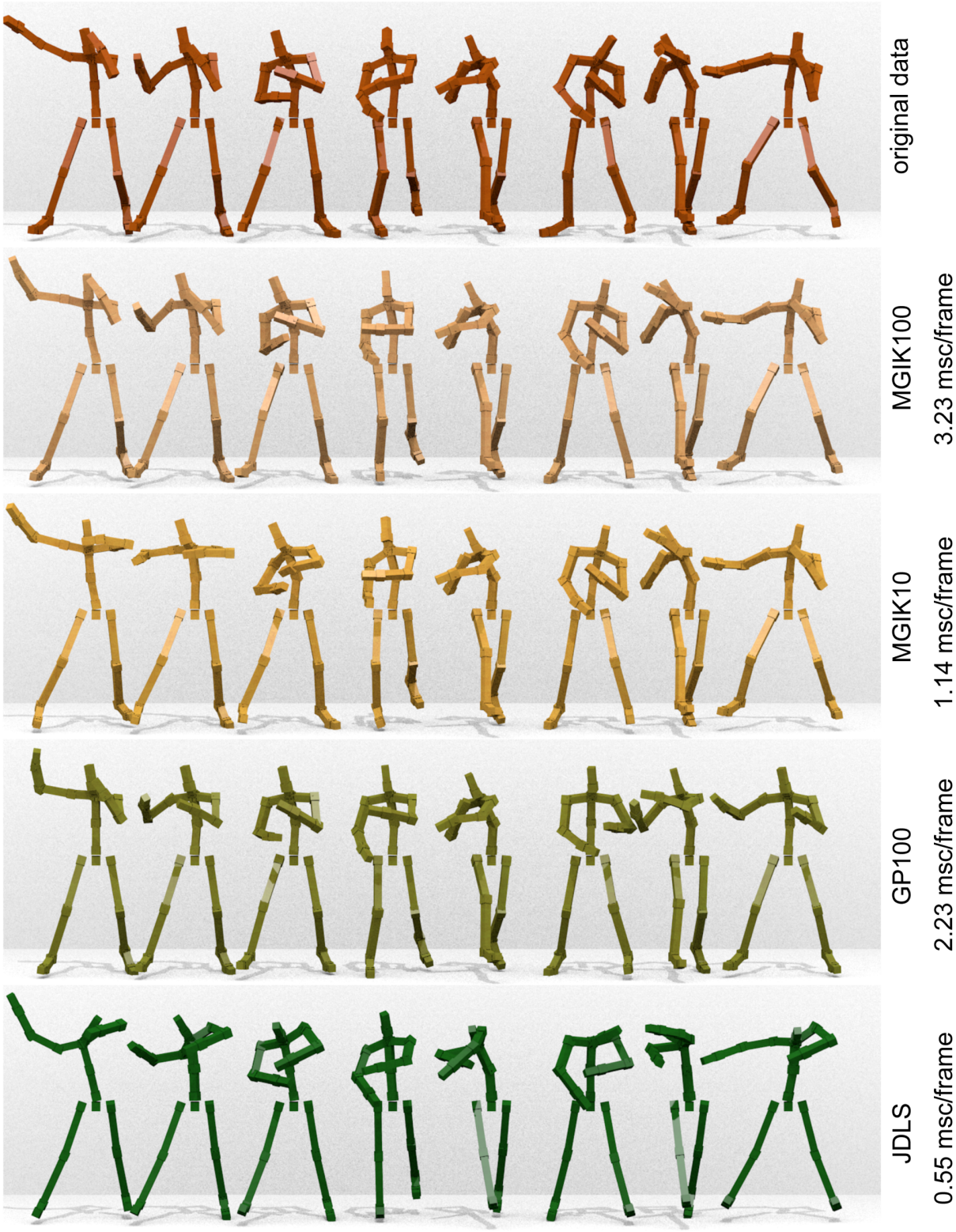

Figure 7: *The poses generated by different solutions according to the given targets in a boxing example, from top to bottom: original data, MGIK with 100 clusters, MGIK with 10 clusters, GP (100 samples) [GMHP04] and JDLS (without manually setup constraints) [BK04].*

distance between the targets and end effectors. For each iteration, $\Delta e_t = \|J_{t-1}\Delta\theta_{t-1} - \Delta e_{t-1}\|$. We set $\lambda_2 = \Delta e \times \lambda_o$, where λ_o is manually setup with a tunable damping factor. In our test, modifying the weight on the fly can speed up convergence compared to traditional approaches.

4.6. Hard and soft constraints

Our solution balances between hard and soft constraints for joint DOF limits. This prevents the problem from having any feasible solution with classic machine learning approaches [GMHP04]. Our method applies a basic hard joint limit solution to bound $\hat{\theta}$ for each DOF by its minimum and maximum joint limits: θ_{min} and θ_{max}. The values of θ_{min} and θ_{max} can either be defined manually by artists, or learned by iterating through all sequential poses to record the minimum and maximum values for each DOF. When λ_o and λ_2 are too small, the variation range of each joint built upon the probability function may become too large. To prevent the solved poses from violating the bio-mechanical properties and physical capacity of human, the hard constraints are needed to limit the joint rotations. According to our experience, by setting up appropriate λ_o, the hard joint constraints are rarely triggered. Therefore, our MGDM approach is a high-quality soft joint constraint model. Other constraint models [Bae01, Chi97] should also work by integrating our solver.

5. Results

Several existing methods are compared with the proposed MGIK solution, such as Jacobian Damped Least Squares (JDLS) and the GP sing the regression model, which is equivalent to the RBF solution without training and optimizing the style-based IK [GMHP04]. All methods consider only end effector trajectories without explicitly specifying other constraint parameters. As shown in Figure 4, JDLS generates unnatural poses if the constraints are not well defined; the GP generates natural poses from a large data set with targeting errors when the data are not dense enough. In contrast, in our solution, the joint limits are internally computed according to the Gaussian distribution and automatically learned from the data set. It solves the targeting problem and produces natural poses. The proposed MGIK solution can be used for various applications with high quality virtual character simulation, such as posturing, motion sequence editing (Figure 5) and motion reconstruction.

In the experiment, MGDMs are built using a 'sport' data set of 10 different sequences of 66 DOFs. This data set is composed of different sports, such as tennis, golf and shooting. Each sequence has more than 3500 frames of poses. MGDMs with both 10 and 100 clusters are generated in the preprocessing step for two tests. GP kernels with dimensions 10×10 and 100×100 are used in the evaluation step for our solution. Given the trajectories of the end effectors, we use the MGDM to regenerate each sequence of motion. The test is performed on an Intel Xeon CPU 2.93 GHz with one processing unit. A performance comparison of solving one posture using various methods is shown in Table 1, and the speed is measured in milliseconds (ms) per frame. In general, our solution performs less efficiently compared to the traditional IK solutions due to the run-time MGDM construction using the GP. However, it is stable and able to generate natural poses with multiple constraints. It is also much faster than other machine learning techniques such as [GMHP04, Law04, WTR11]. Considering that both the Jacobian solutions and matrix operations are parallelizable [HMCB16], the speed of our MGIK algorithm can be further improved by using multi-thread programming. The error metric comparisons are shown in Figure 6. We generate different motion sequences given five trajectories for the head, left wrist, right wrist, left foot and right foot. We measure the average errors of methods for all examples. The mean squared error (MSE) (averaged by all 66 DOFs) in the joint space (joint MSE in radian) and the MSE of the distance (in metres) between the end effectors and the targets (averaged by 5 end-effectors) are also computed. A small difference in the joint space error metric may result in a large variation for a specific joint. The density of the MGDM distribution in GP influences the resulting animation quality. Our MGIK solution offers high-quality natural looking poses and has much better results compared to traditional methods. In Figure 7, we show a boxing animation example generated by different approaches, including: MGIK with 100 clusters, MGIK with 10 clusters, GP (100 samples) [GMHP04] and JDLS (without manually setup constraints) [BK04]. For the speed performance, the most time-consuming step is the run-time MGDM estimation using a high-dimensional kernel. It is noted that by reducing the size of the GP kernel, the performance can be improved significantly.

6. Conclusions

In this paper, we propose a novel extended IK method using a multivariate Gaussian distribution as the internal constraints. A run-time MGDM is estimated from m pre-processed MGDMs (m clusters of similar postures) by a GP given a constraint target vector. Our MGIK algorithm offers several advantages over the conventional methods for natural posture synthesis, such as internally defining joint constraints, generating coherence motions in the local joint space by a covariance matrix and guaranteeing that the target constraints are reached. The solution is an extension of the conventional Jacobian solution and can be easily integrated into traditional animation pipelines.

For future work, we plan to apply a dimension reduction algorithm to simplify the MGDM. We also intend to build a level of details (LOD) structure for the computation of a skeletal model to further save computational time. As an interesting note, the Gaussian mixture model might fit our algorithm well for solving motion style transfer problems [XWCH15].

Acknowledgments

This work was partially supported by the National Science Foundation of China (No. 61602431) and the foundation of talents start-up project of China Jiliang University (No. 000485). We thank China Scholarship Council for supporting the thesis of the author Qi Wang.

References

[AF02] Arikan O., Forsyth D. A.: Interactive motion generation from examples. *ACM Transactions on Graphics (TOG) 21* (2002), 483–490.

[AL09] Aristidou A., Lasenby J.: *Inverse Kinematics: A Review of Existing Techniques and Introduction of a New Fast Iterative Solver*. Tech. Rep. CUEDF-INFENG, TR-632, Department of Information Engineering, University of Cambridge, September 2009.

[AL11] Aristidou A., Lasenby J.: Fabrik: A fast, iterative solver for the inverse kinematics problem. *Graphical Models 73*, 5 (Sept. 2011), 243–260.

[Bae01] Baerlocher P.: *Inverse Kinematics Techniques for the Interactive Posture Control of Articulated Figures*. PhD thesis, Ecole Polytechnique Federale de Lausanne, 2001.

[Bai85] Baillieul J.: Kinematic programming alternatives for redundant manipulators. In *1985 IEEE International Conference on Robotics and Automation. Proceedings*. (St. Louis, Missouri, USA, March 1985), vol. *2*, pp. 722–728.

[BB98] Baerlocher P., Boulic R.: Task-priority formulations for the kinematic control of highly redundant articulated structures. In *Proceedings, 1998 IEEE/RSJ International Conference on Intelligent Robots and Systems, 1998*, (Victoria, BC, Canada, Oct. 1998), vol. 1, pp. 323–329.

[BB04] Baerlocher P., Boulic R.: An inverse kinematics architecture enforcing an arbitrary number of strict priority levels. *The Visual Computer 20*, *6* (2004), 402–417.

[BH00] Brand M., Hertzmann A.: Style machines. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (New Orleans, LA, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 183–192.

[BK04] Buss S. R., Kim J.-S.: Selectively damped least squares for inverse kinematics[J]. *Journal of graphics, gpu, and game tools*, *10*, 3 (2005), 37–49.

[BMT96] Boulic R., Mas R., Thalmann D.: A robust approach for the control of the center of mass with inverse kinetics. *Computers & Graphics 20*, *5* (1996), 693–701.

[BT92] Boulic R., Thalmann D.: Combined direct and inverse kinematic control for articulated figure motion editing. In *Computer Graphics Forum* (1992), vol. 11, Wiley Online Library, pp. 189–202.

[CCZB00] Chi D., Costa M., Zhao L., Badler N.: The emote model for effort and shape. In *Proceedings of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (New Orleans, LA, USA, 2000), ACM Press/Addison-Wesley Publishing Co., pp. 173–182.

[Chi97] Chiaverini S.: Singularity-robust task-priority redundancy resolution for real-time kinematic control of robot manipulators. *IEEE Transactions on Robotics and Automation 13*, *3* (1997), 398–410.

[FHKS12] Feng A. W., Huang Y., Kallmann M., Shapiro A.: An analysis of motion blending techniques. In *International Conference on Motion in Games* (Rennes, France, Nov. 2012).

[FXS12] Feng A. W., Xu Y., Shapiro A.: An example-based motion synthesis technique for locomotion and object manipulation. In *Proceedings of the ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games*, Stephen N. Spencer (Eds.), (Costa Mesa, CA, USA, 2012), ACM, pp. 95–102.

[Gle01] Gleicher M.: Comparing constraint-based motion editing methods. *Graphical Models*, *63*, 2 (2001), 107–134.

[GMHP04] Grochow K., Martin S. L., Hertzmann A., Popović Z.: Style-based inverse kinematics. In *ACM SIGGRAPH 2004 Papers*, Joe Marks (Ed.), (New York, NY, USA, 2004), SIGGRAPH '04, ACM, pp. 522–531.

[HCMTH15] Hou J., Chau L.-P., Magnenat-Thalmann N., He Y.: Human motion capture data tailored transform coding. *IEEE Transactions on Visualization and Computer Graphics*, *21*, 7 (2015), 848–859.

[HG07] Heck R., Gleicher M.: Parametric motion graphs. In *Proceedings of the 2007 Symposium on Interactive 3D Graphics and Games* (Seattle, WA, USA, 2007), ACM, pp. 129–136.

[HMCB16] Harish P., Mahmudi M., Callennec B. L., Boulic R.: Parallel inverse kinematics for multithreaded architectures. *ACM Transactions on Graphics 35*, 2 (Feb. 2016), 19:1–19:13.

[HRE*08] Hecker C., Raabe B., Enslow R. W., DeWeese J., Maynard J., van Prooijen, K.: Real-time motion retargeting to highly varied user-created morphologies. *ACM Transactions on Graphics 27*, 3 (Aug. 2008), 27:1–27:11.

[HS87] Hollerbach J. M., Suh K. C.: Redundancy resolution of manipulators through torque optimization. *IEEE Journal of Robotics and Automation*, *3*, 4 (1987), 308–316.

[HSCY13] Ho E. S., Shum H. P., Cheung Y.-m., Yuen P. C.: Topology aware data-driven inverse kinematics. In *Computer Graphics Forum* (2013), vol. 32, Wiley Online Library, pp. 61–70.

[JMF99] Jain A. K., Murty M. N., Flynn P. J.: Data clustering: A review. *ACM Computing Surveys (CSUR)*, *31*, 3 (1999), 264–323.

[KG04] Kovar L., Gleicher M.: Automated extraction and parameterization of motions in large data sets. *ACM Transactions on Graphics 23*, 3 (Aug. 2004), 559–568.

[KGP02] Kovar L., Gleicher M., Pighin F.: Motion graphs. *ACM Transactions on Graphics (TOG)*, *21* (2002), 473–482.

[Law04] Lawrence N. D.: Gaussian process latent variable models for visualisation of high dimensional data. *Advances in Neural Information Processing Systems*, *16*, 3 (2004), 329–336.

[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.

[LCR*02] Lee J., Chai J., Reitsma P. S., Hodgins J. K., Pollard N. S.: Interactive control of avatars animated with human motion data. *ACM Transactions on Graphics (TOG)*, *21* (2002), 491–500.

[LHP05] Liu C. K., Hertzmann A., Popović Z.: Learning physics-based motion style with nonlinear inverse optimization. *ACM Transactions on Graphics 24*, 3 (July 2005), 1071–1081.

[LPL09] Liu G., Pan Z., Li L.: Motion synthesis using style-editable inverse kinematics. In *Intelligent Virtual Agents* (Amsterdam, Netherlands, 2009), Springer, pp. 118–124.

[LPLT11] Lin I.-C., Peng J.-Y., Lin C.-C., Tsai M.-H.: Adaptive motion data representation with repeated motion analysis. *IEEE Transactions on Visualization and Computer Graphics*, *17*, 4 (2011), 527–538.

[MAT14] MATLAB,: *Version 8.4.0 (R2014b)*. The MathWorks Inc., Natick, MA, 2014.

[MLC10] Min J., Liu H., Chai J.: Synthesis and editing of personalized stylistic human motion. In *Proceedings of the 2010 ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2010), I3D '10, ACM, pp. 39–46.

[NH86] Nakamura Y., Hanafusa H.: Inverse kinematic solutions with singularity robustness for robot manipulator control. *Journal of Dynamic Systems, Measurement, and Control*, *108*, 3 (1986), 163–171.

[PJ09] Park H.-S., Jun C.-H.: A simple and fast algorithm for k-medoids clustering. *Expert Systems with Applications*, *36*, 2 (2009), 3336–3341.

[PP10] Pejsa T., Pandzic I. S.: State of the art in example-based motion synthesis for virtual characters in interactive applications. *Computer Graphics Forum*, *28* (2010), 202–226.

[Ras06] Rasmussen C. E.: *Gaussian Processes for Machine Learning*, 2006, MIT Press. URL: http://citeseerx.ist.psu.edu/viewdoc/summary?doi:10.1.1.86.3414.

[RPpSC01] Rose C. F., Peter-pike I., Sloan J., Cohen M. F.: Artist-directed inverse-kinematics using radial basis function interpolation. *Computer Graphics Forum*, *20* (2001), 239–250.

[TGB00] Tolani D., Goswami A., Badler N. I.: Real-time inverse kinematics techniques for anthropomorphic limbs. *Graphical Models*, *62*, 5 (2000), 353–388.

[TWC*09] Tournier M., Wu X., Courty N., Arnaud E., Revéret L.: Motion compression using principal geodesics analysis. In *Computer Graphics Forum* (2009), vol. 28, Wiley Online Library, pp. 355–364.

[UGB*04] Urtasun R., Glardon P., Boulic R., Thalmann D., Fua P.: Style-based motion synthesis. In *Computer Graphics Forum* (2004), vol. 23, Wiley Online Library, pp. 799–812.

[Wam86] Wampler II C. W.: Manipulator inverse kinematic solutions based on vector formulations and damped least-squares methods. *IEEE Transactions on Systems, Man, and Cybernetics 16*, 1 (Jan. 1986), 93–101.

[WB99] Wilson A. D., Bobick A. F.: Parametric hidden Markov models for gesture recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *21*, 9 (1999), 884–900.

[WC91] Wang L.-C. T., Chen C. C.: A combined optimization method for solving the inverse kinematics problems of mechanical manipulators. *IEEE Transactions on Robotics and Automation*, *7*, 4 (1991), 489–499.

[WC11] Wei X. K., Chai J.: Intuitive interactive human-character posing with millions of example poses. *Computer Graphics and Applications, IEEE*, *31*, 4 (2011), 78–88.

[WE84] Wolovich W., Elliott H.: A computational technique for inverse kinematics. In *The 23rd IEEE Conference on Decision and Control* (Las Vegas, Nevada, USA, 1984), no. 23, pp. 1359–1363.

[WP95] Witkin A., Popovic Z.: Motion warping. In *Proceedings of the 22nd Annual Conference on Computer Graphics and Interactive Techniques*, Susan G. Mair, Robert Cook (Eds.), (Los Angeles, CA, USA, 1995), ACM, pp. 105–108.

[WSB78] Weber L., Smoliar S. W., Badler N. I.: An architecture for the simulation of human movement. In *Proceedings of the 1978 Annual Conference—Volume 2* (New York, NY, USA, 1978), ACM '78, ACM, pp. 737–745.

[WTR11] Wu X., Tournier M., Reveret L.: Natural character posing from a large motion database. *Computer Graphics and Applications, IEEE*, *31*, 3 (2011), 69–77.

[XWCH15] Xia S., Wang C., Chai J., Hodgins J.: Realtime style transfer for unlabeled heterogeneous human motion. *ACM Transactions on Graphics 34*, 4 (July 2015), 119:1–119:10.

[YKH04] Yamane K., Kuffner J. J., Hodgins J. K.: Synthesizing animations of human manipulation tasks. In *ACM SIGGRAPH 2004 Papers* (New York, NY, USA, 2004), SIGGRAPH '04, ACM, pp. 532–539.

[ZB94] Zhao J., Badler N. I.: Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Transactions on Graphics*, *13* (October 1994), 313–336.

Supporting Information

Additional Supporting Information may be found in the online version of this article at the publisher's web site:

Video S1.

Video S2.

Video S3.

Video S4.

Video S5.

Video S6.

Video S7.

Video S8.