DOI: 10.1111/cgf.13310

COMPUTER GRAPHICS *forum*
Volume 00 (2017), number 00 pp. 1–24

Inverse Kinematics Techniques in Computer Graphics: A Survey

A. Aristidou[1,2], J. Lasenby[3], Y. Chrysanthou[1] and A. Shamir[2]

[1]Department of Computer Science, University of Cyprus, Cyprus
a.aristidou@ieee.org, yiorgos@cs.ucy.ac.cy
[2]The Interdisciplinary Center, Herzliya, Israel
arik.shamir@gmail.com
[3]Department of Engineering, University of Cambridge, United Kingdom
jl221@cam.ac.uk

Abstract

Inverse kinematics (IK) is the use of kinematic equations to determine the joint parameters of a manipulator so that the end effector moves to a desired position; IK can be applied in many areas, including robotics, engineering, computer graphics and video games. In this survey, we present a comprehensive review of the IK problem and the solutions developed over the years from the computer graphics point of view. The paper starts with the definition of forward and IK, their mathematical formulations and explains how to distinguish the unsolvable cases, indicating when a solution is available. The IK literature in this report is divided into four main categories: the analytical*, the* numerical*, the* data-driven *and the* hybrid *methods. A timeline illustrating key methods is presented, explaining how the IK approaches have progressed over the years. The most popular IK methods are discussed with regard to their performance, computational cost and the smoothness of their resulting postures, while we suggest which IK family of solvers is best suited for particular problems. Finally, we indicate the limitations of the current IK methodologies and propose future research directions.*

Keywords: inverse kinematics, motion capture, biomechanical constraints

ACM CCS: General and reference—Surveys and overviews; Computing methodologies—Animation

1. Introduction

Kinematics describes the rotational and translational motion of points, bodies (objects) and systems of bodies (groups of objects) without consideration of what causes the motion or any reference to mass, force or torque. Inverse kinematics (IK) was initiated in robotics as the problem of moving a redundant kinematic arm with specific degrees of freedom (DoFs) to a pre-defined target. Beyond its use in robotics, IK has found applications in computer graphics, generating particular interest in the field of animating articulated subjects. This survey focuses on IK applications in computer graphics, aiming to provide insights about IK to young researchers by introducing the mathematical problem, and surveying the most popular techniques that tackle the problem.

Computer graphics applications usually deal with articulated figures, which are convenient models for humans, animals or other legged virtual creatures from films and video games. Animating such articulated characters is a challenging problem. Most virtual character models are complicated, made up of many joints, thus having a high number of DoFs. In addition, they are required to satisfy numerous constraints, including joint and/or contact restrictions. One way to handle this complexity is to manually adjust all the DoFs by carefully modifying the joint rotations to achieve the desired pose and ensure their temporal coherence—an extremely complex and time-consuming process.

Therefore, it was a necessity to find efficient ways to manipulate systems consisting of complex and multi-link models. IK has become one of the fundamental techniques for editing motion data. IK is commonly used for animating articulated figures using only the desired positions (and sometimes the orientations) of certain joints, commonly referred to as end effectors (e.g. usually end effectors are control points, and can be either end joints, such as feet and hands, or inner joints, such as the elbow and knee). The end effector positions are usually specified by the animator or a motion capture system, and must reach the desired positions in order to accomplish the given task. The remaining DoFs of the articulated model are automatically determined according to different criteria that depend on the employed IK solver and the model constraints. This can save a lot of work for the animator, while still maintaining fine control and temporal coherence.

The end effector positions can be modelled as a function of the DoFs, leading to a formal definition of the IK problem as finding $\boldsymbol{\theta}$ in $\mathbf{s} = f(\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ is the column vector giving the DoFs, and $\mathbf{s}$ is a vector that gathers all the desired end effector positions. This problem is highly under-constrained as $\boldsymbol{\theta}$ usually has a much larger dimension than $\mathbf{s}$. In addition, it is a non-linear problem as f involves complex combinations of trigonometric functions. The efficiency and effectiveness of the IK solver is generally measured in terms of the smoothness of the produced motion,[1] its scalability and the computational cost required for the resulting pose. The scalability trades computational time with the ability to address increasingly difficult constraints or kinematic chains with a large number of joints and DoFs.

The IK problem has attracted the attention of scholars for many years. There are many tasks in applications of virtual humanoids that need realistic postures and motions, including character re-targeting [Gle98, MBBT00, HRE*08], skeleton control [SLSG01, ACL16], solving foot skating [KSG02, MK11] and ergonomic evaluation [Wan99]. Further investigation is needed to advance the current state-of-the-art IK solvers, including the improvement of their computational performance, avoidance of deadlock situations or singularity problems, production of smooth transitions without oscillations, support of anthropometric and contact constraints, as well as enhancements to applications, such as obstacle avoidance, style modification, motion re-targeting, motion control, etc.

This report presents the most popular techniques for solving the IK problem in an interactive and/or intuitive fashion for the design, control and manipulation of articulated figures. Its purpose is to illustrate the evolution of IK in computer graphics, where the research has been focused in the past, and how it has progressed over the years. Over the past decade, there have been a few papers surveying IK; Boulic and Mas [BM96] studied motion control and reviewed the properties and limitations of IK compared to other techniques, while later Boulic and Kulpa [BK07] presented a tutorial in Eurographics 2007 on IK for virtual humanoids, focusing on pseudo-inverse methods with priorities and their hybrid method. Some other surveys focus on more specific aspects of IK, such as Colomè and Torras [CT12] work on redundant IK solvers, Buss's [Bus09] analytical review on Jacobian-based solvers and Henrich *et al.* [HKW97] survey on parallel computation of robot kinematics.

We present a comprehensive review of IK solvers, which goes beyond the previous surveys, including recent solvers and new trends. The solvers presented here are divided into four main categories: the *Analytic* family (Section 4), the *Numerical* family (Section 5), *Data-driven* methods (Section 6) and the *Hybrid* family of solvers (Section 7). It also highlights the advantages and disadvantages of each family of methods with regard to convergence, singularity handling, support of joint constraints, the capability of reaching multiple tasks, computational cost, scalability and smoothness of motion. In addition, it provides indications on which IK solvers are the best for solving different goals/problems, while directions for future work and applications are provided, giving insights to where effort should be placed to advance the current solvers.

2. The Articulated Body Model

A rigid multi-body system consists of a set of rigid objects, called *links*, connected together by *joints*. Most models assume that body parts are rigid, although this is just an assumption approximating reality. A *joint* is the component concerned with motion; it permits some degree of relative motion between the connected links. The skeletal structure is usually modelled as a hierarchy of links connected by joints, each defined by their length, shape, volume and mass properties. A *posture* is defined as the skeletal configuration of a figure. A realistic posture must satisfy a set of criteria. First, the joints of all character models have natural articulation limits. Second, inter-penetration of body parts with themselves or with other objects is not permitted. In addition, physical laws should be considered as external factors. Virtual body modelling is important for posture control; a well-constrained model can restrict postures to a feasible set, therefore allowing a more realistic motion.

A skeletal configuration is usually separated into chains such as a robot arm or an animated graphics character leg. Each *chain* is a sequence of rigid *links* connected to each other at their ends by rotating *joints*. Any translation and/or rotation of the ith joint affects the translation and rotation of any joint placed later in the chain. Chains can be formalized as follows: all links with no children are marked as *end links*; a chain can be built for each end link by moving back through the skeleton, going from child to parent, until the *root* (the start of the chain) is reached. Each root can be the starting point of multiple chains.

2.1. Motion

Once a body model has been defined, it can then be animated, manipulated or used for simulation purposes. *Motion* is the change in position of an object with respect to a reference. The movement and position of some specific joints in the skeleton are of more interest than others. For example, the position of the grabbing end of a robotic arm or the position of the legs in a walking character. These are designated as the *end effectors*. Motion of a body can be obtained using kinematics in two ways.

- *Forward Kinematics* (FK) can be defined as the problem of locating the end effectors' positions after applying known transformations to the chain. In this case, the joint angles and the link lengths are known and given.
- IK is the problem of determining an appropriate joint configuration for which the end effectors move to desired *target positions*, as rapidly, and accurately as possible.

The input to the FK problem are the joint angles of all joints. The FK problem has a unique solution, and its success depends on whether the joints are allowed to perform the desired transformations. The input to an IK problem is a set of specified positions and orientations for the end effectors, called *targets*; each end effector

[1] For many solvers, the resulting motion, and therefore the associated smoothness, is simply that which results from convergence of the algorithm under a given set of constraints.

is assigned a different target, but under certain circumstances, they can coincide. A solution to the IK problem is a joint configuration of the full skeleton that allows the end effectors to reach their targets. By definition, the root joint is assumed to be fixed, but methods can also cope with translation of the root.

In contrast to FK, IK may have multiple solutions (which is the most common case), a unique solution or no solution. The number of available solutions depends on the target(s) position(s) and/or the DoFs of the kinematic chain. For example, a target may be unreachable when it is located further than the chain can reach, or can be at a point where no pivoting of links can bend the chain to reach the target(s). Another example is when two or more targets for chains conflict and cannot be satisfied simultaneously. IK problems with unreachable targets are known as *over-constrained* problems. When a target is reachable, multiple solutions may exist for a two or more links chain. This makes the IK problem *under-constrained* (or *redundant*)—it may have an infinite number of solutions that satisfy the desired target. For instance, using the human skeleton with (at least) 70 DoF, any of its end effectors can reach a desired target in multiple ways, resulting in many different possible poses. It is up to the user to choose the most appropriate IK solver for the given application, which may depend on many criteria, including the smoothness of the given solution, and the computational cost of choosing that solution.

There are many factors to take into consideration and decide which is the best posture (solution) for an IK solver. Usually, it is desirable to move in a straight line between the end effector and the target positions when the motion of the kinematic chain requires large changes in energy and momentum; however, continuous speed curves can also be used for linking positions (on expensive professional robotic arms, there is an option to choose how to go between the two points). Certain joint or model limitations must be satisfied so that the final joint configuration is within a feasible set of poses. It is important to note that the naturalness of the human movement can be evaluated based on observation of natural human movements or neurophysiological experiments. The smoothness of the motion produced, as well as the temporal coherency, is another important factor for evaluating the IK solution. Nevertheless, what is natural is subjective and varies based on the problem and/or the kinematic chain that is moved.

2.2. Reachable and non-reachable targets

There are instances when a solution to the IK problem does not exist due to an unreachable target. The space of targets that the end effector can reach is called *reachable workspace* (in robotics, the space in which the robot can generate velocities that span the complete tangent space at that point is known as *dexterous space*). It is important to check whether a target is within reach or not, as a significant amount of processing time can be saved if we avoid searching for a solution that does not exist.

In cases where joint constraints and target orientations are not taken into account, a simple reachability check can be applied as follows: let the distance between the (sub-)root[2] and the target be d; if this distance is larger than the total sum of all the inter-joint distances, i.e. if $d > \sum_{i=1}^{n-1} d_i$, where $d_i = |\mathbf{p}_{i+1} - \mathbf{p}_i|$, for $i = 1, \ldots, n-1$, and $\mathbf{p}_i$ is the position of the ith joint of the kinematic chain, the target is unreachable. In the case that the distance is smaller, it is necessary first to evaluate whether the kinematic chain can bend enough to reach the target, before deciding that it is within the reachable bounds. Such a case occurs when the kinematic chain consists of a link with size $d_{\max}$ that is larger than the sum of all the remaining links $d_{\max} > \sum_{i=1}^{n-1} d_i - d_{\max}$ and the target is located at a distance $d < d_{\max} - \sum_{i=1}^{n-1} d_i$ from the root. This is illustrated in Figure 1, where a solution cannot be formed if the target is either inside the inner circle of radius $d_1 - (d_2 + d_3)$, or outside the outer circle of radius $d_1 + d_2 + d_3$. Note that the aforementioned formalization is also valid when end effectors are taken to be inner joints (a joint in the chain).

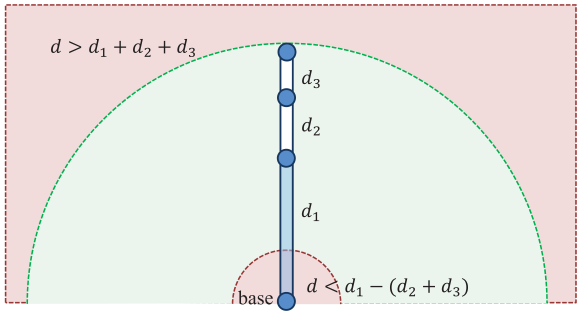

Figure 1: *The reachable and unreachable cases of the IK problem. The target is unreachable if the distance between the target and the base d is larger than the total sum of all inter-joint distances $d > \sum_{i=1}^{n-1} d_i$ or smaller than $d < d_1 - \sum_{i=2}^{n-1} d_i$. The reachable bounds are shaded in green, while the unreachable cases are outside the outer circle or inside the inner circle shaded in red.*

The problem of identifying the reachable workspace for an n-link chain (where $n > 3$) with rotational and translational constraints can become challenging. Further investigations are needed to find methods to determine *a priori* if the target is within reaching bounds and whether optimization procedures can be applied. Thus, to avoid cases where the iterative process enters an endless loop, it is advisable to add termination conditions even when a solution is not reached. One termination condition could depend on the difference between the position of the end effector at the previous and current iteration; if this difference is less than an indicated tolerance, the IK solver should terminate. In addition, if the number of iterations has exceeded a specified limit, the algorithm should terminate.

3. Formulation of Inverse Kinematics

There are many different joint types that deal with different motion properties, e.g. rotational, translational, etc. A more detailed description of various joint types is given in Section 8. A multi-body model, such as a humanoid, consists of many interconnected joints.

Let $\theta_1, \ldots, \theta_n$ be the scalars which describe the complete joint configuration of the multi-body, where n is the number of joints. Each θ_j value is called a *joint angle*, and represents the angle in

[2] Sub-roots are joints that connect two or more kinematic chains.

the plane of rotation, assuming the rotation axis is known (note that angle joints are just one way to describe a joint configuration). Certain points on the kinematic chain, not necessarily located at the end of the chain, are defined as *end effectors*, and the input to the IK problem are the target positions (and possibly orientation) of these end effectors.

In order to solve the IK problem, the joint angles must be set so that the resulting configuration moves each end effector as close as possible to its target position. Let the end effectors' positions be denoted as $\mathbf{s}_1, \ldots, \mathbf{s}_k$, where k is the number of end effectors. Each $\mathbf{s}_i$ can be expressed as a function of the joint angles. The vector $s = (\mathbf{s}_1, \mathbf{s}_2, \ldots, \mathbf{s}_k)^T$ can be viewed as a column vector either with $m = 3k$ scalar entries or with k entries from $\mathbb{R}^3$. The target positions, one for each end effector, are defined by a vector $\mathbf{t} = (\mathbf{t}_1, \mathbf{t}_2, \ldots, \mathbf{t}_k)^T$, where $\mathbf{t}_i$ is the target position for the ith end effector. Finally, the desired change in position of the ith end effector is given by $\mathbf{e}_i = \mathbf{t}_i - \mathbf{s}_i$; this equation can be equally formatted as $\mathbf{e} = \mathbf{t} - \mathbf{s}$. The raw potential change defined by $\mathbf{e}$ is guiding the convergence, but its amplitude might be clamped to account for the validity domain of some IK methods (e.g. Jacobian, heuristic).

Given a set of joint angles in the form of a column vector $\boldsymbol{\theta} = (\theta_1, \ldots, \theta_n)^T$, the end effector positions can be expressed as functions of the joint angles:

$$\mathbf{s} = f(\boldsymbol{\theta}), \tag{1}$$

or, for $i = 1, \ldots, k$, $\mathbf{s}_i = f_i(\boldsymbol{\theta})$. This is called the *FK* solution.

On the other hand, the goal of *IK* is to find a vector $\boldsymbol{\theta}$ such that $\mathbf{s}$ is equal to a given desired configuration $\mathbf{s}_d$:

$$\boldsymbol{\theta} = f^{-1}(\mathbf{s}_d), \tag{2}$$

where f is a highly non-linear operator which is difficult to invert. There are multiple possible solutions for $\boldsymbol{\theta}$ and we want one that gives us the pose that returns the smoothest motion. Aside from that, we want the solution $\boldsymbol{\theta}$ to be stable, meaning that we prefer a solution that when the end effectors' position and orientation slightly change, the configuration of the chain $\boldsymbol{\theta}$ will also only exhibit a small change.

4. Analytic Solutions

Analytical methods are the first family of IK solvers discussed in this paper. They are meant to find all possible solutions as a function of the lengths of the mechanism, its starting posture and the rotation constraints, but usually are built upon some assumptions to compute just a single solution. The simplest non-trivial manipulator is the planar two-link manipulator; when the chain consists of such a small number of joints, the solution, $\boldsymbol{\theta}$, can be computed analytically by trying out all possible ways in which the links can be placed.

Let the two links of the chain have lengths l_1 and l_2 and the coordinate (x, y) be the target position; for simplicity it is assumed that the target is within the reachable bounds of the chain. Looking at Figure 2, it is obvious that there are two possible configurations with the end effector reaching the target position (x, y); in this survey, only the rotation angles of the upper solution are calculated as an example of implementation. The relative rotation angles are denoted by θ_1 and θ_2; since the lengths of the two links l_1 and l_2 are known,

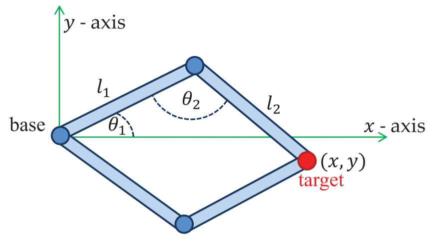

Figure 2: *Analytic solution of a two-link kinematic chain; the presented manipulator may have two possible solutions.*

as well as the desired end effector position, the joint angles can be determined as:

$$\theta_1 = \cos^{-1}\left(\frac{l_1^2 + x^2 + y^2 - l_2^2}{2l_1\sqrt{x^2 + y^2}}\right) \tag{3}$$

and

$$\theta_2 = \cos^{-1}\left(\frac{l_1^2 + l_2^2 - (x^2 + y^2)}{2l_1 l_2}\right). \tag{4}$$

Obviously, the calculation of the relative joint angles to find a solution becomes harder when the chain is larger. In addition, it is hard to find a convenient way to define which of the different solutions should be chosen. One constraint will be that for a smooth motion we desire the solution $\boldsymbol{\theta}$ to be stable. Clearly, when we move from planar to three-dimensional space, more solutions will be available and the calculations will become correspondingly more difficult.

There are numerous papers that deal with analytical IK solutions in robotics, solving general 6R manipulators[3] [RR93, MC94] or multi-body mechanisms [PSM88, GORH05]. A review of analytical methods is given in Craig's book [Cra03]. IKFast [Dia10] is a tool that solves robot IK equations in an analytic form, deploying motion planning algorithms in real-world robotics applications. Analytic solutions have also been used to animate anthropomorphic limbs, such as the method presented by Korein [Kor85] to manipulate human arms and legs, or the method developed by Tolani *et al.* [TGB00], which used the swivel representation to form an analytic solution of human limbs. Kallmann [Kal08] revised the formulation described in Tolani *et al.* using quaternion algebra, extending the method to automatically determine swivel angles. More recently, Molla and Boulic [MB13] proposed a middle-axis-rotation parameterization of human limbs to deal with ill-conditioned cases that happen due to the swivel representation; thus, rather than projecting onto a fixed vector that may result in large deviation in the swivel angle, the authors define a reference coordinate system by decomposing the rotation in a manner which avoids singularities.

The analytical IK solutions usually do not suffer from singularity problems, they offer a global solution, and they are reliable, which is the main reason why they are exploited in robotics. In addition, closed-form solutions are preferred for motion planning due to their low computational cost. Numerical and/or iterative IK solvers are

[3] A general 6R manipulator has six rotary DoFs.

much slower, while planners (IK solvers) are required to process thousands of configurations per second. However, the non-linear nature of the kinematic equations and their lack of scalability make them less suitable for redundant systems, they often fall into local minima, and (in their simple form) they cannot handle prioritized constraints. They are mainly used for mechanisms with low DoF, and are not scalable enough to meet the demands of modern computer-based IK problems. For instance, even in well-behaved situations, a closed-form equation cannot generally be achieved for the full human body, which has approximately 70 DoF. In this manner, scholars search for alternative methods to iteratively approximate a good solution to the problem; such methods are presented in the next section and are based on a numerical approximation of the non-linear problem.

5. Numerical Solutions

Numerical methods cover those that require a set of iterations to achieve a satisfactory solution. The iterative methods formulate the problem using a cost function to be minimized. The numerical family of methods can be generally divided into three categories: *Jacobian*, *Newton* and *Heuristic* methods.

5.1. Jacobian inverse methods

The Jacobian J is a matrix of partial derivatives of the entire chain system with respect to the angle parameters, $\boldsymbol{\theta}$. The *Jacobian solutions* offer a linear approximation to the IK problem (see Figure 3). An excellent review of the Jacobian methods is given by Buss in [Bus09]. For convenience, the introduction of the Jacobian solutions in this survey follows the work of Buss and will use the same notation.

The Jacobian methods iteratively solve the IK problem by repeatedly changing the configuration of a complete chain such that it brings the end effector position and orientation closer, at each step, to a target position and orientation. Differentiation of Equation (3) gives the forward dynamics equation (where over-dot denotes the time derivative)

$$\dot{\mathbf{s}} = J(\boldsymbol{\theta})\dot{\boldsymbol{\theta}}. \tag{5}$$

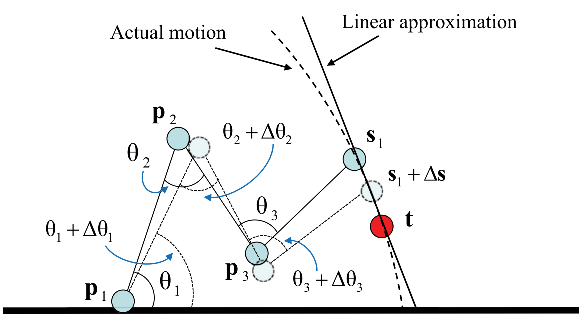

Figure 3: *The Jacobian solution is a linear approximation of the actual motion of the kinematic chain.*

The Jacobian matrix J can be described as a function of the $\boldsymbol{\theta}$ values and is given by

$$J(\boldsymbol{\theta})_{ij} = \left(\frac{\partial \mathbf{s}_i}{\partial \theta_j}\right), \tag{6}$$

where $i = 1, \ldots, k$ and $j = 1, \ldots, n$ (where k is the number of end effectors, and n is the number of joints). Thus, J would be a $k \times n$ matrix with vector entries. In practice, this would be converted to a $3k \times n$ matrix of scalar entries. Following the notation of Buss, we can calculate the entries of J using quantities $\mathbf{v}_j$, which are the unit vectors pointing along the rotation axis of the jth joint:

$$\frac{\partial \mathbf{s}_i}{\partial \theta_j} = \mathbf{v}_j \times (\mathbf{s}_i - \mathbf{p}_j), \tag{7}$$

where $\mathbf{p}_j$ is the position of the joint.

Now, suppose the target position for end effector i is $\mathbf{t}_i$, we then attempt to find the values, $\boldsymbol{\theta}$, which minimize the errors, $\mathbf{e}_i$, between the actual end effector positions and the target positions:

$$\mathbf{e}_i = \mathbf{t}_i - \mathbf{s}_i(\boldsymbol{\theta}). \tag{8}$$

To do this, we make a small change, $\Delta\boldsymbol{\theta}$, in the joint angles and approximate the consequent change in end effector positions as

$$\Delta\mathbf{s} \approx J\Delta\boldsymbol{\theta}. \tag{9}$$

J can be calculated from the current values of $\mathbf{s}$ and $\boldsymbol{\theta}$. Since we are looking for a value of $\Delta\mathbf{s}$ which is as close as possible to the error $\mathbf{e}$ (the error term $\mathbf{e}$ should be clamped to avoid instabilities in convergence), we can estimate the change in $\boldsymbol{\theta}$ to be $\Delta\boldsymbol{\theta} \approx J^{-1}\mathbf{e}$.

However, J may be neither square nor invertible, and, in addition, can suffer from singularity problems.[4] We note here that the Jacobian considers the influence of each joint independently of other joints (it is a first-order approximation)—when one joint changes, all its children segments are viewed as a single rigid body. Various methods have been proposed over time that differ in terms of the approximations used to solve the IK problem, aiming to avoid singularity problems, and improve both the convergence and stability of the solution. Some methods focus more on local modification of the inverse differential kinematic mapping, which is ill-conditioned near singularities, by suitably-defined mappings relating the task-space to the joint-space, while others target smoothing of the overall motion [SK16]. Next, we summarize these techniques.

5.1.1. *Jacobian transpose*

The Jacobian transpose ensures that it is invertible, using the transpose instead of its inverse. Hence,

$$\Delta\boldsymbol{\theta} = \alpha J^T \mathbf{e}, \tag{10}$$

for some appropriate scalar α that can be calculated as

$$\alpha = \frac{\langle \mathbf{e}, JJ^T\mathbf{e}\rangle}{\langle JJ^T\mathbf{e}, JJ^T\mathbf{e}\rangle}. \tag{11}$$

where $\langle \mathbf{a}, \mathbf{b}\rangle$ indicates the dot product between vectors $\mathbf{a}$ and $\mathbf{b}$.

[4]Singularities occur when no change in joint angle can achieve a desired change in a chain's end position.

The Jacobian transpose method is not ill-defined near a geometric singularity, but it requires many iterations to converge. It is important to note that it is possible to verify whether there are singularity issues by determining if the Jacobian matrix has a zero row. The unnatural results of the Jacobian transpose have also been observed in [UPBS08], where the authors observed jerky movements, resulting in poor poses especially in cases where there is a significant difference between the end effector positions and the targets. In addition, the transpose approximation does not consider the relative contribution of joint variables and does not support strict priorities among constrained dimensions. Note also that α should be small due to the non-linearity of the direct kinematics model; otherwise, oscillations and discontinuities may appear.

5.1.2. *Jacobian pseudo-inverse*

The Jacobian pseudo-inverse, also known as the *Moore–Penrose* inverse of the Jacobian, sets $\Delta\boldsymbol{\theta} = J^{\dagger}\mathbf{e}$, where $J^{\dagger}$ is an $n \times m$ matrix and is called the *pseudo-inverse* of J. The main advantage of the pseudo-inverse is that it is defined for all matrices J, even ones which are not square or not of full rank. The pseudo-inverse has the property that the matrix $(I - J^{\dagger}J)$ performs a projection onto the null space of J.

The Jacobian pseudo-inverse is computed as

$$\Delta\boldsymbol{\theta} = J^T(JJ^T)^{-1}\mathbf{e}. \tag{12}$$

Another formula to estimate the pseudo-inverse when J is not full rank is given in [Bus03]. Several authors have used the null space method to help avoid singular configurations, such as Liegeois in [Lie77] and Maciejewski and Klein in [MK85]. In addition, Girard and Maciejewski [GM85] used the Jacobian pseudo-inverse to address the locomotion of a legged figure. However, the Jacobian pseudo-inverse is not free from drawbacks; for instance, when the configuration of a chain is close to a singularity, then the pseudo-inverse method will lead to very large changes in joint angles, while the movement to the target might be very small, resulting in oscillations and discontinuities in motion.

5.1.3. *Damped least squares*

Another variation of the Jacobian is the damped least squares (DLS) method, also known as the *Levenberg–Marquardt* algorithm, which was first used for IK by Wampler in [Wam86] and Nakamura and Hanafusa in [NH86]. Using DLS $\Delta\boldsymbol{\theta}$ is stabilized, avoiding many of the pseudo-inverse method's problems with singularities. The DLS solution can be expressed as

$$\Delta\boldsymbol{\theta} = J^T(JJ^T + \lambda^2 I)^{-1}\mathbf{e}, \tag{13}$$

where $\lambda \in \mathbb{R}$ is a non-zero damping constant. The damping constant must be chosen carefully in order to make Equation (13) numerically stable. Buss and Kim [BK05] observed that DLS works better than the pseudo-inverse and transpose methods. However, its superior behaviour is subject to the damping constant λ. A large damping constant makes the solutions for $\Delta\boldsymbol{\theta}$ well behaved near singularities, but also lowers the convergence rate, reduces the accuracy in tracking the targets and generates oscillation and shaking.

5.1.4. *Singular value decomposition*

The presence of singularities considerably complicates the Jacobian inversion process. For this purpose, the singular value decomposition (SVD) has been proposed as another variation of the Jacobian method that utilizes the pseudo-inverse matrix [PTWF92]. SVD is particularly useful since it provides orthonormal bases for the fundamental subspaces of a matrix. Formally, the SVD of an $m \times n$ Jacobian matrix J is a factorization of the form $J = UDV^T$, where U is an $m \times m$ unitary orthogonal matrix, D is an $m \times n$ rectangular diagonal matrix with nonnegative real numbers on the diagonal and V^T (the transpose of V) is an $n \times n$ unitary orthogonal matrix. The diagonal entries of the D matrix $\sigma_i = d_{ii}$ are known as the singular values of J. Note that σ_i may be zero, and the rank of J is equal to the largest value r such that $\sigma_r \neq 0$, while $\sigma_r = 0$ when $i > r$.

The Jacobian pseudo-inverse can be expressed using the SVD as $J^{\dagger} = VD^{\dagger}U^T$. In particular, the pseudo-inverse $J^{\dagger}$ is given by

$$J^{\dagger} = \sum_{i=1}^{r} \sigma_i^{-1}\mathbf{v}_i\mathbf{u}_i^T. \tag{14}$$

Colomè and Torras [CT12] proposed a singular value filtering (SVF) approach, which is an alternative way of filtering the Jacobian matrix so that it is always a full-rank matrix; the new alternative pseudo-inverse has lower-bounded singular values and tends to $J^{\dagger}$ when its singular values move away from 0.

5.1.5. *Pseudo-inverse damped least squares*

The pseudo-inverse DLS method is an extension of the DLS method that uses the SVD under the DLS method [Mac90], [PTWF92]. Thus, it can be expressed as

$$J^T(JJ^T + \lambda^2 I)^{-1} = \sum_{i=1}^{r} \frac{\sigma_i}{\sigma_i^2 + \lambda^2}\mathbf{v}_i\mathbf{u}_i^T. \tag{15}$$

The pseudo-inverse DLS method performs similarly to the simple pseudo-inverse when away from singularities, but smooths out the performance in areas near singularities. More specifically, the Jacobian in both methods is inverted by an expression $\sum_{i=1}^{n} \tau_i\mathbf{v}_i\mathbf{u}_i^T$. However, in the pseudo-inverse DLS case, $\tau_i = \sigma_i/(\sigma_i^2 + \lambda^2)$, whereas for the simple pseudo-inverse method, $\tau_i = \sigma_i^{-1}$, which makes it unstable as σ_i approaches zero.

5.1.6. *Selectively damped least squares*

The selectively SDLS method was presented by Buss and Kim in [BK05] and is an extension of the pseudo-inverse DLS method. SDLS adjusts the damping factor separately for each singular vector of the Jacobian SVD based on the difficulty of reaching the target positions. The damping constants of SDLS depend not only on the current configuration of the articulated multi-body, but also on the relative positions of the end effector and the target position. SDLS needs fewer iterations to converge, does not require ad hoc damping constants and returns the best results in terms of lack of oscillations. Buss and Kim [BK05] showed that it performs better than any other inverse Jacobian method, with its drawback being that it has high computational cost (it has the slowest performance time among all Jacobian methods) due to the SVD computation. Bensalah *et al.* [BGQHA13] has recently extended the SDLS method for the calculation of the IK of anthropomorphic robotic hands. They attempt to reduce the high cost of the original method, which arises from computing the SVD of the extended Jacobian matrix, by computing only the relevant smallest singular values and corresponding vectors. They associate a specific damping factor for each estimated singular value and calculate the joint angles using Cholesky decomposition.

5.1.7. *Other Jacobian solutions*

There are various methods that use Jacobian alternatives to solve the IK problem (e.g. [WW92, ZB94]). For instance, Kenwright in [Ken12b] employed the Gauss–Seidel algorithm to solve character IK problems. The author constructed a Jacobian matrix with a linear equation format, $\Delta\mathbf{s} \approx J\Delta\boldsymbol{\theta}$, and estimated the unknown $\Delta\boldsymbol{\theta}$ using the Gauss–Seidel iterative method; $J = J^T J + \delta I$, where the damping value δI was incorporated to prevent singularities and make the final method more stable and robust. Meredith and Maddock [MM04] demonstrate how a 'half-Jacobian' alternative can be used instead of the full Jacobian solution, resulting in reduced computational cost when applying IK in articulated characters.

Baillieul in [Bai85] proposed the extended Jacobian technique, which adds additional rows to the Jacobian. Khatib [Kha87], as well as Orin and Schrader [OS84], computed a first-order Jacobian matrix for the robot, which maps joint velocities into task space velocities, and inverted this to map the error into a joint state update. Another Jacobian alternative technique has been proposed by Boulic *et al.* [BMT95], where the range of IK has been extended by integrating the mass distribution information to control the position of the centre of gravity of articulated figures.

Most numerical methods can be further strengthened by imposing priorities for the proper implementation of specific constraints [NHY87]; see also the concept of a pin-and-drag interface [YN03]. Taking into account that the human structure is highly redundant, leading to conflicts between multiple tasks, Siciliano and Slotine [SS91] formulated the general multiple priority DLS framework to avoid such conflicts between tasks. Similarly, Choi and Ko [CK99] presented the online motion re-targetting (OMR) method based on the Jacobian pseudo-inverse; they used two levels of priorities to control the posture of humanoids and cope with multiple end effectors. Baerlocher and Boulic [BB98] proposed the *Augmented* Jacobian to solve the IK problem with priorities (again for two levels), that was later extended, in a scheme called *Prioritized IK* (PIK), to deal with full-body manipulation [BB04]. PIK allows constraints to be associated with a priority level in order to enforce important properties first. The task-based Jacobian and its null space projection operators can handle kinematic problems with multiple-priority levels and with highly redundant structures. However, the simple inversion of the Jacobian does not take into account unilateral constraints, such as the joint range, velocity and acceleration limits.

In [Pec08], Pechev introduced the feedback IK (FIK) method, which solves the IK problem from a control prospective, minimizing the difference between demanded and actual Cartesian velocities. Within the feedback loop, the required joint parameters are derived through a control sensitivity function. The algorithm operates as a filter and does not require matrix manipulations (e.g. inversion). Singularities are handled without the necessity of a damping factor and this makes it computationally more efficient than pseudo-inverse-based methods. The author also describes how manipulator constraints can be applied, weighting both joints and end effectors to a more feasible set of postures. As with the other Jacobian-based algorithms, it can easily handle problems with multiple end effectors.

5.1.8. *Incorporating constraints*

Implementing constraints in the Jacobian family of methods, so as to improve the performance and increase the realism of the reconstructed pose, is not straightforward. Some effort has been made over the years to deal with joint limitations. For instance, a simple projection of the unconstrained solution onto a feasible posture has been proposed by Welman in [Wel93]. However, it is not guaranteed that the result will lie close to an optimal solution. A penalty-based method adding movement restrictions is presented by Fêdor in [FÔ3], with the drawback that this often converges to poor results. Some other popular methods to control the kinematic chain under joint constraints are task-priority [NHY87, SS91] and the null space projection of the Jacobian [CD95, DVS01, CW93]. Several optimizations have been proposed, especially in the robotics domain, such as Kanoun *et al.* [KLW11], the SNS (saturation in the null space) [FDLK12] and IKTC (IK with task corrections) [KO13] methods. These approaches generally discard the use of joints that exceed their motion bounds when using the minimum norm solution, reintroducing them in a suitable null space.

The simplest way of incorporating constraints can be achieved by weighting the moves of the individual joints, as proposed by Meredith and Maddock in [MM05]. Their technique provides the ability to predictably modify how much different DoFs change when configuring a posture using the equation $\Delta\boldsymbol{\theta} = WJ^{-1}\mathbf{e}$, where W is a weighting vector that contains values between 0 and 1. Finally, Kenwright [Ken13] incorporated joint limits by modifying the update scheme to include an iterative projection technique, named projected Gauss–Seidel; the angular limits form bounds that are enforced through clamping. More details about joint and model restrictions are given in Section 8.

5.2. Newton methods

The Newton methods are based on a second-order Taylor expansion of the objective function $f(x+\sigma)$

$$f(x+\sigma) \approx f(x) + [\nabla f(x)]^T \sigma + \frac{1}{2}\sigma^T H_f(x)\sigma, \qquad (16)$$

where $f(x+\sigma)$ are the desired joint parameters, $f(x)$ are the current joint parameters, σ is the required modification so that $f(x)$

will satisfy $f(x+\sigma)$ and $H_f(x)$ is the Hessian matrix. In contrast to the Jacobian methods, here the solution is found as a second-order approximation of the function $f(x+\sigma)$ using the quasi-Newton method, as described by Nocedal and Wright in [NW99]. In this way, the singularity problems of the Jacobian matrices are avoided. The search direction p_k^N is found by solving the equation:

$$p_k^N = \nabla^2 f_k^{-1} \nabla f_k, \tag{17}$$

where $\nabla^2 f_k^{-1}$ is the inverted Hessian matrix and ∇f_k is the gradient of the objective function given by:

$$\nabla f_k = J(e-g), \tag{18}$$

where e is the end effector position and g is the goal position. However, the calculation of the Hessian matrix is very complex and results in high computational cost for each iteration. Hence, several approaches have been proposed which, instead of calculating the Hessian matrix, use an approximation of the Hessian based on a function gradient value. The most well-known methods are the Broyden's method, the Powell's method and the Siciliano's method, as well as the Broyden, Fletcher, Goldfarb and Shanno (BFGS) method [Fle87, Sic90, CvKM97]. For instance, BFGS uses the following formula to obtain an estimate of the Hessian:

$$B_{i+1} = B_i - \frac{B_i s_i s_i^T B_i}{s^T - i B_i s_i} + \frac{g_i g_i^T}{g_i^T s_i}, \tag{19}$$

where i denotes the present iteration, $s_i = x_{i+1} - x_i$, $g_i = \nabla f(x_{i+1}) - \nabla f(x_i)$, and B is a positive definite approximation of the Hessian.

Since the Newton methods are posed as a minimization problem, they return smooth motion without erratic discontinuities. It is also straightforward to incorporate joint restrictions. The most obvious method for constraints is the gradient projection method proposed by Zhao and Badler in [ZB94]; they propose a search for a plausible solution by solving a constrained non-linear optimization process. In addition, Rose *et al.* [RGBC96] extended the constrained non-linear optimization formula of Zhao and Badler in order to handle variational constraints that hold over an interval of motion frames. The Newton methods also have the advantage that they do not suffer from singularity problems, such as to those that occur when finding the Jacobian inverse. However, they are complex, difficult to implement and have high computational cost per iteration.

5.3. Heuristic inverse kinematics algorithms

The heuristic sub-family of algorithms implements simple ways for solving the IK problem without using complex equations and calculations. These algorithms are usually composed of simple operations, in an iterative fashion, that gradually lead to an IK solution. Heuristic IK algorithms have low computational cost, thus usually end up in the final pose very quickly, and are very good for simple problems, especially for non-anthropometric skeletons (e.g. spiders, insects). One of their main limitations is that, even if all joint constraints are satisfied, they suffer from unnatural or biomechanically unfeasible motions and gestures. Heuristic solvers do not take into consideration spatio-temporal corrections between nearby joints, as they treat each joint's constraint independently with no global constraints. The following sub-sections present and discuss the most popular heuristic IK solvers, as well as ways to overcome the current limitations.

5.3.1. *Cyclic coordinate descent*

Cyclic coordinate descent (CCD), initially proposed in robotics by Luenberger [Lue89] and Wang *et al.* [WC91], is an iterative heuristic technique that is suitable for interactive control of an articulated body. CCD is one of the most popular IK iterative algorithms; it has been implemented for human-like manipulation in many computer graphics and robotics applications [Lan98]. Kenwight has presented a comprehensive review of CCD in [Ken12a], examining its viability for creating and controlling highly articulated characters; in addition, he discussed implementation details and the algorithm limitations.

The CCD method attempts to minimize position and orientation errors by transforming one joint variable at a time. The main idea behind CCD is to align each joint position with the end effector and the target at each step; starting from the end effector and moving inward towards the manipulator base, each joint angle is transformed so that the last bone of the chain gets closer to the target. Assume a kinematic chain consists of n joints, where $\mathbf{p}_1$ is the root joint and $\mathbf{p}_n$ is the end effector, and let $\mathbf{t}$ be the target position. First, find the angle θ_{n-1} defined by the target position, $\mathbf{p}_{n-1}$ and the end effector. Then, update the end effector's position by rotating $\mathbf{p}_n$ so that θ_{n-1} is set to zero. Similarly, find the angle θ_{n-2} defined by the target position, $\mathbf{p}_{n-2}$, and the end effector, and update $\mathbf{p}_{n-1}$ and $\mathbf{p}_n$ positions so that θ_{n-2} is zero. An iteration is completed when all joints are updated. This procedure is repeated until the end effector is satisfactorily close[5] to the target position.

CCD is very simple to implement, requiring only a dot and cross product; thereby, it has low computational cost per iteration. It provides a numerically stable solution and it has linear-time complexity in the number of degrees of freedom. CCD can be easily extended to include local constraints, such as the methods described by Welman [Wel93] and Lander [Lan98], where the allowable angle transformation is bounded at each step by upper and lower limits. By definition, CCD only handles serial chains; in this direction, Merrick and Dwyer, [MD04], described an extension which deals with tree articulated structures and multiple end effectors. The proposed multiple-chain method can be applied successively over multiple articulated chains; it divides the articulated structure into smaller serial chains and treats each chain independently. Similarly, Shin *et al.* [SLSG01] and Kulpa *et al.* [KM05, KMA05] proposed to divide the skeleton into sub-categories and then apply CCD hierarchically and iteratively. However, CCD is not free from problems; the first issue arises from its poor motion distribution, since it tends to overemphasize the movements of the joints closer to the end effector, which may lead to the production of unnatural postures. CCD may generate large angle rotations that often produce motion with erratic discontinuities and oscillations. In some cases, particularly when the target is located close to the base, it causes a chain to form a loop, rolling and unrolling itself before reaching the target. Similarly, for certain target positions (especially when high accuracy

[5] Less than a user specified threshold.

is required), the algorithm can take a large number of iterations, resulting in a slow zigzag motion of the end effector.

Generally, even if constraints have been incorporated, unrealistic poses may be generated, especially in highly articulated characters, since it is difficult to implement global manipulation restrictions. In this manner, Mahmudi and Kallmann's proposed a variation of CCD named inverse branch kinematics (IBK) [MK11], aiming to deal with the production of unnatural pose, by assigning a continuous rotational range to control a pre-defined global-transition-cost threshold. Moreover, Kenwright [Ken12a] introduced a biasing factor into each iteration in order to correct the rotation at each time. He also improved the CCD convergence by adding a feedback constant, which is the distance between end effector and target, which later multiplies with the under-damped factor to produce smaller values closer to the target. This allows movement only in those links that are necessary to accomplish the given task.

There are many variations and extensions of the CCD algorithm that have tried to improve its performance; for instance, the circular alignment algorithm (CAA) [Muk12] placed the given joint chain along a circular arc between the base and the target position. Thus, it is ensured that there is a solution available and there is no possibility of the chain intersecting itself (a common problem in CCD). CAA, however, requires all links to have the same length and only works on a two-dimensional plane containing the base of the kinematic chain and the target. The inductive IK (IIK) algorithm [KLC*03] is another extension of CCD; it uses a uniform posture map (UPM) to control the posture of a human-like 3D character, while the learning algorithm prevents the generation of invalid output neurons. The IIK algorithm forms a forward kinematic table containing the FK values of each output neuron, and searches for the FK points that are closest to the desired position so as to get a more natural posture.

5.3.2. *Forward and backward reaching inverse kinematics*

Another heuristic method for solving the IK problem is forward and backward reaching IK (FABRIK), introduced by Aristidou and Lasenby [AL11]; the idea of FABRIK is similar to the follow the leader (FTL) algorithm [BLM04] that has been used for rope simulation, which, instead of using angle rotations, updates the joint's new positions along a line to the next joint. However, unlike FTL that operates in a single iteration, FABRIK works in a forward and backward iterative mode, minimizing at each time the distance between the target and the end effector.

Let $\mathbf{p}_1, \ldots, \mathbf{p}_n$ be the joint positions of a kinematic chain, where again $\mathbf{p}_1$ is the root joint, $\mathbf{p}_n$ is the end effector and $\mathbf{t}$ is the target position. The distances between each pair of joints are denoted as $d_i = |\mathbf{p}_{i+1} - \mathbf{p}_i|$, for $i = 1, \ldots, n-1$. In the forward stage, the algorithm estimates each joint position starting from the end effector, $\mathbf{p}_n$, moving inwards to the manipulator base, $\mathbf{p}_1$. The new position of the end effector is assigned to be the target position, $\mathbf{p}'_n = \mathbf{t}$. Then, find the line, l_{n-1}, which passes through the joint positions $\mathbf{p}_{n-1}$ and $\mathbf{p}'_n$, and place $\mathbf{p}'_{n-1}$ at the position that lies on this line at a distance d_{n-1} from $\mathbf{p}'_n$. Similarly, the joint position $\mathbf{p}'_{n-2}$ is computed using the line l_{n-2}, which passes through the $\mathbf{p}_{n-2}$ and $\mathbf{p}'_{n-1}$, with a distance d_{n-2} from $\mathbf{p}'_{n-1}$. The algorithm continues until all joint positions $\mathbf{p}_1, \ldots, \mathbf{p}_n$ are updated. In the backward stage of the algorithm, the same procedure is repeated but this time moving backwards from the manipulator's base to the end effector, this completes one full iteration. The procedure is then repeated, for as many iterations as needed, until the end effector is identical or close enough to the desired target. FABRIK always converges to any given goal positions, when the target is within reach, as proved in [ACL16].

FABRIK's main advantages are its simplicity, its flexibility to be easily adapted to different problems, the low computational cost and its effectiveness on handling closed loops or problems with multiple end effectors. The latter can be achieved by dividing the algorithm into smaller sub-sections and using sub-bases as intermediate stops. FABRIK also supports most of the anthropometric and robotic joint constraints by repositioning and reorienting the target to be within the allowable bounds [ACL16, Xie16]. FABRIK reaches the target very fast, converges to the target even if the error tolerance is set to zero and is well balanced near singularities [PR13]. Another feature of FABRIK is that, in contrast to CCD that overemphasizes motion close to the end effector, it distributes movements to all joints.

Numerous methods have been proposed to extend FABRIK. Huang and Pelachaud [HP12] used a variation of FABRIK to solve the IK problem from an energy transfer perspective. They used a mass-spring model to adjust the joint positions by minimizing the force energy which is conserved in springs. Moya and Colloud [MC13] proved that FABRIK can cope with target priorities, adjusting the initial algorithm to deal with joints that have more than two segments. Aristidou *et al.* [ACL16] applied a constrained version of the FABRIK algorithm in a hierarchical and sequential fashion with priorities to control the movement of a humanoid model. Agrawal and van de Panne [AvdP16] used a data-driven prior to warm start FABRIK for the production of more natural looking human poses, while Tao and Yang [TY16] extended FABRIK to deal with collision free tasks. Collision problems can alternatively be solved using the Brown *et al.* [BLM04] method. Lansley *et al.* [LVSF16] have recently introduced CALIKO, an open library for the FABRIK algorithm.

FABRIK solves the IK problem in position space, instead of orientation space, while joint orientations are dealt with in a separate step (which also adds extra complexity). Hence, it demonstrates less continuity under orientation constraints. In particular, the constrained version of the algorithm encounters a deadlock situation when the kinematic chain is small in size and the joints close to the end effector have strict constraints. In such a case, FABRIK is incapable of finding a solution since the kinematic chain is unable to bend enough and reach the target. This is due to the structure of the algorithm where each joint is treated independently. The target position is projected and re-oriented on the surface of a conic section locally, at each step of the algorithm. Thus, the algorithm does not take into consideration the restrictions on the previous (parent) or next (child) joint in order to push, if possible, the kinematic chain to bend further in the current joint. The authors coped with the deadlock situation using a random perturbation technique to push the parent joints to their limits, allowing more flexion for the child joints [ACL16]. Furthermore, similar to CCD, FABRIK encounters a problem of inability to integrate global model constraints that meet the spatio-temporal correlations between nearby joints.

6. Data-Driven Inverse Kinematics

The recent advances in motion capture technology and the large availability of motion capture (mocap) data in online libraries led to data-driven methods for solving the IK problem. The main idea behind data-driven methods is to use pre-learned postures to match the positions of the end effectors to a feasible pose learnt from the database. In robotics, this is usually done via methods that are based on neural networks and artificial intelligence [OCA*01, XW01, DVS01].

6.1. Learnt methods

One can divide existing learnt methods for learning IK into two groups: the error-based and the example-based methods. Error-based methods improve the inverse kinematic estimate that is used to reach the target position [JR92, WK98], while the example-based methods use example configurations for learning the inverse kinematic estimate [BGG93, RSG10]. The most common strategy followed in computer graphics is the example-based, where postures are reconstructed based on a database of pre-recorded motions. In this direction, Rose *et al.* [RISC01] introduced a method that interpolates example motions and positions to accomplish given human figure tasks. Each example pose is parameterized by the position of certain bones, and based on the parameters of the desired positions, several poses are blended using radial basis functions. Later, Grochow *et al.* [GMHP04] presented a style-based IK method which is based on a learned model of human pose; they model the probability distribution over all possible whole-body poses using scaled Guassian process latent variable models (SGPLVM). Given a set of pose constraints, the proposed system was able to produce the most likely pose satisfying those constraints. During the learning procedure, the system searches for the low-dimensional representation of the input data and creates a probability distribution function (PDF) in the latent space. Finally, it uses the PDF to find the pose which is most likely compared to the input constraints. In general, style-based IK generates natural poses but only in a narrow, human reachable space because of its limited training capacity, as the estimated poses are highly related to the training data. Liu *et al.* [LHP05] extended this work to consider the physical properties of motion, such as muscle forces, gravity and the friction of the feet on the ground. They have utilized non-linear inverse optimization in order to choose movements that use as little energy as possible, preventing sudden changes in the resulting postures.

Searching for pose similarities in large-scale motion capture databases implies high computational cost. A way to decrease the complexity is to reduce the dimensionality of the database, encapsulating only the essential aspects of a specific motion pattern. In this way, Chai and Hodgins [CH05] apply local principal component analysis (PCA) to construct a latent space during run-time and find the closest motion from a pre-defined motion set that matches the current marker movement. Carvalho *et al.* [CBT07] select a small subset of postures using probabilistic PCA, and solve the IK problem completely within the latent space associated with a skill (e.g. golf), whereas Raunhardt and Boulic [RB09] take advantage of an encapsulated skill with PCA to guide the solution of an IK problem without restricting it to the latent space. Priorities are used to achieve some unusual spatial constraints that were not part of the initial dataset used for learning the skill. Tournier *et al.* [TWC*09] approximate the pose manifold using a principal geodesics analysis (PGA), and then use an iterative minimization algorithm to search this manifold for pose matching end effector constraints. Another way to deal with the high computational complexity of searching similar poses in mocap databases was proposed by Wu *et al.* [WTR11], named NAT-IK. Instead of searching in full-scale motion libraries that contain continuous poses, they reduced the search space by utilizing the *k*d-tree clustering technique to select only a representative set of poses. The main advantages over the style-based IK method is that it is designed for per-frame problems, and needs to be trained only once; it is able to deal with large datasets and generates natural poses in a much wider, human-reachable space. Nevertheless, NAT-IK does not run at a high frame-rate, and constraints add a significant amount of extra computational burden.

Given a set of end effector positions, Ong and Hilton [OH06] constrained the pose using a hierarchical cluster model learnt from a motion capture database. Wei and Chai [WC11] formulated the interactive character posing problem via a maximum *a posteriori* (MAP) framework; they segment the motion database into local regions and model each of them individually. More recently, Ho *et al.* [HSCY13] proposed a framework that conserves the topology of the synthesizing postures; using Gauss linking integral (GLI), they avoid body part penetration by distinguishing topologically different postures, thus reducing the processing time required to search for data in the motion database. There is an ongoing effort, looking for efficient methods to establish temporal coherence and continuity in the generated motion [SH12, YVN*14, HMC*15].

Sumner *et al.* [SZGP05] proposed a mesh-based IK method (MESH-IK) which, instead of using human poses as training data, learns the space of meaningful shapes from example meshes. Using the learned space, MESH-IK generates new shapes that respect the deformations exhibited by the examples, yet still satisfy vertex constraints imposed by the user. Der *et al.*in [DSP06] describe an extension of the MESH-IK method which provides interactive control of reduced deformable models via an intuitive IK framework. The collection of transformations compactly represents articulated character movement that has been derived automatically from example data. The IK problem is formulated in a reduced space to achieve an independent resolution performance, meaning the speed of the posing task is a function of the model parameters, rather than of character geometry. Inverse blending [HK10] is another data-driven method which interpolates similar motion examples according to blending weights; in this way, it is able to precisely control end-effectors positions, and meet multiple spatial constraints. More recently, Yoshiyasu and Yamazaki [YY12] introduced Cage IK (CageIK); CageIK, in contrast to MESH-IK, is applicable to more general types of mesh representations. The authors provide a set of cage geometries as examples and interpolate them based on handle movements. CageIK seems to be able to edit a larger range of mesh models than MESH-IK, it can place handles directly on the target model, and can deform the model locally. However, it requires that the target model is in a similar pose to the reference pose, it cannot preserve the shape of each component perfectly when multi-component models are edited, and cannot reproduce high-frequency changes.

Data-driven methods are generally used to ensure natural movements selecting candidates from a feasible set of solutions. Their efficiency depends on the size of the database. Example-based methods require an offline training procedure, and the results are highly correlated with the training data. Therefore, data-driven IK solvers usually suffer from discontinuity in poses [RPE*05]; if the desired postures are too distant from those of the database, the result may be temporally incoherent. In this direction, Chai and Hodgins [CH05] search for the K-closest samples of a posture using a combination of metrics that involves pose and velocity features, to ensure smooth transition and temporal consistency between poses. Then, they build a graph to accelerate the search of the nearest neighbours. Similarly, Krüger *et al.* [KTWZ10] and later Tautges *et al.* [TZK*11] construct a *k*d-tree, and then create a lazy neighbourhood graph (LNG) for fast selection of motions that are temporally coherent. Liu *et al.* [LWC*11] used MAP estimation of human poses in sequential mode to match the control signals obtained from motion sensors, increasing the temporal consistency of the movements. In addition, there are several methods that reconstruct motion streams from sparse representations by retrieving matched motion sequences from a motion capture library [SH08, RTK*15, XWCH15].

6.2. Deep learning methods

The growing popularity of deeply learned (DL) networks, which have been successfully integrated for human pose reconstruction [ST13, YR13, TS14], has prompted researchers to investigate their use as motion controllers. Instead of searching for the closest posture that matches the constraints (i.e. identify full body poses), DL uses neural networks to learn the articulated motion by optimization processes from real movements. In other words, they are used to learn a mapping between high-level control parameters and low-level joint translations and rotations. Convolutional neural networks (CNNs) have been used for physically based simulations, in order to learn motion controllers for articulated characters [TGLT14, MLA*15, DWW15]; these controllers can be parameterized, e.g. setting new targets for the end effectors, to generate movements that satisfy the given constraints. Taking advantage of the growing availability of motion capture data, Allen and Faloutsos [AF09] presented a system that creates physics-based locomotion controllers to naturally simulated characters using automatically evolving networks of connected neurons. Levine and Koltun [LK14] utilized neural networks to learn high-dimensional policies, that are later used to control motions as an optimization over trajectory distributions. More recently, Holden *et al.* [HSK16] presented an entirely machine-learned motion controller, where the embedding of non-linear motion manifolds is learnt using convolutional autoencoders. To map from high-level parameters to the motion manifold, and thus synthesize character movements, the authors integrate a deep neural network, allowing editing of the generated motion by performing optimization in the space of the motion manifold. In this way, it is possible to edit/control motion for style transfer and synthesis, task completion, motion retrieval or motion cleaning [HSKJ15], while ensuring that the edited motion remains plausible. However, they are in general computationally expensive to train; for instance, training a CNN requires a large number of iterations to learn an acceptable controller [Sim94].

7. Hybrid Methods

Hybrid techniques comprise another family of methods that solve the IK problem. In an attempt to reduce the complexity of the optimization problem, they decompose the IK problem into analytical and numerical components [TGB00]. For instance, Lee and Shin [LS99] formulated a hybrid IK approach that combines a numerical optimization technique, where its optimization is based on penalty configuration, with an analytical method that is designed to reduce the burden needed for the numerical optimization. Shin *et al.* [SLSG01] introduced a solver for full-body human puppetry that divides the process into three sub-problems: computing the root position, the body posture and the limb posture. The authors solved each sub-problem independently, employing IK methods specially designed to achieve high performance; given a good estimation of the root position, they used numerical optimization to refine the body posture, and then used inexpensive closed-form solutions to fix limb postures. Kulpa *et al.* [KM05, KMA05] extended the Shin *et al.* and Tolani *et al.* approaches for more natural-looking character poses, by extending the model to encapsulate control of the centre of mass. They readapt pre-recorded animations to satisfy certain constraints and apply these algorithms separately in different body parts such as the head, the two arms, the two legs and the trunk. In similar context, Bouënard *et al.* and Vahrenkamp *et al.* [BGW12, VAD12] divided the problem into different motor tasks that are solved independently using robotics-inspired proportional-derivative controllers.

7.1. Statistical methods

Sequential Monte Carlo methods (SMCM) have also been used for solving IK problems; Courty and Arnaud [CA08] proposed such a solution. Using a sampling approach, the IK problem was solved with FK, hence avoiding the computation of the inverse matrix. The problem is cast as a hidden Markov model (HMM), whose hidden state is given by all the parameters that define the articulated figure. The state space consists of all the possible configurations of the articulated figure. The inverse kinematics is then reformulated in a filtering framework. The proposed SMCM IK solver does not require explicit numerical inversion and joint restrictions can be added to the system in an intuitive manner. These can be easily implemented without the need for complex optimization algorithms.

Particle IK solvers have been implemented in the works of Hecker *et al.* [HRE*08] and Sapra *et al.* [SMM14], which use a body pose goals set and attempt to satisfy the goals by forming a system of constraints over the linked character bodies. Recently, Huang *et al.* [HWF*17] introduced an objective function over a probability density function (PDF), which is built upon multivariate Gaussian distribution models that are learnt from real data to describe natural motion. The authors employed the PDF within a Jacobian framework as soft constraints that control the DoFs of the joints to ensure the smoothness and coherence of the motion in the local joint space.

7.2. Parallel computing

Recently, some effort has been devoted to parallel IK algorithms, which allow solutio of the problem for complex articulated bodies with multiple constraints [AH11]. More specifically, Farzan and DeSouza [FD13] proposed a parallelization method that relies on the computation of multiple numerical estimations of the Jacobian inverse; they handle singularities and multiple paths in redundant robots by selecting the best path to the desired configuration of the end effector on multi-core architectures for the Denavit–Hartenberg representation of robots. Similarly, Harish *et al.* [HMCB16] extend the DLS method to exploit maximum parallelism by mapping the internal DLS steps to the data-parallel GPU architecture. Huang *et al.* [HFDP16] employ a variation of the SDLS method, while the joint motion parameters are learned automatically from pre-captured motion data that are stored in an octree for fast access. Given the end effectors trajectories, a smooth animation is achieved by parallel filtering of the joint information, allowing the constraints to be learnt dynamically and reducing the required computational time.

7.3. Sequential inverse kinematics

Unzueta *et al.* [UPBS08] presented the sequential IK method (SIK), a direct extension of the Boulic *et al.* work [BVU*06]. SIK is an analytic-iterative IK method that reconstructs 3D human full-body movements in real time. The inputs to this method are end effector positions, such as wrists, ankles, head and pelvis (the least possible input in order to be usable within a low-cost motion capture system in real time), which are used to find the human pose. The main idea of SIK is that the reconstruction should be solved sequentially using simple analytic IK algorithms in different parts of the body, in a specific order. SIK starts from the configuration of the spine using a hybrid IK method that uses the positions of the root and the head. Then, using the spine and the known end effector positions, the approach determines the positions and orientations of the clavicles. Finally, using the known end effector positions, an analytic IK method is incorporated to situate each of the limb positions and orientations. In addition, biomechanical limitations are applied to constrain the joints and prevent unnatural movement to ensure visually plausible human poses.

8. Biomechanical constraints

In a redundant system, for example, an articulated figure for which we are seeking IK solutions, it is necessary to incorporate joint restrictions to choose only the appropriate solutions that satisfy the user/model constraints. A variety of different joint and model constraints have been proposed. The first family of constraints is based on types of high-level control, such as the position, orientation, gaze and balance. In the second family of constraints, the goal is defined by the user or automatically depends on the environment, such as in interaction tasks (e.g. floor, reach, grasp), or in collision avoidance. Kinematic chains with multiple end effectors have an additional parameter that needs to be assessed; having multiple tasks may result in conflict between goals, which cannot be achieved simultaneously. Thus, the IK solver should take into account the importance of each task. The importance can be controlled using a priority or a weighted approach, and the IK solver will find different solutions under different case scenarios. The priority approach verifies the most important task first and tries to reach the others only if possible, as in [HYN81] and [BB98], while the weighted approach finds a compromise solution using a weighted sum of all constraints, as in [Gle98].

A joint is defined by its position and orientation and, in the most general case, has 3 DoF. The essential characteristic of a joint is that it permits a relative motion between the two limbs it connects. The most common anthropometric joints are: (1) the *ball-and-socket*, which allows rotary motion in every direction within certain limits, (2) the *hinge*, which permits motion only in one plane about a single axis, (3) the *pivot*, which permits only rotating movement, whereby the axis of the convex articular surface is parallel to the longitudinal axis of the bone, (4) the *condyloid*, which allows biaxial movements, i.e. foreword-backward and side to side, (5) *saddle*, which is similar to the condyloid joint but different angle limits describe the allowable bounds, and (6) the *gliding*, which allows only gliding or sliding movements. In robotics, there is also the *prismatic* joint, which provides single-axis sliding function with the axis of the joint coincident with the centre line of the sliding link.

Most of the existing structure models use techniques which restrict the bone to lie within the rotational and translational limits of the joint. Grassia [Gra98] used a practical parameterization of rotations using the exponential map and compared different parameterizations of rotation; he concluded that the performance of each parameterization depends on the application or the joint model. Technically, it is possible to incorporate constraints on the rotation of a particular joint by directly limiting the Euler angles. However, the results will not be realistic for modelling complex joints or articulated models. Blow [Blo02] proposes a loop hung in space, limiting the range of motion of the bone to “reach windows” described by star polygons. Wilhelms and Van Gelder [WG01], instead of using reach windows, present a 3D “reach cone” methodology using planes, treating the joint limits in the same way as [Blo02]. Korein [Kor85], as well as Baerlocher and Boulic [BB01], parameterize realistic joint boundaries of the ball-and-socket joint by decomposing the arbitrary orientation into two components and control the rotational joint limits with spherical polygons, so they do not exceed their bounds. A similar parameterization to [Kor85] was also used by Unzueta *et al.* [UPBS08], where they modelled swing movements using a spherical parameterization of orientations. Aristidou and Lasenby [AL11] factorized the bone rotation into a rotational and orientational step and added joint constraints by repositioning and reorienting the target to be within the allowable bounds. They later extended their approach to cope with different anthropometric and robotic joints, as well as humanoid models, using a hierarchical framework [ACL16].

For human-like models, as well as for most legged body models, the joints have motion restrictions to keep the movements within a feasible range and prevent unrealistic movements. Several biomechanically and anatomically correct models have been presented that formalize the range of motion of an articulated figure [MB91]. These models are hierarchically structured and are characterized by the number of parameters which describe the motion space. Because of their complex nature, most of the proposed joint models are simplified or approximated by more than one joint. The most well-known models are: *the shoulder model*, a complex model composed of three different joints [MT00, KTL07]; *the spine model*, a complex arrangement of 24 vertebrae (usually, for simplicity, the spine is modelled as a simple chain of joints [Kor85, BPW93]); *the* *hand model*, which is the most versatile part of the body comprising a large number of joints [MSZ94, Ari17]; *the strength model*, which takes into account the forces applied from the skeletal muscles to the bones [BPW93].

Incorporating biomechanical constraints may limit the allowable motion to a smaller set of movements so that the humanoid model avoids unnatural poses. However, it does not prevent the emergence of self-collisions. There is much research devoted to detecting collisions between rigid objects (e.g. [Qui94, GLM96]), deformable objects (e.g. [vdB98, MKE03]) as well as self-collisions in deformable objects, such as Volino and Magnenat-Thalmann [VMT94], James and Pai [JP04] and Govindaraju *et al.* [GKJ*05]. Brown *et al.* [BLM04] presented a self-collision methodology to detect and avoid self-collisions in a knot tying simulation, while recently, Schvartzman *et al.* [SPO10] used the star-shaped property of a polygon to manage self-collisions. Self-collision can also be handled using the penetration depth (PD); Zhang *et al.* [ZKVM07] presented a detailed review for the computation of the PD. The authors conclude that the computation of PD is not sufficient for many applications because it does not take into account the rotational motion. Thus, they introduced a more generalized PD approach that considers possible rotations throughout the path in order to separate the overlapping objects. Nawratil *et al.* [NPR09] defined the generalized PD with respect to a distance metric, allowing efficient computation. However, to handle self-collisions within a humanoid model, it is important to examine limb constraints. This can be done by integrating the self-collision detection step within the IK algorithm, as presented by Unzueta *et al.* [UPBS08], where they estimated PD to prevent the penetration of the elbows in the torso.

Even if joint limits are taken into consideration, unnatural poses may still be obtained. Joint limits provide no information about the most likely pose of a human in motion; they only attempt to reach a given target position and satisfy the rotational and translational limitations. Thus, it is necessary to incorporate model constraints, e.g. a humanoid model, to additionally consider the importance priorities, and physiological constraints of the model. Since IK does not directly address dynamic constraints, such as momentum conservation during ballistic motion, significant efforts have been made to design physically based interactive tools to ensure the production of plausible motions [KSPW04, YLvdP07, LPY16, ATK16]. Physics-based character animation aims at guiding the IK solvers to provide a more natural motion that remains within a feasible set of movements. For instance, Lee and Goswami [LG07] used the momentum and inertia to improve the balance of the animation, Shapiro and Lee [SL11] utilized certain dynamic physical properties, such as the centre of mass and angular momentum, to allow the improvement of unrealistic motions, while Sok *et al.* [SYLH10] have additionally used momentum and force constraints. Inverse kinodynamics (IKD) [KRRS12] is another kinematic workflow that encapsulates short-lived dynamics and allows precise space-time constraints. More recently, Rabbani and Kry [RK16] introduced a method that respects physics during certain dynamic activities by controlling both the centre of mass, and the magnitude of the character's inertia tensor. Finally, muscle-based control methods are also very important to maintain plausible and natural movements [GvdPvdS13]. Nevertheless, another way to address motion naturalness, and ensure the production of plausible movements, is to incorporate data-driven methods, with the cost of having to learn a wide range of complex and dynamic movements through a time-consuming training session.

9. Discussion

IK was commenced in robotics to determine the joint parameters that move each of the robot's end effectors to a desired position. However, in computer graphics and game programming, it has been introduced to deal with entirely different problems, including to efficiently animate 3D articulated subjects, to connect game characters physically to the world, to allow virtual characters to complete specific tasks, for obstacle avoidance, motion synthesis, motion retargeting, contact constraints, etc. In this section, we discuss the performance of numerous IK solvers, suggesting which family of solvers is best suited for particular problems, e.g. human-like or non-anthropomorphic characters, real-time interaction, multi-chain problems, etc. The discussion evaluates the performance of the solvers with regard to their scalability, computational cost, as well as the smoothness of their resulting motion, and additionally their limitations. Finally, we propose future research directions that involve potential advances in IK usage and performance.

The choice of which IK solver should be employed mainly depends on the definition and peculiarities of the problem, and includes several parameters, such as the computational cost (speed of performance, e.g. for real-time interactive applications), the desired smoothness of the final posture (depends on the application e.g. humans, animals or robots), scalability (e.g. the necessity to be easily extendable to different models and to support multiple end effectors) and the possibility of applying priorities, joint and model restrictions. For instance, analytic solvers are best suited for simple case IK problems, such as isolated arm/legs, with maximum 7 DoF, while iterative methods are more general but require multiple steps to converge towards the solution due to the non-linearity of the system. Jacobian methods can be a good choice in cases where biomechanical laws, priorities and weighted constraints are important, while heuristic methods are more attractive if the computational cost is of major significance. The following subsections discuss the performance of each family of methods for particular applications, suggesting which method is more appropriate for different tasks.

9.1. Trends

Figure 4 illustrates a timeline diagram, where key methods have been grouped and sorted in a chronological order. One can observe what methods researchers have focused on in the last few decades, and predict possible future research directions. Since IK was initiated in robotics to move 6R robot manipulators, the analytical methods were among the first to be employed for simple applications in computer graphics; the closed-form solutions have low computational cost and fast convergence rate that are of high importance in motion planning. However, due to their complexity, limited scalability and inability to solve complex kinematic chains, they are not preferred for computer graphics applications. One solution was to move to hybrid methods, such as [TGB00] and [SLSG01], which divide the human skeleton into kinematic parts, and solve each part independently using a

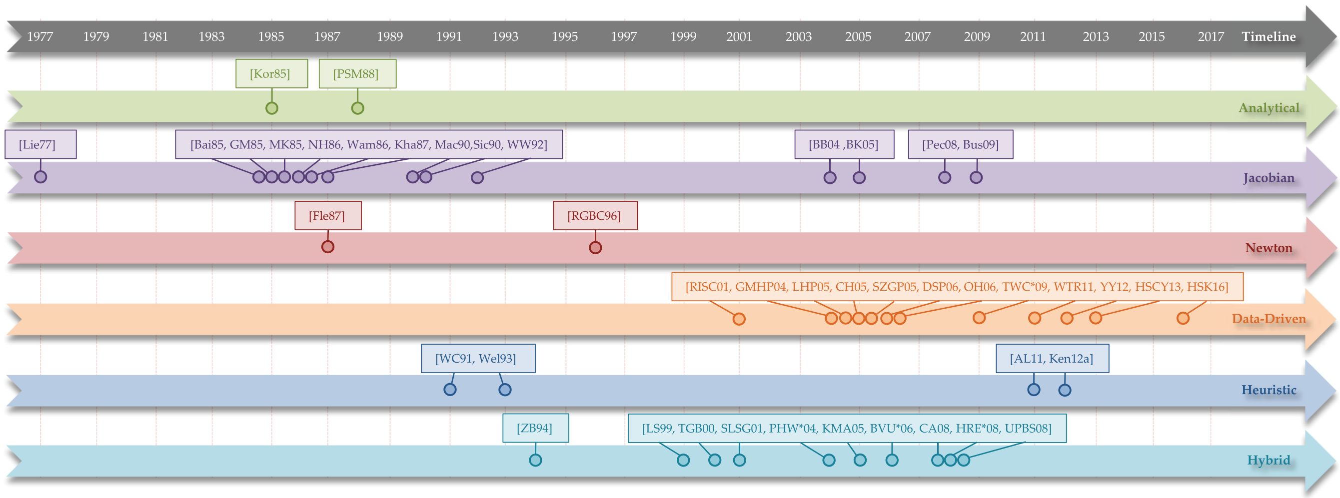

Figure 4: *A timeline showing how the IK approaches have progressed over the years by examining publication years of key methods.*

combination of analytical and numerical methods. Another solution was to use linear approximations to estimate the angle parameters of the manipulator. Thus, the Jacobian methods became popular in the 1980s and early 1990s, and have been extensively studied for over a couple of decades [Bus09]. Currently, they are of less interest in the computer graphics community since modern real-time interactive environments require control of complex multi-chain characters in a fast and realistic way. Nevertheless, some effort has been recently devoted to parallel computation of the Jacobian based methods, e.g. [HMCB16], to gain efficiency and deal with singularities and multiple path problems.

A different way to deal with the IK solver was conceived around 1990s [Lue89, WC91]. The idea was to integrate a heuristic approach, estimating the joint configurations in a simplistic and iterative fashion. This family of methods offers solutions that are simple and computationally efficient. The solver can be easily adapted to different problems, but their ability to produce realistic human motion and their performance in terms of quality are limited.

The need for realistic animations, along with the development of technology in interactive applications, led to the development of methods that combine knowledge in statistical analysis and classification. In the early 2000s, there was a tendency to extend the current solvers with pre-defined hierarchically and/or sequentially structured models, in order to improve their performance with regard to the naturalness of the produced motion. Moreover, they use different solvers for different tasks, employing the most suitable methods for each case to achieve the best possible performance. As a result, hybrid methods are faster, more reliable and can animate complex characters consisting of multiple kinematic chains.

Over the last decade, as demonstrated in the timeline diagram, the trend has shifted towards using data-driven methods, aiming to produce, as much as possible, more natural and realistic movements [GMHP04, WTR11, HSK16]. The widespread use of this family of methods is mainly due to the latest advances in motion capture technology, and the easy availability of feasible movements in motion libraries. Many scholars, to ensure their solver produces plausible and anatomically correct motions, have employed data-driven algorithms, or learnt motion from optimizations through convolution. In this way, the results are guaranteed to be within a feasible set that respects the physiological constraints of the character's movement.

9.2. Human-like characters

Human-like characters are the most common models used in computer graphics. In general, this kind of character consists of a large number of joints (at least 24) that are limited by biomechanical and physiological constraints, ending with up to 70 DoFs. Many solvers have been proposed to handle the high articulation of human movements. The first family of methods used to animate humanoids were the Jacobian-based methods. Jacobian methods can easily handle the high articulation of the human skeleton, but they generally produce oscillating motion with discontinuities and jerky movements, especially when the clamping for the linear approximation allows large steps between the target and the end effector, or when the limbs are near singular postures [UPBS08]. Reducing the amplitude of the clamping steps improves their stability, at the cost of higher computational time. Altogether the time needed to track the target without a jerky convergence is a limitation; for instance, DLS and SDLS methods perform well in terms of jerkiness and singularity issues, but are computationally expensive, and thus less popular in controlling or interacting with human-like characters.

Some effort has been devoted to heuristic, iterative methods to animate human-like characters by tracking a number of control points attached at certain positions, such as the end effectors [CMGM16, ACL16]. The problem with heuristic methods is their limited capability to ensure that spatio-temporal correlations between nearby joints are satisfied. Even if joint constraints are applied, not all generated poses meet the physiological constraints that produce natural human movements. This is mainly due to the fact that the heuristic approaches apply constraints locally, at each iteration, where each joint is treated independently, without offering the potential of incorporating global constraints. A way to deal with the production of unnatural poses from a global perspective is to divide the skeleton into sub-chains and work in a hierarchical and sequential order with priorities, as well as to study the temporal correlations between joints and then allow motion to be further controlled by pull-weights,[6] such as [Roo17]. For instance, in [AZS*17], a number of end effectors control the style of a given motion by minimizing an optimization function that correlates human pose and emotions. For simpler models, such as hands, it is possible to define a well-constrained model that integrates biomechanical constraints and then to apply IK [Ari17].

Hybrid methods that combine numerical and analytic techniques in a hierarchically structured model may be employed for tasks where low computation cost is desired and a well-constrained humanoid model is sought. A detailed evaluation of various numerical and hybrid methods on a humanoid model was presented by Unzueta *et al.* [UPBS08], where the authors compared their SIK method against several variations of the Jacobian (transpose, pseudo-inverse, DLS, SVD-DLS and SDLS), CCD, the Kulpa *et al.* algorithm [KMA05], the Priority IK [PHW*04], and the Tolani *et al.* method [TGB00]. Their main observations were that CCD and the hybrid methods (SIK, Kulpa *et al.* and Tolani *et al.*) are computationally more efficient than other methods, while in terms of reconstruction quality, SIK, Priority IK, Kulpa *et al.* and Tolani *et al.* methods give the best results among the methods used in the evaluation. However, even for well-defined models, it is not straightforward to understand how to apply global limitations, such as physically based or physiological constraints. Thus, there is no guarantee that the solution returned by the solver will be within a feasible set of natural human movements.

The current trend for controlling and animating human-like characters is to use data-driven learnt or deep learning) approaches. Learnt methods select the closest candidate pose, which meets the given constraints, gained from a database [RISC01, GMHP04, WTR11, WC11, HSCY13]; they have the advantage that no constraints are required to be integrated into the solver, since reconstructed poses are matched only to feasible ones. This ensures that the generated pose is natural and satisfies the anatomical and physiological constraints of the human skeleton. However, the data-driven methods require an offline learning stage and the quality of the results depends on the size of the motion database used. In cases where the true pose differs dramatically from the range of postures stored, abnormal solutions may be produced. The large availability of motion clips in recently established databases overcomes this problem, but the choice of the most suitable pose from those available is not trivial. There are many other factors that affect the performance of data-driven methods, such as a mechanism to ensure the temporal consistency and the smooth transition between the selected poses [CH05, TZK*11]. In addition, there is no guarantee that the best candidate retrieved from the matched poses does not contain errors due to bad capturing. On the other hand, deep learning methods, e.g. [HSK16], can deal with the error issues; they use pre-captured motions to learn motion manifolds through convolution, smoothing the errors, thus reconstructing natural and plausible movements. However, similarly to the learnt methods, learning a CNN model is time-consuming, the quality of the resulting motion is correlated with the amount of data used for training. When motion is smoothed through the convolution process, it interpolates fine details of the movement. Nevertheless, both learnt and deep learning methods seem currently to be among the most popular approaches for controlling human pose and reconstructing motion from sparse data [SH08, KTWZ10, LWC*11, RTK*15, XWCH15, HSKJ15] as well as for animating highly articulated human parts (e.g. hands) [dLGPF08, OKA11, LYTZ13].

Even when working on human-like characters, data-driven methods are not always suitable. There are instances where directors desire their characters to have a cartoony look and behaviour (e.g. in computer animated films such as Shrek), or superhero powers. Such movements cannot be captured on a large scale so as to establish a convenient library of movement to use for retrieving the desired pose. In such cases, it is necessary to consult alternative methods by defining well-constrained hybrid solutions that operate in a hierarchical and sequential fashion. In addition, data-driven methods are not computationally efficient enough for real-time interaction.

9.3. Non-anthropomorphic characters or objects

Non-anthropomorphic characters contain articulations that are difficult to match with human data (e.g. snakes, insects, multi-legged animals, dragons), as well as other objects that allow some degrees of articulation (e.g. a rope). It is extremely difficult to create large databases that consist of all feasible movements for each individual creature or object; in some cases, it is even challenging to capture the subject (e.g. insects), or there are no similar real creatures in nature (e.g. virtual characters in cartoons). This is also valid for virtual human-like creatures that have extra articulated parts (e.g., a tail). Thus, data-driven methods seem to be less convenient methods to use in these cases.

The best way to deal with non-anthropomorphic objects is to integrate the most suitable method for the specific problem; different articulations can be treated differently, based on the properties of the kinematic chain(s). When applicable, the problem can be divided into parts and the most appropriate method can be applied in a hierarchical and sequential fashion. Hybrid methods, such as [TGB00, SLSG01, KMA05] and [UPBS08], can be adjusted to deal with cartoony and monster-looking characters. On the other hand, heuristic methods offer smooth looking motion in more simplistic problems. For instance, since FABRIK is computationally efficient, flexible enough to be adapted to different models, and able to control multiple end effectors, it seems to be a good choice for animating, in real time, single kinematic chains with limited or no constraints (e.g. tails), or kinematic chains with multiple end effectors (e.g. insects). CCD is also very popular for single kinematic chains due to its efficiency; however, it suffers from poor motion distribution, while the rolling and unrolling of the chain before reaching the target can lead to unnatural poses. The Jacobian methods are also effective in animating single and multiple end effectors; for instance, Li *et al.* [LABK17] designed cable-driven mechanisms to perform

[6]Pull-weights allow us to distribute the joint modification along the kinematic chain, taking into consideration spatial and temporal correlations between nearby joints.

push and pick-and-place tasks using a Jacobian-based morphology optimization process. However, they generally converge slowly since they use a linear approximation with a small step (changing the step size may result in undesirable oscillations and discontinuities). The approximation used in SVD-DLS converges slightly faster than its Jacobian counterparts, but the Jacobian transpose and SDLS returns the smoothest motion.

For the production of natural animation and locomotion for non-anthropometric characters, the design of the model's skeleton and the IK selection should take into consideration the morphology and musculoskeletal structure of the creature. There are many papers that deal with the animation of such complex creatures, e.g. bipedals [GvdPvdS13], quadrupeds [KRFC09, CKJ*11], insects [FJT13] or the swimming behaviour for given articulated creatures [TGTL11], which are mainly designed via incorporating a physically based Jacobian IK solver. For more complex structures, which consist of many articulated parts, the use of hybrid methods is recommended.

Other factors that are important for non-anthropomorphic characters are scalability and handling of multiple end effectors. Scalability is usually measured by how general the system can be made, and its capability to be adjusted and customized to cope with different problems; the computational time needed to solve problems with large kinematic chains, and the resulting accuracy, play an important role for this evaluation. Fêdor [F03̂] explores the balance between speed, accuracy and scalability in a number of methods. More specifically, the analytical methods tend to suffer from poor scalability; slight changes in the problem or the model used require complete re-design of the solution. Similarly, even though the data-driven methods can be efficiently down-scaled to partitions, their capability to be up-scaled or adapted into models with different skeletal structure is limited and highly dependent on the training data. For models with different structure, new training sessions must be performed. On the other hand, the Jacobian and Newton methods can be easily adjusted to problems with different kinematic chains, but the time required for convergence increases as the number of joints and DoFs grows. Similarly, FABRIK is easily adaptable to models with different structures and multiple end effectors; unlike CCD whose performance and convergence speed are reduced, FABRIK's computational time changes little when the kinematic chain increases. Finally, scaling hybrid methods, even though not straightforward, can be used by re-defining and re-adjusting the model's structure, priorities and hierarchies.

9.4. Future directions

IK is a well-known problem that has challenged researchers for many years. Nevertheless, the fast evolving technology in computer graphics and the increasing use of virtual characters in interactive and entertainment applications create challenging research questions that need to be studied/answered.

Motion capture technology has been used extensively to capture the movement and portray it as a 3D virtual representation. However, one of the main limitations of this technology is that only realistic motions can be captured, while directors sometimes want their characters to have a non-realistic look and behaviour. In this context, IK techniques can be employed for partial-body motion synthesis, so as to create new actions from existing movements. In addition, they can be incorporated to efficiently synthesize movements, avoiding common problems such as foot skating and oscillations. While motion capture is a useful tool for 3D animation, sometimes, it does not give the director enough control over the subtleties of an animation. For example, it is not possible to add more expression to a pre-captured motion. The ability to automatically adapt captured movements into characters with different style, behaviour, gender, age, etc., is an important aspect that needs further investigation; therefore, IK algorithms should be improved to offer directors a greater degree of flexibility and the ability to control stylistic variations of the animation (e.g. [HSK16]).

One of the most significant applications of IK is real-time skeletal re-targeting, which is essential for mapping movements captured from one character onto another with different proportions (taller, smaller, longer legs, etc.). Most re-targeting techniques suffer from flying, penetrating and skating due to the differences in the skeletal configurations and/or bone lengths. In addition, motion re-targeting must ensure that contact constraints are satisfied. In this direction, it is essential to study methods that automatically adapt skeletons undergoing complex deformations into different input meshes, preserving the essence of motion naturally and without oscillations (e.g. [MGDB17]). They should also be able to re-establish violated constraints that may occur. Future research directions should focus on achieving a smooth transition between skeletons with fundamentally different joint constraints (e.g. biped to quadruped), entirely different skeletal structure, different body proportions, way of movement and behaviour. Another way to address the design of efficient locomotion controllers, which requires no other *a priori* information other than the mechanical structure of the creature, is deep learning; a great effort has been devoted in this direction over the last few years [PBYvdP17, HKS17].

More consideration should also be given to ways of incorporating more anatomical and physiological constraints, aiming to limit the resulting postures to a feasible set and allowing physically based animations (e.g. [RK16]). Further investigation is also needed to consider small changes in limb size (e.g. by adding a spring/mass model), in addition to biomechanical laws, such as the force and energy needed to accomplish a task, with the intention of adding extra realism in human animation. Nevertheless, there is also a question of how much the chain can be expanded or shrunk (keeping the human anatomy and constraints) without the user perceiving the change [HRvdP04].

Komura *et al.* [KSK01] has solved the IK redundancy by using a criterion of minimum muscle-force change; however, the effect of muscles, different passive torques, etc., has not yet been fully studied. IK methods must be advanced in order to deal with human body shapes, taking into consideration anatomy principles and musculoskeletal models. Recent works on skeletal muscle and subcutaneous fat growth, such as [SZK15], have revealed new challenges which need to be dealt with. Muscle and fat modelling requires computation and anatomical effort to keep the appropriate deformation of the body shape; when the muscle and fat quantity change not only are the volumetric and surface models affected, but also the human skeletal structure and the movement style.

10. Conclusion

In this paper, we have surveyed some of the most popular IK solvers, emphasizing, but not limiting, the discussion to approaches from a computer graphics point of view. We describe where the research in IK has been focused in the past, how it has progressed over the years and indicate reasons for such progression. The main scope of this survey is to offer a guide that highlights the advantages and disadvantages of key IK solvers, giving indications about which method is best suited to solve different problems. It aims to introduce IK to new researchers that aim to optimize their IK-based projects. Finally, it provides future directions to extend current limitations, and research challenges that need further investigation.

Acknowledgements

This work is co-financed by the European Regional Development Fund and the Republic of Cyprus through the Research Promotion Foundation under contract DIDAKTOR/0311/73. A. Aristidou and A. Shamir are funded from ISF - NSFC grant number 2216/15.

References

[ACL16] Aristidou A., Chrysanthou Y., Lasenby J.: Extending FABRIK with model constraints. *Computer Animation and Virtual Worlds 27*, 1 (2016), 35–57.

[AF09] Allen B. F., Faloutsos P.: Evolved controllers for simulated locomotion. In *MIG '09: Proceedings of the 2nd International Workshop on Motion in Games* (Berlin, Heidelberg, 2009), Springer-Verlag, pp. 219–230.

[AH11] Aguilar O. A., Huegel J. C.: Inverse kinematics solution for robotic manipulators using a cuda-based parallel genetic algorithm. In *MICAI'11: Proceedings of the 10th Mexican International Conference on Advances in Artificial Intelligence - Volume Part I* (Berlin, Heidelberg, 2011), Springer-Verlag, pp. 490–503.

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

[Ari17] Aristidou A.: Hand tracking with physiological constraints. *The Visual Computer* (2017). https://doi.org/10.1007/s00371-016-1327-8.

[ATK16] Andrews S., Teichmann M., Kry P.: Blended linear models for reduced compliant mechanical systems. *IEEE Transactions on Visualization and Computer Graphics 22*, 3 (2016), 1209–1222.

[AvdP16] Agrawal S., van de Panne M.: Task-based locomotion. *ACM Transactions on Graphics 35*, 4 (July 2016), 82:1–82:11.

[AZS*17] Aristidou A., Zeng Q., Stavrakis E., Yin K., Cohen-Or D., Chrysanthou Y., Chen B.: Emotion control of unstructured dance movements. In *SCA '17: Proceedings of the 16th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (NY, USA, 2017), ACM.

[Bai85] Baillieul J.: Kinematic programming alternatives for redundant manipulators. In *Proceedings of the IEEE International Conference on Robotics and Automation* (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 of the International Conference on Intelligent Robots and Systems* (October 1998), vol. 1, pp. 323–329.

[BB01] Baerlocher P., Boulic B.: Parametrization and range of motion of the ball-and-socket joint. In *Deformable Avatars*. Kluwer Academic Publishers, Dordrecht, The Netherlands (2001), pp. 180–190.

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

[BGG93] Bullock D., Grossberg S., Guenther F. H.: A self-organizing neural model of motor equivalent reaching and tool use by a multijoint arm. *Journal of Cognitive Neuroscience 5*, 4 (1993), 408–435.

[BGQHA13] Bensalah C., Gonzalez-Quijano J., Hendrich N., Abderrahim M.: Anthropomorphic robotics hand inverse kinematics using estimated SVD in an extended SDLS approach. In *Proceedings of International Conference on Advanced Robotics (ICAR)* (November 2013), pp. 1–7.

[BGW12] Bouënard A., Gibet S., Wanderley M. M.: Hybrid inverse motion control for virtual characters interacting with sound synthesis: Application to percussion motion. *The Visual Computer 28*, 4 (April 2012), 357–370.

[BK05] Buss S. R., Kim J.-S.: Selectively damped least squares for inverse kinematics. *Journal of Graphics Tools 10*, 3 (2005), 37–49.

[BK07] Boulic R., Kulpa R.: Inverse kinematics and kinetics for virtual humanoids. In *Proceedings of Tutorial Eurographics*, 2007.

[BLM04] Brown J., Latombe J.-C., Montgomery K.: Real-time knot-tying simulation. *The Visual Computer 20*, 2 (2004), 165–179.

[Blo02] Blow J.: Inverse kinematics with quaternion joint limits. *Game Developer Magazine* (April 2002), 16–18.

[BM96] Boulic R., Mas R.: Hierarchical kinematic behaviors for complex articulated figures. In *Interactive Computer Animation*. N. Magnenat Thalmann and D. Thalmann (Eds.), Prentice-Hall, Inc., Upper Saddle River, NJ (1996), pp. 40–70.

[BMT95] Boulic R., Mas R., Thalmann D.: Inverse kinetics for center of mass position control and posture optimization. In *Image Processing for Broadcast and Video Production*. Y. Paker and S. Wilbur (Eds.), Workshops in Computing. Springer, London (1995), pp. 234–249.

[BPW93] Badler N. I., Phillips C. B., Webber B. L.: *Simulating Humans: Computer Graphics Animation and Control*. Oxford University Press, New York, Oxford, 1993.

[Bus03] Buss S. R.: *3-D Computer Graphics: A Mathematical Introduction with OpenGL*. Cambridge University Press, Cambridge, 2003.

[Bus09] Buss S. R.: *Introduction to Inverse Kinematics with Jacobian Transpose, Pseudoinverse and Damped Least Squares methods*. Tech. rep., University of California, 2009.

[BVU*06] Boulic R., Varona J., Unzueta L., Peinado M., Suescun A., Perales F.: Evaluation of on-line analytic and numeric inverse kinematics approaches driven by partial vision input. *Virtual Reality 10*, 1 (2006), 48–61.

[CA08] Courty N., Arnaud E.: Inverse kinematics using sequential Monte Carlo methods. In *Proceedings of the V Conference on Articulated Motion and Deformable Objects, AMDO'08* (Mallorca, Spain, 2008), vol. 5098, LNCS, pp. 1–10.

[CBT07] Carvalho S. R., Boulic R., Thalmann D.: Interactive low-dimensional human motion synthesis by combining motion models and PIK. *Computer Animation and Virtual Worlds 18, 4–5* (2007), 493–503.

[CD95] Chan T. F., Dubey R.: A weighted least-norm solution based scheme for avoiding joint limits for redundant joint manipulators. *IEEE Transactions on Robotics and Automation 11*, 2 (April 1995), 286–292.

[CH05] Chai J., Hodgins J. K.: Performance animation from low-dimensional control signals. *ACM Transactions on Graphics 24*, 3 (July 2005), 686–696.

[CK99] Choi K.-J., Ko H.-S.: On-line motion retargetting. In *Proceedings of the 17th Pacific Conference on Computer Graphics and Applications* (Seoul, South Korea, 1999), IEEE, pp. 32–42.

[CKJ*11] Coros S., Karpathy A., Jones B., Reveret L., van de Panne M.: Locomotion skills for simulated quadrupeds. *ACM Transactions on Graphics 30*, 4 (July 2011), 59:1–59:12.

[CMGM16] Carreno-Medrano P., Gibet S., Marteau P.-F.: From expressive end-effector trajectories to expressive bodily motions. In *CASA '16: Proceedings of the International Conference on Computer Animation and Social Agents* (NY, USA, 2016), ACM, pp. 157–163.

[Cra03] Craig J. J.: *Introduction to Robotics: Mechanics and Control* (3rd edition). Addison-Wesley Longman Publishing Co., Reading, MA, 2003.

[CT12] Colomè A., Torras C.: Redundant Inverse Kinematics: Experimental comparative review and two enhancements. In *Proceedings of IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (October 2012), pp. 5333–5340.

[CvKM97] Chin K. W., von Konsky B. R., Marriott A.: Closed-form and generalized inverse kinematics solutions for the analysis of human motion. In *Proceedings of the IEEE Conference of Engineering in Medicine and Biology Society* (1997), vol. 5, pp. 1911–1914.

[CW93] Chen C., Walker I.: A consistent null-space based approach to inverse kinematics of redundant robots. In *Proceedings of the IEEE International Conference on Robotics and Automation* (May 1993), vol. 3, pp. 374–381.

[Dia10] Diankov R.: *Automated Construction of Robotic Manipulation Programs*. PhD thesis, Carnegie Mellon University, Robotics Institute, August 2010.

[dLGPF08] de La Gorce, M., Paragios N., Fleet D. J.: Model-based hand tracking with texture, shading and self-occlusions. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (June 2008), pp. 1–8.

[DSP06] Der, K. G., Sumner R. W., Popović J.: Inverse kinematics for reduced deformable models. *ACM Transactions on Graphics 25*, 3 (July 2006), 1174–1179.

[DVS01] D'Souza A., Vijayakumar S., Schaal S.: Learning inverse kinematics. In *Proceedings of the International Conference on Intelligent Robots and Systems* (Maui, Hawaii, USA, 2001), vol. 1, pp. 298–303.

[DWW15] Du, Y., Wang W., Wang L.: Hierarchical recurrent neural network for skeleton based action recognition. In *CVPR'15: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition* (June 2015), pp. 1110–1118.

[F03] Fêdor M.: Application of inverse kinematics for skeleton manipulation in real-time. In *Proceedings of the Spring Conference on Computer Graphics, SCCG '03* (NY, USA, 2003), ACM, pp. 203–212.

[FD13] Farzan S., DeSouza G.: From d-h to inverse kinematics: A fast numerical solution for general robotic manipulators using parallel processing. In *Proceedings of the International Conference on Intelligent Robots and Systems (IROS)* (November 2013), pp. 2507–2513.

[FDLK12] Flacco F., De Luca A., Khatib O.: Motion control of redundant robots under joint constraints: Saturation in the null space. In *Proceedings of the IEEE International Conference on Robotics and Automation* (May 2012), ICRA, pp. 285–292.

[FJT13] Fang J., Jiang C., Terzopoulos D.: Modeling and animating myriapoda: A real-time kinematic/dynamic approach. In *SCA'13: Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (NY, USA, 2013), ACM, pp. 203–212.

[Fle87] Fletcher R.: *Practical Methods of Optimization* (2nd edition). Wiley-Interscience, NY, 1987.

[GKJ*05] Govindaraju N. K., Knott D., Jain N., Kabul I., Tamstorf R., Gayle R., Lin M. C., Manocha D.: Interactive collision detection between deformable models using chromatic decomposition. *ACM Transactions on Graphics 24*, 3 (2005), 991–999.

[Gle98] Gleicher M.: Retargetting motion to new characters. In *SIGGRAPH '98: Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques* (NY, USA, 1998), ACM, pp. 33–42.

[GLM96] Gottschalk S., Lin M. C., Manocha D.: OBBTree: A hierarchical ttructure for rapid interference detection. In *SIGGRAPH '96: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (NY, USA, 1996), ACM, pp. 171–180.

[GM85] Girard M., Maciejewski A. A.: Computational modeling for the computer animation of legged figures. In *Proceedings of SIGGRAPH '85* (NY, USA, 1985), ACM, pp. 263–270.

[GMHP04] Grochow K., Martin S. L., Hertzmann A., Popović Z.: Style-based inverse kinematics. *ACM Transactions on Graphics 23*, 3 (August 2004), 522–531.

[GORH05] Gan J. Q., Oyama E., Rosales E. M., Hu H.: A complete analytical solution to the inverse kinematics of the pioneer 2 robotic arm. *Robotica 23* (1 2005), 123–129.

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

[GvdPvdS13] Geijtenbeek T., van de Panne M., van der Stappen A. F.: Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics 32*, 6 (November 2013), 206:1–206:11.

[HFDP16] Huang J., Fratarcangeli M., Ding Y., Pelachaud C.: Inverse kinematics using dynamic joint parameters: Inverse kinematics animation synthesis learnt from sub-divided motion micro-segments. *The Visual Computer 33*, 12 (December 2017), 1541–1553.

[HK10] Huang Y., Kallmann M.: Motion parameterization with inverse blending. In *MIG'10: Proceedings of the 3rd International Conference on Motion in Games* (Berlin, Heidelberg, 2010), Springer-Verlag, pp. 242–253.

[HKS17] Holden D., Komura T., Saito J.: Phase-functioned neural networks for character control. *ACM Transactions on Graphics 36*, 4 (July 2017).

[HKW97] Henrich D., Karl J., Wörn H.: *A Review of Parallel Processing Approaches to Robot Kinematics and Jacobian*. Tech. Rep. 10/97, University of Karlsruhe, 1997.

[HMC*15] Hahn F., Mutzel F., Coros S., Thomaszewski B., Nitti M., Gross M., Sumner R. W.: Sketch abstractions for character posing. In *SCA '15: Proceedings of the 14th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (NY, USA, 2015), ACM, pp. 185–191.

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

[HP12] Huang J., Pelachaud C.: An efficient energy transfer inverse kinematics solution. In *Proceedings of Motion in Games (MIG)* (2012). M. Kallmann and K. E. Bekris (Eds.), vol. 7660 of Lecture Notes in Computer Science, Springer, pp. 278–289.

[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 (2008), 1–11.

[HRvdP04] Harrison J., Rensink R. A., van de Panne M.: Obscuring length changes during animated motion. *ACM Transactions on Graphics 23*, 3 (August 2004), 569–573.

[HSCY13] Ho E. S. L., Shum H. P. H., Cheung Y.-m., Yuen P. C.: Topology aware data-driven inverse kinematics. *Computer Graphics Forum 32*, 7 (October 2013), 61–70.

[HSK16] Holden D., Saito J., Komura T.: A deep learning framework for character motion synthesis and editing. *ACM Transactions on Graphics 35*, 4 (July 2016), 138:1–138:11.

[HSKJ15] Holden D., Saito J., Komura T., Joyce T.: Learning motion manifolds with convolutional autoencoders. In *SA '15: SIGGRAPH Asia 2015 Technical Briefs* (NY, USA, 2015), ACM, pp. 18:1–18:4.

[HWF*17] Huang J., Wang Q., Fratarcangeli M., Yan K., Pelachaud C.: Multi-variate gaussian-based inverse kinematics. *Computer Graphics Forum 36*, 8 (December 2017), 418–428.

[HYN81] Hanafusa H., Yoshikawa T., Nakamura Y.: Analysis and control of articulated robot with redundancy. In *Proceedings of the 8th Triennal World Congress* (1981), pp. 1927–1932.

[JP04] James D. L., Pai D. K.: Bd-tree: Output-sensitive collision detection for reduced deformable models. *ACM Transactions on Graphics 23*, 3 (2004), 393–398.

[JR92] Jordan M. I., Rumelhart D. E.: Forward models: Supervised learning with a distal teacher. *Cognitive Science 16*, 3 (1992), 307–354.

[Kal08] Kallmann M.: Analytical inverse kinematics with body posture control. *Comp. Animation & Virtual Worlds 19*, 2 (2008), 79–91.

[Ken12a] Kenwright B.: Inverse kinematics - Cyclic coordinate descent (CCD). *Journal of Graphics Tools 16*, 4 (2012), 177–217.

[Ken12b] Kenwright B.: Real-time character inverse kinematics using the gauss-seidel iterative approximation method. In *Proceedings of the 4th International Conference on Creative Content Technologies* (Nice, France, July 2012), IARIA, pp. 63–68.

[Ken13] Kenwright B.: Inverse kinematics with dual-quaternions, exponential-maps, and joint limits. *International Journal on Advances in Intelligent Systems 6*, 1 and 2 (2013), 53–65.

[Kha87] Khatib O.: A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal of Robotics and Automation 3*, 1 (1987), 43–53.

[KLC*03] Kim J. O., Lee B. R., Chung C. H., Hwang J., Lee W.: The inductive inverse kinematics algorithm to manipulate the posture of an articulated body. In *Proceedings of the International Conference on Computational Science* (June 2–4, 2003), vol. 2657 of ICCS 2003, Springer, pp. 305–313.

[KLW11] Kanoun O., Lamiraux F., Wieber P.-B.: Kinematic control of redundant manipulators: Generalizing the task-priority framework to inequality task. *IEEE Transactions on Robotics 27*, 4 (August 2011), 785–792.

[KM05] Kulpa R., Multon F.: Fast inverse kinematics and kinetics solver for human-like figures. In *Proceedings of International Conference on Humanoid Robots, IEEE-RAS* (December 2005), pp. 38–43.

[KMA05] Kulpa R., Multon F., Arnaldi B.: Morphology independent representation of motions for interactive human-like animation. *Computer Graphics Forum 24*, 3 (2005), 343–352.

[KO13] Kim I., Oh J.-H.: Inverse kinematic control of humanoids under joint constraints. *International Journal of Advanced Robotic Systems 10* (2013), 1–12.

[Kor85] Korein J. U.: *A Geometric Investigation of Reach*. MIT Press, Cambridge, MA, 1985.

[KRFC09] Kry P. G., Reveret L., Faure F., Cani M.-P.: Modal locomotion: Animating virtual characters with natural vibrations. *Computer Graphics Forum 28*, 2 (2009), 289–298.

[KRRS12] Kry P. G., Rahgoshay C., Rabbani A., Singh K.: Inverse kinodynamics: Editing and constraining kinematic approximations of dynamic motion. *Computers & Graphics 36*, 8 (2012), 904–915. Graphics Interaction Virtual Environments and Applications 2012.

[KSG02] Kovar L., Schreiner J., Gleicher M.: Footskate cleanup for motion capture editing. In *SCA '02: Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (NY, USA, 2002), ACM, pp. 97–104.

[KSK01] Komura T., Shinagawa Y., Kunii T. L.: An inverse kinematics method based on muscle dynamics. In *CGI '01: Computer Graphics International* (Washington, DC, USA, 2001), IEEE Computer Society, pp. 15–22.

[KSPW04] Khatib O., Sentis L., Park J., Warren J.: Whole-body dynamic behavior and control of human-like robots. *International Journal of Humanoid Robotics 1*, 1 (2004), 29–43.

[KTL07] Kloṕcar N., Tomšič M., Lenarčič J.: A kinematic model of the shoulder complex to evaluate the arm-reachable workspace. *Journal of Biomechanics 40*, 1 (2007), 86–91.

[KTWZ10] Krüger B., Tautges J., Weber A., Zinke A.: Fast local and global similarity searches in large motion capture databases. In *SCA '10: Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, 2010), Eurographics Association, pp. 1–10.

[LABK17] Li J., Andrews S., Birkas K. G., Kry P. G.: Task-based design of cable-driven articulated mechanisms. In *SCF '17: Proceedings of the 1st Annual ACM Symposium on Computational Fabrication* (NY, USA, 2017), ACM, pp. 6:1–6:12.

[Lan98] Lander J.: Making kine more flexible. *Game Developer 5*, 3 (1998), 15–22.

[LG07] Lee S.-H., Goswami A.: Reaction mass pendulum (rmp): An explicit model for centroidal angular momentum of humanoid robots. In *Proceedings of 2007 IEEE International Conference on Robotics and Automation* (April 2007), pp. 4667–4672.

[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.

[Lie77] Liegeois A.: Automatic supervisory control of the configuration and behavior of multibody mechanisms. *IEEE Transactions on Systems, Man and Cybernetics 7*, 12 (December 1977), 868–871.

[LK14] Levine S., Koltun V.: Learning complex neural network policies with trajectory optimization. In *ICML'14: Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32* (2014), JMLR.org, pp. II-829–II-837.

[LPY16] Liu L., Panne M. V. D., Yin K.: Guided learning of control graphs for physics-based characters. *ACM Transactions on Graphics 35*, 3 (May 2016), 29:1–29:14.

[LS99] Lee J., Shin S. Y.: A hierarchical approach to interactive motion editing for human-like figures. In *Proceedings of the 26th Annual Conference on Computer graphics And Interactive Techniques* (NY, USA, 1999), SIGGRAPH '99, pp. 39–48.

[Lue89] Luenberger D. G.: *Linear and Nonlinear Programming*. Addison Wesley, Reading, MA, 1989.

[LVSF16] Lansley A., Vamplew P., Smith P., Foale C.: Caliko: An inverse kinematics software library implementation of the FABRIK algorithm. *Journal of Open Research Software 4*, 1 (2016), e36–e36.

[LWC*11] Liu H., Wei X., Chai J., Ha I., Rhee T.: Realtime human motion control with a small number of inertial sensors. In *I3D '11: Proceedings of the Symposium on Interactive 3D Graphics and Games* (NY, USA, 2011), ACM, pp. 133–140.

[LYTZ13] Liang H., Yuan J., Thalmann D., Zhang Z.: Model-based hand pose estimation via spatial-temporal hand parsing and 3d fingertip localization. *The Visual Computer 29*, 6–8 (June 2013), 837–848.

[Mac90] Maciejewski A. A.: Dealing with ill-conditioned equations of motion for articulated figures. *IEEE Computer Graphics and Applications 10*, 3 (1990), 63–71.

[MB91] Monheit G., Badler N.: A kinematic model of the human spine and torso. *IEEE Computer Graphics and Applications 11*, 2 (March 1991), 29–38.

[MB13] Molla E., Boulic R.: Singularity free parametrization of human limbs. In *MIG '13: Proceedings of Motion on Games* (NY, USA, 2013), ACM, pp. 187–196.

[MBBT00] Monzani J.-S., Baerlocher P., Boulic R., Thalmann D.: Using an intermediate skeleton and inverse kinematics for motion retargeting. *Computer Graphics Forum 19*, 3 (2000), 11–19.

[MC94] Manocha D., Canny J. F.: Efficient inverse kinematics for general 6r manipulators. *IEEE Transactions on Robotics and Automation 10*, 5 (October 1994), 648–657.

[MC13] Moya S., Colloud F.: A fast geometrically-driven prioritized inverse kinematics solver. In *ISB'13: Proceedings of the Congress of the International Society of Biomechanics* (August 2013), pp. 1–3.

[MD04] Merrick D., Dwyer T.: Skeletal animation for the exploration of graphs. In *Proceedings of Australasian Symposium on Information Visualisation (invis.au'04)* (Christchurch, New Zealand, 2004), N. Churcher and C. Churcher (Eds.), vol. 35 of CRPIT, ACS, pp. 61–70.

[MGDB17] Molla E., Galvan Debarba H., Boulic R.: Egocentric mapping of body surface constraints. *IEEE Transactions on Visualization and Computer Graphic 99* (2017), https://doi.org/10.1109/TVCG.2017.2708083.

[MK85] Maciejewski A. A., Klein C. A.: Obstacle avoidance for kinematically redundant manipulators in dynamically varying environments. *The International Journal of Robotics Research 4*, 3 (1985), 109–117.

[MK11] Mahmudi M., Kallmann M.: Feature-based locomotion with inverse branch kinematics. In *MIG'11: Proceedings of the 4th International Conference on Motion in Games* (Berlin, Heidelberg, 2011), Springer-Verlag, pp. 39–50.

[MKE03] Mezger J., Kimmerle S., Etzmuss O.,: Hierarchical techniques in collision detection for cloth animation. *Journal of WSCG 11* (2003), 322–329.

[MLA*15] Mordatch I., Lowrey K., Andrew G., Popovic Z., Todorov E.: Interactive control of diverse complex characters with neural networks. In *NIPS'15: Proceedings of the 28th International Conference on Neural Information Processing Systems* (Cambridge, MA, USA, 2015), MIT Press, pp. 3132–3140.

[MM04] Meredith M., Maddock S.: Using a half-jacobian for real-time inverse kinematics. In *Proceedings of the 5th International Conference on Computer Games: Artificial Intelligence, Design and Education* (2004), pp. 81–88.

[MM05] Meredith M., Maddock S.: Adapting motion capture data using weighted real-time inverse kinematics. *Computers in Entertainment 3*, 1 (2005), 1–15.

[MSZ94] Murray R. M., Sastry S. S., Zexiang L.: *A Mathematical Introduction to Robotic Manipulation*. CRC Press, Inc., Boca Raton, FL, 1994.

[MT00] Maurel W., Thalmann D.: Human shoulder modeling including scapulo-thoracic constraint and joint sinus cones. *Computers & Graphics 24*, 2 (2000), 203–218.

[Muk12] Mukundan R.: *Advanced Methods in Computer Graphics: With Examples in OpenGL*. Springer, New York, 2012.

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

[NHY87] Nakamura Y., Hanafusa H., Yoshikawa T.: Task-priority based redundancy control of robot manipulators. *The International Journal of Robotics Research 6*, 2 (1987), 3–15.

[NPR09] Nawratil G., Pottmann H., Ravani B.: Generalized penetration depth computation based on kinematical geometry. *Computer Aided Geometric Design 26*, 4 (May 2009), 425–443.

[NW99] Nocedal J., Wright S. J.: *Numerical Optimization*. Springer, Berlin/New York, August 1999.

[OCA*01] Oyama E., Chong N. Y., Agah A., Maeda T., Tachi S.: Inverse kinematics learning by modular architecture neural networks with performance prediction networks. In *Proceedings of the IEEE International Conference on Robotics and Automation* (Seoul, Korea, 2001), vol. 1, pp. 1006–1012.

[OH06] Ong E.-J., Hilton A.: Learnt inverse kinematics for animation synthesis. *Graphical Models 68*, 5 (2006), 472–483.

[OKA11] Oikonomidis I., Kyriazis N., Argyros A.: Efficient model-based 3d tracking of hand articulations using kinect. In *BMVA '11: Proceedings of the British Machine Vision Conference* (2011), BMVA Press, pp. 101.1–101.11.

[OS84] Orin D. E., Schrader W. W.: Efficient computation of the Jacobian for robot manipulators. *The International Journal of Robotics Research 3*, 4 (1984), 66–75.

[PBYvdP17] Peng X. B., Berseth G., Yin K., van de Panne M.: Deeploco: Dynamic locomotion skills using hierarchical deep reinforcement learning. *ACM Transactions on Graphics 36*, 4 (July 2017).

[Pec08] Pechev A. N.: Inverse kinematics without matrix inversion. In *Proceedings of the IEEE International Conference on Robotics and Automation* (Pasadena, CA, USA, May 19–23, 2008), pp. 2005–2012.

[PHW*04] Peinado M., Herbelin B., Wanderley M., Callennec B. L., Boulic R., Thalmann D., Méziat D.: Towards configurable motion capture with prioritized inverse kinematics. In *Proceedings of third International Workshop on Virtual Rehabilitation* (Laussane, Switzerland, 2004).

[PR13] Poddighe R., Roos N.: A NAO robot paying tic-tactoe: Comparing alternative methods for Inverse Kinematics. In *BNAIC'13: Proceedings of the 25th Belgium-Netherlands Artificial Intelligence Conference* (November 2013).

[PSM88] Paul R. P., Shimano B., Mayer G. E.: Kinematic control equations for simple manipulators. *IEEE Transactions on System, Man and Cybernetics 11*, 6 (1988), 449–455.

[PTWF92] Press W. H., Teukolski S. A., Wetterling W. T., Flannery B. P.: *Numerical Recipees in C* (2nd edition). Cambridge Press, Cambridge, 1992.

[Qui94] Quinlan S.: Efficient distance computation between nonconvex objects. In *Proceedings of the IEEE International Conference on Robotics and Automation* (1994), vol. 4, pp. 3324–3329.

[RB09] Raunhardt D., Boulic R.: Motion constraint. *The Visual Computer 25*, 5–7 (April 2009), 509–518.

[RGBC96] Rose C., Guenter B., Bodenheimer B., Cohen M. F.: Efficient generation of motion transitions using spacetime constraints. In *SIGGRAPH '96: Proceedings of the 23rd Annual Conference on Computer Graphics and Interactive Techniques* (NY, USA, 1996), ACM, pp. 147–154.

[RISC01] Rose III C. F., Sloan P.-P. J., Cohen M. F.: Artist-directed inverse-kinematics using radial basis function interpolation. *Computer Graphics Forum 20*, 3 (2001), 239–250.

[RK16] Rabbani A., Kry P.: Physik: Physically plausible and intuitive keyframing. In *GI '16: Proceedings of the 42nd Graphics Interface Conference* (School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2016), Canadian Human-Computer Communications Society, pp. 153–161.

[Roo17] Root-Motion: FINAL-IK: http://root-motion.com/, accessed January 2017, 2017.

[RPE*05] Ren L., Patrick A., Efros A. A., Hodgins J. K., Rehg J. M.: A data-driven approach to quantifying natural human motion. *ACM Transactions on Graphics 24*, 3 (July 2005), 1090–1097.

[RR93] Raghavan M., Roth B.: Inverse kinematics of the general 6r manipulator and related linkages. *ASME Journal of Mechanical Design 115*, 3 (1993), 502–508.

[RSG10] Rolf M., Steil J., Gienger M.: Goal babbling permits direct learning of inverse kinematics. *IEEE Transactions on Autonomous Mental Development 2*, 3 (September 2010), 216–229.

[RTK*15] Rhodin H., Tompkin J., Kim K. I., de Aguiar E., Pfister H., Seidel H.-P., Theobalt C.: Generalizing wave gestures from sparse examples for real-time character control. *ACM Transactions on Graphics 34*, 6 (October 2015), 181:1–181:12.

[SH08] Slyper R., Hodgins J. K.: Action capture with accelerometers. In *SCA '08: Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, 2008), Eurographics Association, pp. 193–199.

[SH12] Shum H., Ho E. S.: Real-time physical modelling of character movements with microsoft kinect. In *VRST'12: Proceedings of the ACM Symposium on Virtual Reality Software and Technology* (NY, USA, 2012), ACM, pp. 17–24.

[Sic90] Siciliano B.: A closed-loop inverse kinematic scheme for on-line joint-based robot control. *Robotica 8* (7 1990), 231–243.

[Sim94] Sims K.: Evolving virtual creatures. In *SIGGRAPH '94: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques* (NY, USA, 1994), ACM, pp. 15–22.

[SK16] Siciliano B., Khatib O.: *Springer Handbook of Robotics* (2nd edition). Springer, Berlin/New York, 2016.

[SL11] Shapiro A., Lee S.-H.: Practical character physics for animators. *IEEE Computer Graphics and Applications 31*, 4 (July 2011), 45–55.

[SLSG01] Shin H. J., Lee J., Shin S. Y., Gleicher M.: Computer puppetry: An importance-based approach. *ACM Transactions on Graphics 20*, 2 (April 2001), 67–94.

[SMM14] Sapra R., Mathew M., Majumder S.: A solution to inverse kinematics problem using the concept of sampling importance resampling. In *Proceedings of 4th International Conference on Advanced Computing Communication Technologies (ACCT)* (February 2014), pp. 471–477.

[SPO10] Schvartzman S. C., Pérez A. G., Otaduy M. A.: Star-contours for efficient hierarchical self-collision detection. *ACM Transactions on Graphics 29*, 4 (July 2010), 1–8.

[SS91] Siciliano B., Slotine J.-J.: A general framework for managing multiple tasks in highly redundant robotic systems. In *Proceedings of the International Conference on Advanced Robotics, ICAR* (June 1991), vol. 2, pp. 1211–1216.

[ST13] Sapp B., Taskar B.: Modec: Multimodal decomposable models for human pose estimation. In *CVPR '13: Proceedings of the 2013 IEEE Conference on Computer Vision and Pattern Recognition* (Washington, DC, USA, 2013), IEEE Computer Society, pp. 3674–3681.

[SYLH10] Sok K. W., Yamane K., Lee J., Hodgins J.: Editing dynamic human motions via momentum and force. In *SCA '10: Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Aire-la-Ville, Switzerland, Switzerland, 2010), Eurographics Association, pp. 11–20.

[SZGP05] Sumner R. W., Zwicker M., Gotsman C., Popović J.: Mesh-based inverse kinematics. *ACM Transactions of Graphics 24*, 3 (2005), 488–495.

[SZK15] Saito S., Zhou Z.-Y., Kavan L.: Computational bodybuilding: Anatomically-based modeling of human bodies. *ACM Transactions on Graphics 34*, 4 (2015).

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

[TGLT14] Tan J., Gu Y., Liu C. K., Turk G.: Learning bicycle stunts. *ACM Transactions on Graphics 33*, 4 (July 2014), 50:1–50:12.

[TGTL11] Tan J., Gu Y., Turk G., Liu C. K.: Articulated swimming creatures. *ACM Transactions on Graphics 30*, 4 (July 2011), 58:1–58:12.

[TS14] Toshev A., Szegedy C.: Deeppose: Human pose estimation via deep neural networks. In *CVPR '14: Proceedings of the 2014 IEEE Conference on Computer Vision and Pattern Recognition* (Washington, DC, USA, 2014), IEEE Computer Society, pp. 1653–1660.

[TWC*09] Tournier M., Wu X., Courty N., Arnaud E., Reveret L.: Motion compression using principal geodesics analysis. *Computer Graphics Forum 28*, 2 (April 2009), 355–364.

[TY16] Tao S., Yang Y.: Collision-free motion planning of a virtual arm based on the FABRIK algorithm. *Robotica 35*, 6 (June 2017), 1431–1450.

[TZK*11] Tautges J., Zinke A., Krüger B., Baumann J., Weber A., Helten T., Müller M., Seidel H.-P., Eberhardt B.: Motion reconstruction using sparse accelerometer data. *ACM Transactions on Graphics 30*, 3 (May 2011), 18:1–18:12.

[UPBS08] Unzueta L., Peinado M., Boulic R., Suescun A.: Full-body performance animation with sequential inverse kinematics. *Graphical Models 70*, 5 (2008), 87–104.

[VAD12] Vahrenkamp N., Asfour T., Dillmann R.: Efficient inverse kinematics computation based on reachability analysis. *Journal of Humanoid Robotics 9*, 4 (2012), https://doi.org/10.1142/S0219843612500351.

[vdB98] van den Bergen G.: Efficient collision detection of complex deformable models using aabb trees. *Journal of Graphics Tools 2*, 4 (January 1998), 1–13.

[VMT94] Volino P., Magnenat-Thalmann N.: Efficient self-collision detection on smoothly discretized surface animations using geometrical shape regularity. *Computer Graphics Forum 13*, 3 (August 1994), 153–166.

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

[Wan99] Wang X.: A behavior-based inverse kinematics algorithm to predict arm prehension postures for computer-aided ergonomic evaluation. *Journal of biomechanics 32*, 5 (May 1999), 453–460.

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

[Wel93] Welman C.: *Inverse Kinematics And Geometric Constraints for Articulated Figure Manipulation*. Master Dissertation, Simon Fraser University, Department of Computer Science, 1993.

[WG01] Wilhelms J., Gelder A. V.: Fast and easy reach-cone joint limits. *Journal of Graphic Tools 6*, 2 (2001), 27–41.

[WK98] Wolpert D. M., Kawato M.: Multiple paired forward and inverse models for motor control. *Neural Networks 11*, 7–8 (October 1998), 1317–1329.

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

[WW92] Watt A., Watt M.: *Advanced Animation and Rendering Techniques*. Addison-Wesley, Reading, MA, 1992.

[Xie16] Xie S.: *Modelling Human Shoulder and Elbow*. Springer International Publishing, Cham, Switzerland, 2016, pp. 81–118.

[XW01] Xia Y., Wang J.: A dual neural network for kinematic control of redundant robot manipulators. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics 31*, 1 (2001), 147–154.

[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.

[YLvdP07] Yin K., Loken K., van de Panne M.: Simbicon: Simple biped locomotion control. *ACM Transactions on Graphics 26*, 3 (July 2007).

[YN03] Yamane K., Nakamura Y.: Natural motion animation through constraining and deconstraining at will. *IEEE Transactions on Visualization and Computer Graphics 9*, 3 (July 2003), 352–360.

[YR13] Yang Y., Ramanan D.: Articulated human detection with flexible mixtures of parts. *IEEE Transactions on Pattern Analysis and Machine Intelligence 35*, 12 (December 2013), 2878–2890.

[YVN*14] YOO I., VANEK J., NIZOVTSEVA M., ADAMO-VILLANI N., BENES B.: Sketching human character animations by composing sequences from large motion database. *The Visual Computer 30*, 2 (February 2014), 213–227.

[YY12] YOSHIYASU Y., YAMAZAKI N.: Example-based inverse kinematics using cage. *Computer Animation and Virtual Worlds 23*, 3–4 (May 2012), 203–213.

[ZB94] ZHAO J., BADLER N. I.: Inverse kinematics positioning using nonlinear programming for highly articulated figures. *ACM Transactions on Graphics 13*, 4 (1994), 313–336.

[ZKVM07] ZHANG L., KIM Y. J., VARADHAN G., MANOCHA D.: Generalized penetration depth computation. *Computer-Aided Design 39*, 8 (2007), 625–638.