DOI: 10.1111/j.1467-8659.2012.03189.x

COMPUTER GRAPHICS *forum*
Volume 31 (2012), number 8 pp. 2492–2515

Interactive Character Animation Using Simulated Physics: A State-of-the-Art Review

T. Geijtenbeek and N. Pronost

Department of Information and Computing Sciences, Utrecht University, The Netherlands
{t.geijtenbeek, n.pronost}@uu.nl

Abstract

Physics simulation offers the possibility of truly responsive and realistic animation. Despite wide adoption of physics simulation for the animation of passive phenomena, such as fluids, cloths and rag-doll characters, commercial applications still resort to kinematics-based approaches for the animation of actively controlled characters. However, following a renewed interest in the use of physics simulation for interactive character animation, many recent publications demonstrate tremendous improvements in robustness, visual quality and usability. We present a structured review of over two decades of research on physics-based character animation, as well as point out various open research areas and possible future directions.

Keywords: computer animation, physics-based animation, physics simulation, motion control, virtual humans

ACM CCS: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism -Animation I.6.8 [Simulation and Modelling]: Types of Simulation -Animation

1. Introduction

Many computer graphics applications involve virtual environments in which characters and objects continuously interact with each other and with their surroundings. Proper animation of such interaction is important for the perceived realism of these virtual environments. However, creating realistic responsive animation is challenging, because the range of possible interactions is enormous, and subtle variations in initial contact may call for substantially different responses.

Kinematics-based animation frameworks rely heavily on existing motion data, either recorded or manually crafted through *keyframing*. During interactions, proper responses are selected based on events, rules and scripts, after which representative animations are generated using motions from a database. Even though the last decade has brought great advances both in availability and utilization of motion data, data-driven animation suffers from one major drawback: the ability to generate realistic and non-repetitive responsive animations is always restricted by the contents of the motion database.

Physics simulation offers a fundamentally different approach to computer animation. Instead of directly manipulating the motion trajectories of objects and characters, this approach lets all motion be the result of a *physics simulation process*. As a consequence, physics-based characters and objects automatically interact in a way that is physically accurate, without the need for additional motion data or scripts.

The idea of using physics simulation to animate virtual characters has been recognized in an early stage of 3D computer animation [AG85, Wil87] and has incited several research publications since (Figure 1). For the animation of *passive* phenomena, such as cloth, water and *rag-doll* characters, physics simulation has been subject to widespread commercial adoption, both in video games and production movies. However, despite more than two decades of research, commercial frameworks still resort to kinematics-based approaches when it comes to animating *active* virtual

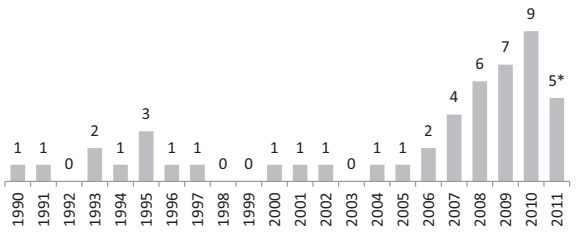

Figure 1: *The yearly number of SIGGRAPH and EUROGRAPHICS publications on interactive character animation using simulated physics. *Statistics do not include the 2011 special issue of Computer Graphics and Applications.*

characters [PP10, HZ11]. We can point out a number of reasons explaining why.

First of all, physics-based characters have several issues related to *controllability* (or lack thereof). Just like with real-world characters, the pose of a physics-based character is controlled indirectly, through forces and torques generated by *actuators* that reside inside the character—similar to muscles in biological systems. Moreover, global position and orientation of a physics-based character are *unactuated* and can only be controlled through deliberate manipulation of external contacts. This characteristic (in literature also referred to as *underactuation*) poses a direct challenge to basic tasks such as balance and locomotion—a challenge that has no equivalent in kinematics-based animation.

The reduced controllability of physics-based characters also affects *visual quality*. Directing style through application of forces and torques is non-trivial, since their effect is indirect and may disrupt basic tasks such as balance. The result is that several (especially early) physics-based character animations have little attention payed to style and appear stiff or robotic [NF02, WFH09, LKL10]. Even though their motions are *physically* accurate, they may not exhibit natural properties of real-world motion, such as gait symmetry, passive knee usage or passive arm swing [BSH99, Nov98, dLMH10], and may therefore not be perceived as natural.

The fact that global position and orientation are unactuated also affects *user control*, especially in high-paced action video games. When a user commands a physics-based character to turn or change walking speed, it must respond to such a command through an intricate scheme of external contact manipulation tasks. The result is that physics-based characters are less reactive during direct control tasks than kinematics-based characters.

A final issue preventing widespread adaptation of physics-based character animation is *implementability*. Even though professional physics simulation software has become readily available, the animation of active physics-based characters is a complex matter that alludes to many different disciplines. Successful implementation of a physics-based character animation framework requires at least some knowledge of multi-body dynamics, numerical integration, biomechanics and optimization theory. In addition, many physics-based control strategies require skilful and time consuming manual tuning before they can be put to use—a process often poorly documented in research publications. Finally, physics-based character animation is computationally more expensive than kinematics-based alternatives. It may only be for about a decade that a single passive physics-based character can be simulated in real-time on a consumer-grade PC. Currently, real-time performance of many recent control strategies (e.g. [dLMH10, JL11b]) is still limited to around a single character at a time on modern hardware.

On the other side, there is also a recent renewed interest in using simulated physics for interactive character animation (Figure 1), which has led to tremendous progress in dealing with these challenges. After years of focus on data-driven animation techniques, we expect physics simulation to play an increasingly important role in interactive character animation in the upcoming years.

This review aims to provide a structured evaluation of different aspects, approaches and techniques regarding interactive character animation using simulated physics. It is intended both as a thorough introduction for people with an interest in physics-based character animation, as well as a source of reference for researchers already familiar with the subject.

In Section 2, we describe the fundamental components of physics-based character animation: *physics simulation*, *physics-based character modelling* and *motion control*. This is followed by descriptions of three basic strategies that have been used for motion control of physics-based characters: *joint-space motion control* (Section 3), *stimulus–response network control* (Section 4) and *constrained dynamics optimization control* (Section 5). We conclude by providing a summary of different approaches and techniques, and point out possible directions for future research (Section 6).

2. Fundamentals

There are three components that are fundamental to interactive character animation using simulated physics:

1. A component performing *real-time physics simulation*.
2. A *physics-based character* to act in the simulation.
3. A *motion controller* to control a physics-based character.

The remainder of this section is dedicated to each of these components.

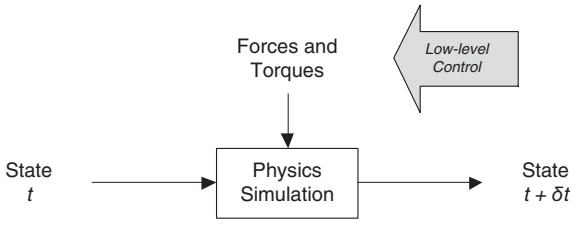

Figure 2: *Animation using physics simulation. Control occurs only through application of forces and torques.*

2.1. Real-time physics simulation

All motion in interactive physics-based animation is the result of a component performing *real-time physics simulation*. Such a component (often called a *physics simulator* or *physics engine*) iteratively updates the state of a virtual environment, based on physics principles (most notably the *Newton–Euler laws of dynamics*). Control of elements in the virtual environment is admissible only through application of external forces and torques. For a schematic overview, see Figure 2.

In physics simulation, the modelling of material properties, contacts and friction can occur in many different levels of detail; there is generally a trade-off between accuracy and performance. In physics-based character animation, it is common to use what is referred to as *constrained rigid body simulation*, where *rigid* indicates that bodies are non-penetrable and non-elastic, while their motion is *constrained* because body parts are linked together through *joints*. A physics simulator generally performs the following steps:

1. *Collision detection:* investigates if intersections exist between the different object geometries.
2. *Forward dynamics simulation:* computes the linear and angular acceleration of each simulated object.
3. *Numerical integration:* updates positions, orientations and velocities of objects, based on accelerations.

Step 2 and 3 are coupled in cases were *implicit integration methods* are used, that is, when positions and velocities are formulated as functions of accelerations.

There are many textbooks available that describe methods for collision detection (e.g. [Cou01]), all of which are beyond the scope of this review. In this section we focus on the stages *after* collision detection. However, one topic that we will briefly mention is *self-collision*, for example collision between two legs of a biped character. Most publications on physics-based character animation ignore this type of collision and allow legs to penetrate each other. It is also common to ignore collisions between bodies that are directly connected via a joint. For such bodies, *joint limits* are imposed instead (Section 2.2.1).

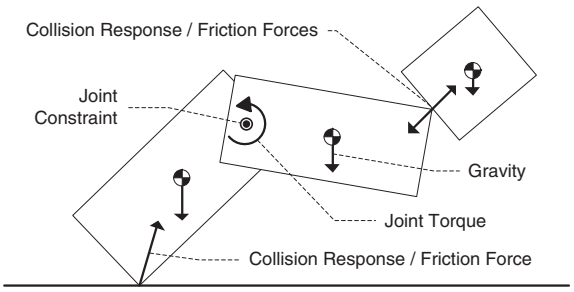

Figure 3: *Different elements in forward dynamics simulation.*

2.1.1. *Forward dynamics simulation*

The goal of forward dynamics is to compute linear and angular accelerations of simulated objects, based on external forces and constraints. We will briefly describe some of the basic principles of forward dynamics and refer to a textbook such as [Fea08] for details. See also Figure 3 for a depiction of different elements in forward dynamics simulation.

Newton–Euler laws of dynamics. The state of a single rigid body is described not only by *position* and *orientation*, but also by *linear velocity* and *angular velocity*. Change in linear velocity v depends on mass m, and the sum of the j applied forces, $F_1, \ldots, F_j$

$$m\frac{\partial v}{\partial t} = \sum_{i=1}^{j} F_i. \tag{1}$$

Change in angular velocity ω depends on how the mass is distributed with respect to the centre of mass, represented by *inertia tensor* matrix I, and the sum of the k applied torques, $T_1, \ldots, T_k$

$$I\frac{\partial \omega}{\partial t} = \sum_{i=1}^{k} T_i. \tag{2}$$

Forces and torques can be *external* (e.g. gravitational forces or collision response forces) or *internal* (e.g. the result of joint constraints or muscles contracting).

The dynamics of a system of bodies can be described using an n-sized vector representing the *generalized degrees of freedom* (DOFs) of the system, $q = [q_1, \ldots, q_n]$

$$M(q)\,\ddot{q} + \tau = 0 \tag{3}$$

in which $M(q)$ is the $n \times n$ matrix representing mass information (in generalized form, depending on q), $\ddot{q}$ are the accelerations of the generalized DOFs, and τ is the n-sized vector representing forces and torques acting on the generalized DOFs in q. Forces and torques include internal *centrifugal* and *Coriolis* forces (virtual forces due to a

changing reference frame), as well as external forces and torques caused by gravity or external contacts. Algorithms for constructing $M(q)$ and computing components in τ are described by Kane and Levinson [KL96]. The relations described by Equation (3) are also referred to as the *equations of motion*.

Joint constraints. The different body parts of a physics-based character are held together through *joints*, which restrict the motion of the connected bodies (see also Figure 3). In forward dynamics simulation, there are two basic approaches to enforce these constraints. One approach is to apply *constraint forces* to the two bodies connected by the joint. These forces must always be of equal magnitude and in opposite direction. Another approach is to reduce the amount of degrees of freedom in Equation (3); this automatically enforces joint constraints by simply not representing motions that would violate these constraints. The former approach is also referred to as *full* or *maximal coordinate* simulation, while the latter approach is referred to as *reduced coordinate* simulation. The use of reduced coordinates allows for higher simulation performance and has the benefit of not having to deal with constraint force tuning. On the other hand, reduced coordinate simulation is more vulnerable to numerical instability in the case of near-singularities, while full coordinate simulation has the benefit that constraint forces enable simple emulation of natural joint compliance [GVE11]. Featherstone [Fea08] describes an efficient approach for performing reduced coordinate simulation.

Collision response and friction. There are two ways to respond to collisions: by applying a *penalty force*, or by constructing a *collision constraint*. A collision response force consists of two components: one that is in the normal direction of a collision surface and pushes objects away from each other to prevent penetration, and one that is perpendicular to a collision surface and is the result of *friction*. When contacts are not sliding, the magnitude of the friction force is limited by the force in the normal direction and the material properties. This relation is often modelled using a *Coulomb friction model*, which can be formulated as

$$||f_{xz}|| \leq \mu ||f_y||, \tag{4}$$

where f_y is the normal component of the collision force, f_{xz} is the perpendicular friction component of the collision force, and μ is a friction coefficient. The volume spanned by the set of possible reaction forces has the shape of a cone, and is often referred to as the *Coulomb friction cone*. Dynamic friction (sliding) will occur when a required collision response force lies outside this cone. Several physics engines approximate the friction cone using an n-sided pyramid.

When collision response is modelled through constraints, a symbolic joint is constructed at the point of collision that prevents interpenetration of colliding objects. This link is removed when objects are pulled away from each other, or transformed into a sliding constraint when the inequality in Equation (4) no longer holds.

Even though it seems like a small aspect of physics simulation, the way in which contacts and friction are modelled can have a significant impact on motion control and the performance of a physics-based character [MdLH10, JL11a].

2.1.2. *Numerical integration*

After the accelerations of the virtual objects are known, they must be integrated to acquire updated velocity and position. A numerical integrator takes the position, velocity and acceleration at time t (q_t, $\dot{q}_t$ and $\ddot{q}_t$), and computes the updated position and velocity at time $t + \delta t$ ($q_{t+\delta t}$ and $\dot{q}_{t+\delta t}$). The choice of integration technique and size of time step δt are crucial for the numeric stability and performance of the simulation. Typically, a larger time step will decrease simulation stability (caused by accumulation of integration errors), while a smaller time step will decrease simulation performance. Several methods have been developed to allow greater robustness at larger time steps, such as the *Runge–Kutta methods*. We refer to a textbook [WL97] for details on numerical integration methods for dynamics simulation.

2.1.3. *Inverse and mixed dynamics*

Instead of computing the accelerations for a known set of joint torques, it is also possible to do this the other way around: compute the torques and forces required for a character to perform a specific motion. This process is called *inverse dynamics*. It is often used in biomechanics to analyse the dynamics of human motion, based on motion data that is augmented with external force measurements [EMHvdB07]. Inverse dynamics is also used in physics-based character animation to find the torques required to achieve a desired acceleration [FDCM97, YCP03, HMPH05, MZS09, LKL10]. It is also possible to combine forward and inverse dynamics, for example to control one half of a virtual character kinematically and the other half dynamically, by adding additional kinematic constraints to the dynamics equations. Such an approach is referred to as *mixed dynamics* [Ott03, VVE*10].

2.1.4. *Available physics simulation engines*

There are several readily available software libraries that implement collision detection, forward dynamics, and numerical integration in a single package. Popular physics engines include the *Open Dynamics Engine* (ODE) [Smi06], *Bullet* [CO], *PhysX*, *Newton*, and *Havok*. ODE is by far the most commonly used third party simulator in research publications on physics-based character animation. Other publications use internally developed reduced coordinate simulation, or use packages that produce efficient code for computing the elements of the matrices involved in the *equations of motion*

(see also Equation 3) for a specific character model. Examples of such packages are *SD/Fast* [HRS94] and *Autolev* [KL96]. Collision detection and response are not integrated in these packages. Another related software package is *Euphoria* [www.naturalmotion.com], which focuses on controlling physics-based characters in real-time, using proprietary algorithms.

An in-depth comparison between different engines is beyond the scope of this review; however, Boeing and Bräunl [BB07] present an evaluation of the performance of several of these engines.

2.2. Physics-based characters

Characters in a simulated physics-based environment must incorporate a number of attributes that are not required for kinematics-based characters. They require mass properties and joint constraints, as well as *actuators* to enable motion control—each of which translates to elements in Equation (3). In this section we describe basic principles of physics-based character modelling, as well as different actuation models.

2.2.1. *Character modelling*

Most physics-based characters are modelled after humans, while some research papers model characters after animals [RH91, GT95, CKJ*11, TGTL11], robots [Hod91, RH91], or creatures that do not exist in nature [Sim94, TGTL11]. Character models are often simplified to increase simulation performance and to make them easier to control. Sometimes humans are reduced to simple biped characters, with head, arms and trunk modelled using a single body [YLvdP07, SKL07]. The feet of physics-based humanoid characters are also mostly modelled using single bodies, although Wang *et al*. [WFH09] include a separate segment for toes in their foot model to allow better locomotion performance. In later work, they add cylinders to the foot model to allow for foot rolling after heel-strike [WHD12].

The choice of character model can sometimes be linked to the control strategy of the character. An illustrative example from robotics are the so-called *passive-dynamic walkers*, which are biped structures that walk downhill robustly without actuation [McG90]. Modified versions of these passive-dynamic walkers are able to perform biped locomotion on straight terrain, with very little actuation [TZS04, CRTW05].

Bodies. The individual body segments of a physics-based character require both a *geometry*, which is used for collision detection, as well as *mass information*, which is used during forward dynamics simulation. Body segments are typically modelled using primitive shapes, such as boxes, spheres or cylinders, to increase collision detection performance. These shapes are also often used to compute the mass information of the individual bodies, assuming uniform mass distribution. Alternatively, mass properties can be derived from cadaveric data [ZSC90, De 96, VDJ99]. The collision detection geometry is independent from the geometry used for visualization, even though coherence between the two is likely to increase perceived realism.

Joints. The bodies of a physics-based character are connected through various types of joints. Commonly, mechanical joint types are selected to approximate natural constraints imposed by bone tissue and ligaments. For instance, knee and elbow joints are often modelled using hinge joints, while hip and shoulder joint are often modelled using ball-and-socket joints. Joints are constrained by *joint limits*, which are specified as lower and upper bounds for each DOF, and are used to mimic natural joint limits. The final range of motion of a character can be verified by comparing it to actual human motion data [Fro79]. Kwon and Hodgins [KH10] add sliding joints below the knee and the hip to emulate shock absorption due to soft tissues, joint compliance and ligaments. In some physics engines (such as the Open Dynamics Engine [Smi06]), it is possible to soften specific joint constraints to emulate this behaviour.

Contact modelling. Research papers sometimes explicitly model passive mechanisms for energy storage and release during contact. Raibert and Hodgins [RH91] use non-linear springs to model the padding material that some creatures have under their feet. Liu *et al*. [LHP05] model the springy tissue of a character's shoe sole. Pollard and Zordan [PZ05] use approximated soft contacts in simulated grasping motions. Jain and Liu [JL11a] show that soft contact modelling applied to the hands and feet of a character can increase robustness during locomotion, as well as allow greater control during object manipulation tasks.

2.2.2. *Character actuation*

Characters need forces and torques to actively move around. In order for such forces or torques to be realistic, they must originate from within the character. We use the term *actuators* to describe the mechanisms that generate the forces and torques that make a character move.

Joint torques. The most straightforward actuation model is to generate joint torques directly for each actuated DOF. This way, the number of DOFs of the actuation model is the same as the number of active DOFs in the character. This actuation method can be visualized by assuming there exists a *servomotor* in each actuated joint, applying torques to its connected limbs.

External forces. Even though real-world characters are controlled through forces or torques originating from inside a character, animated characters can also be controlled through

external forces and torques—similar to a puppet master controlling a puppet. The direct control over global translation and rotation greatly simplifies the control strategy, but the fact that real-world characters do not possess such 'actuators' may result in loss of realism. To emphasize its supernatural quality, this actuation method is also referred to as the *Hand-of-God* [vdPL95]. Wrotek *et al*. [WJM06] demonstrate how external forces can greatly increase balance capabilities for interactive gaming applications.

Virtual forces. This method is strictly speaking no actuation method, but a control abstraction that acts like an actuator. In this method, the effect of applying an external force at a specific point is emulated by computing the internal joint torques that would have had the same effect (for as far as such torques exist). The method uses the *Jacobian* that describes the relation between joint rotations and the position at which the virtual force is applied, calculated from a *fixed base link*. If P is a position and θ a vector of joint orientations, then the Jacobian is defined as $J = \frac{\partial P}{\partial \theta}$. A force F applied at position P can be translated into a vector of torques τ, applied to the chain of bodies from fixed base to target link, using the *Jacobian transpose*: $\tau = J^T F$. This process is described in detail as part of the *virtual model control* strategy of Pratt *et al*. [PCTD01], and is used extensively in the work of Coros *et al*. [CBvdP10, CKJ*11].

Muscle contraction. Biological systems are actuated through contraction of *muscles*, which are attached to bones through *tendons*. When muscles contract, they generate torques at the joint(s) over which they operate (a single muscle can span multiple joints), causing the attached limbs to rotate in opposite directions. In addition to contracting when activated, muscles (and tendons, in a lesser degree) have the ability to stretch. This makes them behave like unilateral springs and allows for an efficient mechanism for energy storage and release [LHP05, AP93].

Since muscles can only pull, at least two muscles are required to control a single DOF (so-called *antagonistic muscles*). Biological systems often contain many more redundant muscles. In physics-based character animation, use of muscle-based actuation models is uncommon, because of the increased number of DOFs that require control and decreased simulation performance [WGF08]. However, examples of muscle based actuation do exist, and we witness an increased interest in using more advanced muscle-based actuation models for controlling physics-based characters [GT95, HMOA03, SKP08, LST09, MY11, WHD12].

Actuation limits. In biological systems, actuation is limited by the maximum force of a muscle. In simulated environments, actuation can too be limited to prevent characters from appearing unnaturally stiff or strong, or to increase the stability of the simulation. With muscle-based actuation models, maximum forces can be derived from biomechanics research [van94]. However, with joint torque actuation models used in animation research, maximum torque values are generally fixed estimates per DOF [LWZB90, KB96, OM01], even though maximum torques in biological systems are pose-dependent [GvdBvBE10].

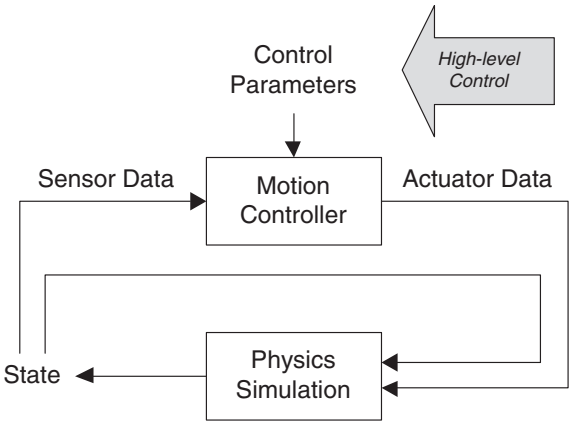

Figure 4: *Closed-loop motion control.*

2.3. Motion control

Motion control is the key of any physics-based character animation framework. A *motion controller* is a component responsible for generating *actuation patterns* that make a physics-based character perform its goal tasks: a *virtual brain*. Possible goal tasks include walking, running, picking up objects, stepping over obstacles, etc.—all while remaining balance and withstanding unexpected external perturbations.

Similar to brains in biological systems, the computation of actuation patterns is based on *sensor data* that provides feedback from the state of the virtual environment and the character itself (Figure 4). Such a control method is referred to as *closed-loop control*, denominadesignating the presence of a closed feedback loop. Sensor data can represent a character's state (e.g. joint orientation, centre of mass (COM) position and velocity, or total angular momentum [GK04]), it can represent external contact information (such as a *support polygon*), or it can be a task-related parameter, such as the relative position of a target object.

The opposite *open-loop control* (or *feed-forward control*) refers to a system that uses no sensor data or feedback loop. The use of open-loop control is uncommon in physics-based character animation, although feed-forward torques are sometimes used to complement torques from a closed-loop control system (Section 3.3).

Motion controllers allow interaction with physics-based characters through *high-level control parameters*, such as desired walking speed, direction, or goal task. In general,

physics-based characters possess a high level of autonomy, although some research papers focus on *direct user control* of physically simulated characters, through interface devices such as keyboard and mouse [LvdPF00, KP11], accelerometers [SH08] or motion capture [IWZL09, NWB*10].

Design approaches. Motion controller design is a multi-disciplinary topic, with ties to several research areas, including biomechanics, robotics, control theory, artificial intelligence and optimization theory. There are many different angles from which to explore and classify attempts at motion control for physics-based characters, and there is currently no consensus as to a good classification scheme. After careful deliberation, we have decided to classify different approaches based on the primary research field from which they originate:

- *Joint-space motion control* (Section 3). This approach originates from classic control theory; it attempts to control characters by defining kinematic targets for all actuated joints, and utilizes *local feedback controllers* to compute the torques required to achieve these targets.
- *Stimulus–response network control* (Section 4). This approach originates from artificial intelligence and is inspired by biological systems; it attempts to achieve motion control by using a generic control network (such as an *artificial neural network*) to construct explicit relations between input sensors and output actuators. Such control frameworks typically contain no *a priori* knowledge of the control task or character; the set of optimal control parameters is sought through off-line optimization, according to high-level objectives.
- *Constrained dynamics optimization control* (Section 5). This approach originates from optimal control theory and optimization theory. It attempts to find the actuator values (i.e. joint torques) that are optimal with regard to a set of high-level objectives, subject to the *equations of motion* describing the constrained dynamics of the character model and its environment.

Independent of the design approach, there are several techniques that are commonly used in motion control for physics-based characters. First of all, motion controllers often maintain a *simplified model* of a physics-based character to help simplify a control strategy. An example is the *Inverted Pendulum Model* (IPM), which is used in biped locomotion control to model the character's COM during single-stance phase, and to predict an optimal target position for the swing leg [TLC*09, CBvdP10].

Also important are the provisions for controlling *motion style*, to reflect mood, personality or intention of a character. Some methods provide style control through editing key poses, others allow inclusion of continuous motion data. Motion capture data allows easy authoring of specific motion styles, but controllers that depend on motion capture data suffer from some of the same limitations as kinematics-based data-driven animation systems.

Motion controllers may benefit from off-line optimization of their internal parameters, based on high-level goals such as *walking speed* or *energy efficiency*—analogous to learning or evolving in biological systems. However, finding an appropriate high-level optimization metric is difficult, because the parameters that describe natural motion are considered elusive and enigmatic [vdP00], while simulated physics-based characters have different optimums than biological equivalents, because they often ignore important physical attributes [LHP05, WHD12].

To achieve higher-level composite behaviours, individual motion controllers can be combined in a meta-control framework. For this purpose, Faloutsos *et al*. [FvdPT01, Fal01] have developed a meta-control framework that allows flexible transition from one controller to another, independent of controller type and implementation. Physics-based controllers may also be used in combination with kinematics-based controllers to get the best of both worlds. For more details on this topic, we refer to the works of Shapiro *et al*. [SPF03] and Zordan *et al*. [ZMCF05, ZMM*07].

3. Joint-Space Motion Control

Joint-space motion control attempts to control physics-based characters by providing *kinematic target trajectories*, which are tracked using *local feedback controllers* that attempt to minimize the difference between the measured state and target state. It has its origin in *control theory* and is the most common control method in industrial robotics [KSnL05].

Figure 5 shows a schematic overview of joint-space motion control for physics-based character animation. Its main component performs *balance and pose control*, based on the current state, high-level control parameters, and (optionally) a time-based reference pose. It outputs a *desired kinematic state* for each joint, usually in the form of desired orientation and angular velocity. For each actuated joint, a *local feedback controller* computes a joint torque based on the difference between measured and target state.

The main appeal of this approach is its ease of implementation: all motion can be specified in the kinematic domain,

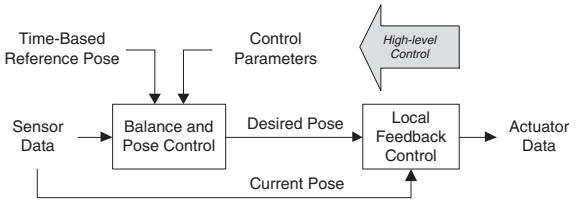

Figure 5: *Joint-space motion control.*

based on kinematic insights, without the need for direct access to the underlying equations of motion. The downside of using local feedback controllers is that they operate individually, while coordinated motion is the product of all joints working together. The key challenge of joint-space motion control is in devising methods to compensate for this lack of coordination.

In the remainder of this section, we first describe techniques for local feedback control, followed by an overview of different approaches for generating kinematic targets.

3.1. Local feedback control

The purpose of a local feedback controller is to compute a torque that minimizes the difference between the current state and desired state of a single actuated joint. Physics-based characters generally contain rotational joints; kinematic targets are therefore mostly represented through joint orientation and angular velocity. However, feedback control strategies can be applied similarly to prismatic (sliding) joints.

3.1.1. *Proportional-derivate control*

The most widely used feedback control technique in joint-space motion control is called *proportional-derivative control*, or *PD control*. It computes a joint torque, τ, that is linearly proportional to the difference between the current state and a desired state. It takes into account both the difference in current orientation θ, and desired joint orientation θ_d, as well as the difference in current angular velocity $\dot{\theta}$, and desired angular velocity $\dot{\theta}_d$

$$\tau = k_p(\theta_d - \theta) + k_v(\dot{\theta}_d - \dot{\theta}). \tag{5}$$

In this equation, k_p and k_v are *controller gains*, and they regulate how responsive the controller is to deviations in position and velocity, respectively.

Often, applications only specify a desired joint orientation and set desired angular velocity to zero ($\dot{\theta}_d = 0$). A PD controller then becomes similar to a spring-damper system, where a spring generates a force to move to its rest position, θ_d. In such a system k_p defines the spring gain (or *spring constant*), and k_v the damping.

Parameter tuning. Finding correct values for k_p and k_v is not a straightforward task, and often requires manual tuning through *trial-and-error*. When controller gains are set too low, a joint may not be able to track its target and lag behind. When set too high, a joint may follow a target trajectory too rigidly and become stiff and unresponsive. The amount of rigidity is also referred to as *impedance*—in skilled human motion, impedance is usually low [TSR01, Hog90], while high impedance motion is regarded as robotic or stiff [NF02].

k_p and k_v must also be set in a proper relation to each other. A relatively high value for k_p may result in *overshoot* (meaning that a joint will oscillate around its desired position before reaching it), while a relatively high value of k_v may cause unnecessary slow convergence. When the relation between k_p and k_d is optimal (fastest convergence without overshoot), a controller is said to be *critically damped*. If the desired state and dynamics characteristics are fixed, the optimal relation between the two gain parameters is: $k_v = 2\sqrt{k_p}$. For physics-based characters this relation is more complex, as both the desired state and the dynamics characteristics are subject to constant change. Still, this relation is often used as an estimate or a starting point for tuning.

To decrease the amount of tuning, Zordan and Hodgins [ZH02] scale controller gains in accordance to an estimate of the moment of inertia of the chain of bodies controlled by a joint. They also increase responsiveness by temporarily lowering controller gains for significant body parts when a perturbation is detected. Another way to decrease manual tuning is to use off-line optimization of the gain parameters [vdP96, HP97].

Stable PD control. Tan *et al*. [TLT11] present a modification to the standard PD control formulation that increases stability at high gains. Instead of using the current position and velocity to compute joint torques, they estimate the position and velocity during the upcoming time step, based on the current velocity and acceleration. These estimates are then used in Equation (5), together with the desired position and velocity during the upcoming time step.

PID control. When affected by external forces such as gravity, the steady state of a PD controller may be different from the desired position [NF02]. Industrial control systems often add an *integral* component to the equation that compensates for accumulated error between current and target position. Such a control method is called *proportional-integral-derivative control*, or *PID control*. However, in character animation such an approach has very limited use, because any change in desired state invalidates the current integral error.

3.1.2. *Alternatives to PD control*

Inspired by muscle-based actuation, Neff and Fiume [NF02] propose *antagonist feedback control* as an alternative to PD control. For each actuated DOF, a pair of antagonistic springs operate in opposing direction to create a joint torque and simultaneously regulate impedance. Each spring has a fixed set point, located at either joint limit. Instead of setting a target angle, this method varies the spring gains to achieve a

desired position. Since both springs have linear control, there exists a mathematical equivalence between PD control and antagonist control. The main advantage of antagonist control is that its parameters allow for more intuitive tension control.

Another alternative feedback control method is called *non-linear force fields* [MI97, MWDM98, MZW99], which is a non-linear feedback control method based on principles from biomechanics research. Kokkevis *et al*. [KMB96] describe a technique called *Model Reference Adaptive Control* (MRAC), which uses a reference model to calculate joint torques that result in a selected convergence speed. We refer to cited papers for more details.

3.2. Balance and pose control

In this section, we will describe different approaches that have been used to generate kinematic trajectories for physics-based characters using joint-space motion control. Such target trajectories have two functions: controlling the *pose* of the character and controlling its *balance*. Since these functions are often entwined, we describe them through specific combined approaches, as they have appeared in research.

3.2.1. *Procedural motion generation*

Early attempts in joint-space motion control generate motion using handcrafted functions or procedures that attempt to model behaviours witnessed in biological systems. Such algorithms are developed based on human intuition, using insights from physics, robotics and biomechanics.

An early example and excellent illustration of this approach is the work of Raibert and Hodgins [RH91], who construct various gait types for a number of low-dimensional robot characters. Their character models include passive springs to emulate natural mechanisms for energy storage-and-release, enabling hopping style motions without explicit control. Control is governed by a *finite state machine* that tracks the current phase of the gait cycle, based on foot sensor data. During each phase, a set of control strategies work in parallel to control balance, speed and posture. One control strategy is responsible for target foot placement during swing phase and uses a simplified inverted pendulum model in combination with inverse kinematics. The foot placement not only handles balance, but also enables high-level speed control: the difference between desired speed and current speed is used to offset the foot placement. Another control strategy attempts to keep the upper body upright in world coordinates, by applying a torque to hip and knee joint of the stance leg.

Extending this approach, Hodgins *et al*. [HWBO95] develop controllers that allow running, cycling and vaulting motions for a full-body human character. In their running controller, just before heel-strike, they adjust the hip torque to ensure the swing foot has zero relative speed when compared to the ground surface—a natural behaviour called *ground speed matching*. Wooten and Hodgins [WH00] demonstrate controllers for leaping, tumbling, landing and balancing. Faloutsos *et al*. [FvdPT01, Fal01] demonstrate controllers for sitting, falling, rolling over and getting back up.

The development of procedural joint-space controllers is a skillful process that requires laborious trial-and-error, while resulting controllers are often inflexible to even minor changes in character morphology or environment. Hodgins and Pollard [HP97] show how to adapt their running and cycling controllers to different character morphologies, but their method requires off-line optimization. Another issue with these controllers is that they focus on function instead of style. As a result, animations generated using these controllers appear robotic [NF02].

In recent research, Tan *et al*. [TGTL11] use procedural motion generation and joint-space motion control to animate underwater creatures, using target trajectories based on geometric functions. The parameters for these functions are found through off-line optimization for high-level criteria, such as maximization of swimming speed and minimization of energy consumption.

3.2.2. *Pose-control graphs*

Pose-control graphs are an attempt at a more generic approach to joint-space motion control. In this approach, each state in a finite state machine is linked to a *key target pose*, which is fed as input to local feedback controllers. The approach bears some resemblance to kinematic *keyframe animation*.

An advantage of the pose-control graphs method is that it allows intuitive creation or modification of different motion styles, without the need for detailed understanding of the underlying control strategy. Animators can create or tune controllers by simply editing the key poses. However, these key poses are not guaranteed to be reached during simulation; sometimes key poses need to be exaggerated, far beyond the intended pose to get a desired effect. Key poses also need to be tuned in conjunction with the gain parameters of the local feedback controllers.

This basic approach has been used successfully for animating low-dimensional 2D characters [vdPKF94], but 3D biped control requires additional balance correction. The main reason for this is that there is no tracking of global translation and rotation of the target poses. Laszlo *et al*. [LvdPF96] use pose-control graphs for balanced 3D locomotion controllers using *limit-cycle control*. To maintain balance, they apply linear pose corrections at the beginning of each walking cycle, based on pelvis rotation or centre of mass. In later research, pose-control graphs have been used to generate a number of interactive skills such as climbing stairs [LvdPF00] or

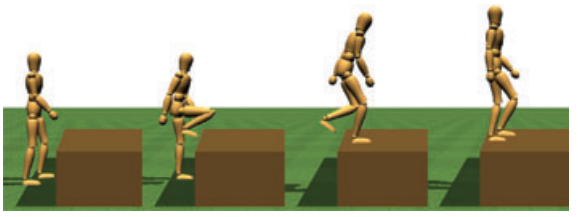

Figure 6: *SIMBICON controller developed using off-line optimization [YCBvdP08] (used with permission).*

user-controlled skiing and snowboarding [ZvdP05]. Yang *et al*. [YLS04] developed a layered pose-base swimming controller.

SIMBICON. Yin *et al*. [YLvdP07] introduce a generic framework for biped locomotion control, based on pose-control graphs. Their SIMBICON framework (an acronym for SIMple BIped CONtroller) allows biped locomotion control with a large variety of gaits and styles. Example controllers can walk in different directions and perform various gaits such as running, skipping and hopping, using only two to four states. At the core of this control framework exists an efficient balance control strategy, which corrects foot placement using the swing hip and provides posture control using the stance hip.

The SIMBICON framework has been at the basis for a range of research projects. Coros *et al*. [CBYvdP08] use off-line optimization to develop a top-level control policy that switches between controller instances, based on the current state of the character and a goal task. Using this strategy, they demonstrate a controller that can walk over gaps while planning several steps ahead [CBYvdP08], as well as a controller that performs tasks with long-term position and heading goals, with increased robustness [CBvdP09].

Other researchers focus on off-line optimization of the parameters of the SIMBICON framework. Yin *et al*. [YCBvdP08] demonstrate how controllers can be optimized to step over obstacles, walk on ice, traverse slopes and stairs, and push or pull large crates (see Figure 6). Wang *et al*. [WFH09] optimize the SIMBICON parameters to influence motion style, using a compound optimization metric that includes terms for energy minimization, gait symmetry and head stabilization. In subsequent research, Wang *et al*. [WFH10] increase robustness by optimizing controllers in simulations with added uncertainty and constrained environments. Both results use Covariance Matrix Adaption (CMA) [Han06] for optimization.

Coros *et al*. [CBvdP10] have extended the SIMBICON framework in a number of ways, resulting in a control framework for walking that no longer needs to be tuned for specific size, motion style or walking speed. Most importantly, they use *virtual forces* (see Section 2.2.2) to compensate for gravity, to control speed and for subtle balance control. The use of virtual forces defeats many of the issues caused by the local and uncoordinated nature of feedback controllers. Another addition is the use of an inverted pendulum model to control swing foot placement, effectively making balance control independent from body height (similar to Tsai *et al*. [TLC*09]). Thirdly, they use key target poses to generate continuous spline-based target trajectories, which increases robustness of tracking. Finally, they use *inverse kinematics* (IK) to adjust the upper body target position, allowing object manipulation tasks during locomotion. Even though the framework allows for various robust walking behaviours, it is not suitable for high-energy locomotion such as running. In later research, Coros *et al*. [CKJ*11] apply similar techniques to develop a control framework for several four-legged creatures, incorporating a flexible spine control model.

3.2.3. *Data-based motion generation*

Motion capture trajectories seem an ideal source for joint-space motion control, since they are known to be physically feasible and include both target positions and target velocities (after proper filtering). However, there are some discrepancies between the original motion and the simulated motion:

- A simulated physics-based character is never exactly identical to the performing actor.
- Motion capture systems do not capture many subtle balance correction behaviours of the performing actor.
- In real-time physics engines, the simulation of real-world phenomena such as friction, tissue deformation or joint compliance are highly simplified at best.

Because physics-based characters are underactuated, errors in global translation and orientation are allowed to accumulate, eventually leading to loss of balance. Sok *et al*. [SKL07] attempt to resolve these discrepancies by 'fixing' the kinematic target trajectories, using off-line optimization of non-linear displacement maps. Their method increases performance of 2D biped locomotion tracking, but still requires an additional balance compensation strategy.

Zordan and Hodgins [ZH02] track full-body boxing and table tennis motions and combine it with an in-place procedural balance strategy, similar to that of Wooten and Hodgins [WH00]. In addition, they use inverse kinematics to adjust upper body motion trajectories, creating interactive boxing and tennis controllers. Yin *et al*. [YLvdP07] use the SIMBICON balance strategy for 3D locomotion, based on pre-processed motion capture data.

Lee *et al*. [LKL10] demonstrate a framework that can track various styles of 3D locomotion data, which can be delivered on-the-fly without the need for pre-processing. They achieve balance by modulating the reference data using

SIMBICON-like balance strategies for stance hip, swing hip, stance ankle and swing foot. For feedback control, they use inverse dynamics to compute joint torques from target accelerations, based on the current state and reference data. Strictly speaking, their method does not classify as joint-space motion control; even though their approach is very similar to SIMBICON, the use of inverse dynamics requires access to the *equations of motion*, which is not supported by many common physics engines.

State-action mapping. This control strategy is based on the idea that the appropriate kinematic target of a physics-based character can directly be derived from the current pose of the character. The method maintains a set of mappings between current state and target pose, where a pose at time t (the source state) is mapped to a pose at time $t + \delta t$ (the target). The method is ideal for use with motion capture data, because both source and target can directly be derived from subsequent frames of motion data.

An early example of state-action mapping is *Banked Stimulus Response* [NM93], which has been used to develop several interesting 2D and 3D locomotion controllers (albeit without the use of motion capture data) [AFP*95]. Sharon and Van de Panne [SvdP05] use state-action maps that are initialized using motion capture data and optimized off-line. The active mapping is selected based on the current pose, using a nearest neighbour criterion. The resulting controllers demonstrate stylized 2D biped locomotion. Sok *et al*. [SKL07] extend this approach, using a much denser set of mappings, which are acquired from optimized motion capture clips of similar motions. Their input state includes velocity, foot position and ground contact information, while their target poses are augmented with joint velocities. Even though their framework allows for a wide range of motions and transitions between motions, it is still limited to 2D.

3.3. Feed-forward control

A number of joint-space motion control methods use predefined feed-forward torque patterns on top of local feedback torques. The motivation for this approach comes from evidence suggesting that biological systems use feed-forward control for most of their motions and use feedback control only for low-gain corrections [TSR01]. In physics-based animation, feed-forward torques have been acquired through optimization [GT95], inverse dynamics [YCP03], *feedback error learning* (FEL) [YLvdP07], and by using an on-line parallel auxiliary simulation of unperturbed motion capture data with high-gain tracking [NVCNZ08].

A problem with feed-forward torques is that they do not work well with discontinuous motions, because the precise timing of sudden torque change is not predictable in advance. An example is the sudden increase in torque during heel strike. As a result, feed-forward methods work best for continuous motions [GT95, YCP03], or for body parts that move more or less continuously during discontinuous motion, such as the upper body during walking [YLvdP07]. Exceptions exists though, as Yin *et al*. [YLvdP07] apply FEL to all joints when using the SIMBICON framework in combination with adapted motion capture data.

3.4. Summary

Joint-space motion control is the most well-studied approach for controlling physics-based characters; this has resulted in a vast number of controllers for various types of balance, locomotion and interaction. The main strengths of joint-space control are its intuitiveness and ease of implementation. Control strategies can largely be defined in the kinematic domain and local feedback controllers are easy to comprehend. Also, several recent publications have shown great robustness against external perturbations (e.g. [YLvdP07, LKL10]), while PD Control has shown to be an effective mechanism for simulating natural joint compliance.

A weakness of joint-space control methods is the local nature of the feedback controllers, which operate individually, in an uncoordinated fashion. As a result, many early examples of joint-space motion controllers require laborious tuning or lengthy optimization, after which they often appear stiff and robotic. The current state-of-the-art overcomes many of the problems of local feedback control by adding control techniques that allow for coordination amongst different actuators. Examples of this are the work of Coros *et al*. [CBvdP10, CKJ*11], who incorporate virtual forces in their control framework, and the work of Lee *et al*. [LKL10], who use inverse dynamics for tracking reference motion trajectories. Additionally, the use of feed-forward control (e.g. [YLvdP07]) implicitly embodies coordinated action. In parallel, research in off-line optimization of control parameters has resulted in impressive controllers, both for new behaviours [YCBvdP08], stylized locomotion [WFH09, WFH10], and swimming behaviours [TGTL11].

Both the implementations of the SIMBICON framework [YLvdP07] and the framework of Coros *et al*. [CBvdP10] are available on-line. In addition, DANCE [SFNTH05, SCAF07] is an open-source framework for developing motion controllers, with a focus on joint-space control methods.

4. Stimulus–Response Network Control

Inspired by motion control found in biological systems, this approach attempts to control characters through a network that connects sensor data (the stimuli) to actuator data (the response). The parameters of such control network are found through optimization using a high-level optimization metric, often referred to as *fitness function*.

The main appeal of this approach is that it allows motion controllers to be constructed automatically using only an

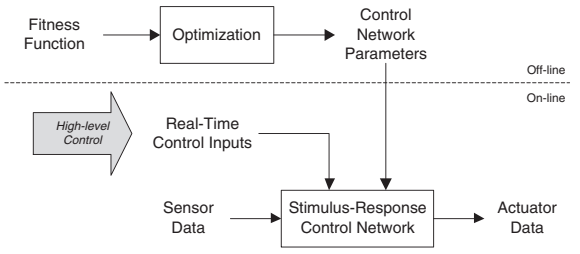

Figure 7: *A schematic overview of stimulus–response network control using off-line optimization.*

optimization metric; it requires no *a priori* knowledge of motion or character. However, devising such a metric is a daunting task that involves expert trial-and-error tuning. In addition, there seems to exist a trade-off between what a control network is capable of and its optimization performance. Control networks that are highly flexible often optimize poorly.

Even though optimization can be performed both off-line and on-line, research on physics-based character animation focuses on off-line optimization, where controller candidates are evaluated through short simulation runs during which the parameters do not change. The final optimized controller is the one with the highest fitness after a (usually large) number of trials. See Figure 7 for a schematic overview. On-line optimization occurs mostly in robotics [TZS04, MAEC07, BT08], where control parameters are adapted on-the-fly by a process called *reinforcement learning* [KLM96]. For robotics research, a generate-and-test approach is often impractical because of the large number of trials and damage risks, a notable exception being the work of Pollack *et al*. [PLF*00].

4.1. Stimulus–response network types

Even though stimulus–response networks exist in many flavors, they all consist of interconnected processing elements (often called *nodes* or *neurons*), each with a variable number of inputs and a single output. Control networks are typically *recurrent*, which means their internal links form a loop. This allows for the representation of an internal state, or memory. Delays can be added to each node output to promote real-time dynamical behaviours inside the control framework [vdPF93]. Nodes that are connected to sensor input are referred to as *sensor nodes*, while nodes that are connected to actuators are called *actuator nodes*.

Artificial neural networks. The most common type of stimulus–response network is the *artificial neural network* (ANN), in which each node outputs a value based on the weighted sum of its input nodes and a *threshold function* [Bis94]. Van de Panne *et al*. [vdPF93] use such a network for controlling various low-DOF 2D characters. Reil and Husbands [RH02] demonstrate locomotion control for a low-DOF 3D biped, using a small circular ANN with fixed topology and no input sensors. Their outputs generate target joint angles, which are converted into joint torques using PD controllers. Allen and Faloutsos [AF09] use a network with a flexible topology that is optimized along with its parameters, using the NEAT method [Sta04]. Their approach has produced a set of amusing gaits, but no stable 3D biped locomotion control.

Genetic programs. Other stimulus–response networks are more related to *genetic programs*, in the sense that their processing nodes can perform logical operations (*and*, *or*, etc.), decision operations (*if*, *then*, *else*), or memory storage [Gar90]. In addition, the networks of Sims [Sim94] use several node types that generate periodic signals, such as sine or sawtooth waves.

Cyclic pattern generators. An important related framework used in locomotion control is the *Cyclic Pattern Generator* (CPG) [Tag95]. The patterns produced by a CPG can represent joint torques, muscle activation, or target joint angles. CPGs have been used in combination with off-line optimization to produce biped locomotion controllers [TYS91, Tag95, Tag98, MTK98]. Grzeszczuk *et al*. [GT95] optimize CPGs that generate feed-forward muscle activation patterns, resulting in various swimming behaviours. The circular neural networks used by Reil and Husbands [RH02] are designed to emulate CPGs.

4.2. Fitness function design

Fitness functions used in physics-based motion control are often a weighted sum of individual terms with specific goals. Both formulation and weighting of the individual terms require manual tuning, often through time consuming trial-and-error. An example fitness function for developing locomotion controllers can be based on the following principles:

- To promote horizontal movement, increase fitness based on the horizontal distance of the COM from the origin after a fixed simulation time.
- To penalize falling down, decrease fitness when the vertical COM position falls below a certain threshold.

Using such basic approach, several researchers have succeeded in producing stable locomotion behaviours for 2D characters [vdPF93], 3D quadrupeds [AFP*95], and 3D bipeds [RM01, RH02]. Sims [Sim94] demonstrates several walking, running swimming and jumping behaviours for creatures with evolving morphology, using basic fitness functions. We have not seen any reports of locomotion controllers for full-body 3D humanoids (i.e. characters

with arms, legs and a head) that have been developed using stimulus–response networks.

Several fitness functions have been designed for interactive behaviours. For example, the distance to a target object has been used to promote following or evading behaviours [Gar90, vdPF93, Sim94, RH02] or stepping over obstacles [Tag98]. Sims [Sim94] demonstrates creatures competing over the control of an object, evaluating fitness in head-to-head competitions. Grzeszczuk and Terzopolous [GT95] optimize a meta-control framework to generate high-level interactive behaviours from a set of low-level controllers.

There is little in the way of work on stimulus–response control that explicitly deals with motion style, or only at a very low level. Van de Panne [vdP00] states that 'determining a proper optimization metric which captures the natural qualities of human and animal motions still remains enigmatic and elusive.' Auslander *et al*. [AFP*95] demonstrate only crude style control by adding fitness terms based on average COM-height and step length. We have not found any reports that use stimulus–response network control with motion capture data as part of a fitness function. However, motion capture data has been used in methods based on *state-action maps* [SvdP05, SKL07], which can be regarded as simplified stimulus–response networks (see also Section 3.2.3).

4.3. Optimization strategies

Even though off-line optimization can be applied to any parametrized control framework, stimulus–response network control is the only method that fully depends on it. This is not only part of the evolution-inspired philosophy behind this control strategy; it is simply the only perceivable way to set the parameters of a stimulus–response network in a meaningful way.

Optimization performance is linked to the shape of the *fitness landscape*, which is a visualization of fitness as a function of the parameters that are being optimized. Roughly speaking, good fitness landscapes have smooth gradients and few local minima. An example quality of a good fitness landscape is that parameters can be altered more or less independently [RH02].

Papers that use stimulus–response network control for physics-based characters generally use *evolutionary algorithms* (EAs) for optimization. The choice of using EAs is often based on a design philosophy that states that behaviours that arise from such strategies are more natural [Ale01]. However, there is little evidence to support this claim; none of the research on stimulus–response network control compares different optimization methods. It is questionable whether EAs are efficient for optimizing motion controllers, as they heavily rely on computationally expensive fitness evaluation. Recent publications on physics-based character animation express a preference for using Covariance Matrix Adaption (CMA) [Han06] for off-line parameter optimization [WFH09, WFH10, WP10, TGTL11]. However, none of these control frameworks are based on stimulus–response networks.

Optimization performance. Apart from the choice of the type of control network and optimization technique, there are some additional techniques that can help increase optimization performance:

- *Initialization*. Results from previous optimizations may be used for initialization in subsequent optimizations.
- *Early Termination*. If a simulation is not promising in early stages, it can be terminated early in order to avoid wasteful computation [RH02].
- *Reward Shaping*. To smooth the fitness landscape, fitness functions can be designed in a way that minimizes local optima in the fitness landscape [NHR99, SvdP05].
- *Bootstrapping*. In addition to shaping the reward function, the fitness landscape can also be smoothed by gradually increasing the complexity of a task during the optimization [Gar90, SvdP05].

Increasing robustness. Auslander *et al*. [AFP*95] demonstrate that robustness of a controller can be improved by using random variation in the initial conditions during controller optimization. Wang *et al*. [WFH10] demonstrate how including noise or random perturbation during optimization can lead to more robust controllers. However, both these techniques have not been applied to stimulus–response network control strategies. Van de Panne *et al*. note that stimulus–response controllers become less robust when they are optimized for more specific conditions [vdPF93].

4.4. Summary

Despite promises based on early successes, recent achievements on motion control using stimulus–response networks are limited. Where off-line parameter optimization has shown to be an effective technique in other frameworks [CBYvdP08, YCBvdP08, WFH09, WFH10, TGTL11], we have seen no demonstration of full-body humanoid biped locomotion using stimulus–response networks. Apparently, the stimulus–response networks used for motion control are either too restrictive to allow for complex character control, or too flexible to allow for effective parameter optimization. The most distinctive quality of stimulus–response network control may very well be its capability to produce unexpected and entertaining animations (e.g. [Sim94, AF09]).

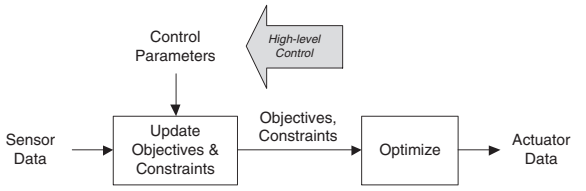

Figure 8: *Constrained dynamics optimization control.*

5. Constrained Dynamics Optimization Control

Derived from *optimal control theory* and *optimization theory*, this approach computes *on-line* the set of actuator values that are optimal for a given goal. The dynamics of a character and its environment are formulated as a set of *constraints*, while the intended behaviour of a character is defined through high-level *objectives*. The optimal set of actuator values is then acquired through *constrained optimization*. Figure 8 displays a schematic overview of this method.

There is a difference between this approach and optimization methods described in previous sections: earlier methods attempt to find *control parameters* through *off-line* trial-based optimization, whereas the methods described in this section attempt to find *actuator values* (i.e. joint torques) through constant *on-line* optimization based on the equations of motion describing the character dynamics.

The main advantage of *constrained dynamics optimization control* over *joint-space motion control* and *stimulus–response network control*, is that this approach explicitly incorporates the equations of motion in the optimization process, establishing a tight link between control and dynamics simulation. Compared to joint-space control methods, this ensures a much better prediction of the effect of applying a set of joint torques. A downside is that these control methods are more difficult to implement than previously described approaches; they require substantial knowledge of both multi-body dynamics and constrained optimization—topics with which computer animation specialists may not be familiar with. In addition, it is more difficult to model natural joint compliance (in the case of unexpected perturbations) using full-body torque optimization, when compared to joint-space control. Finally, these methods are computationally more expensive than joint-space control equivalents, typically limiting real-time performance to a single character at a time.

This approach is related to *optimal control theory*, a method that has been applied to computer animation earlier by Witkin and Kass [WK88]. In their *space–time optimization* framework, a user defines a set of constraints (e.g. a character must have a specific pose at a specific time), and a set of objectives (e.g. a character must use as little energy as possible), after which dynamically accurate motion trajectories are generated that meet these constraints and objectives. Based on this approach, several researchers have demonstrated physically realistic animations for different characters and behaviours [PW99, FP03, SHP04, LHP05, LYvdP*10]. Related to this is the work of Grzeszczuk *et al.* [GTH98], who use neural networks to learn the dynamics constraints from a physics-based character model.

Space–time optimization is not suitable for interactive applications, because it optimizes motion sequences *as a whole* [LHP06]. Any unexpected disturbance invalidates the remaining trajectory; even numerical integration errors can be considered disturbances [LvdPF96, dSAP08a]. It is computationally very expensive to re-optimize the entire motion sequence at every time step. In Section 5.2 we provide an overview of strategies developed to overcome this challenge.

5.1. Constraints and objectives

The basic task of any optimization-based motion controller is to solve the following optimization problem:

$$\text{argmin}_{\tau_t} \ \{G_1, G_2, \ldots, G_n\} \ , \ \text{subject to} \ \{C_1, C_2, \ldots, C_m\} , \tag{6}$$

where G_i are the n high-level *objectives* and C_i describe the m *constraints*. The result is a set of joint torques, τ_t, that is optimal for the given goals and constraints at time t. This optimization is performed at regular intervals, but typically at a lower rate than the update rate of the physics simulation itself.

Constraints. The conditions that all parameters must fulfil during optimization are captured in constraints. The most important set of constraints ensure that solutions are in accordance with the Newton–Euler laws of dynamics. These constraints are represented in Equation (3), which describes the relation between a set of torques and forces τ (which include the set of joint torques τ_t), and a set of generalized accelerations $\ddot{q}$. Because of its form, it is called an *equality constraint*. An example of an *inequality constraint* is the Coulomb friction of Equation (4). Inequality constraints are also used to enforce bounds or limits, such as *joint limits* or *maximum torques* (Section 2.2.1).

Objectives. The behaviour of the controller is shaped through objectives, usually through *cost functions* that need to be minimized. An example cost function that can be used for static balance measures the horizontal displacement between the character's COM and its centre of support. Other example objectives minimize energy usage, or promote tracking of a reference motion. Objectives are also referred to as *soft constraints* [SC92b].

Note that on-line cost functions are different from cost functions (or fitness functions) used in off-line optimization. The key difference is that off-line optimization shapes the entire motion, while on-line optimization methods only

involve the current time step. For example, a cost function based on maximum COM height would in *off-line optimization* promote jumping, where in *on-line optimization* it might promote tiptoeing and raising hands.

Multiple objectives can be combined into a single objective function using a weighted average. The downside of such an approach is that, apart from the additional tuning that is required, different competing objectives may interfere with each other. For instance, a motion tracking objective may interfere with a balance objective, causing a character to fall [AdSP07]. To get around this, de Lasa *et al*. [dLH09] demonstrate a solver that allows for prioritized optimization, by only meeting lower-priority objectives when they do not interfere with higher-priority objectives such as balance. Their method is based on *task-space* and *operation-space* formulations found in robotics research (e.g. [Lie77, Kha87]), which have also been used in physics-based animation research by Abe and Popović [AP06].

5.2. Control strategies

The main challenge in constrained dynamics optimization control is that the optimization must be performed on-line. We distinguish between three basic approaches to overcome this challenge:

1. Only optimize for the current situation. This approach works well for motions that do not require planning, such as static balance, or for motions constructed through finite state machines, similar to those described in Sections 3.2.1 and 3.2.2.
2. Use off-line optimized trajectories to guide on-line optimization. This approach incorporates optimized look-ahead control information of a specific motion, without having to perform full trajectory optimization on-line.
3. Constantly re-optimize future motion trajectories, reflecting the current state. To manage computational costs, these methods optimize only for a short period ahead in time and often use simplified character models.

Each of these approaches will be described in detail in the upcoming sections.

5.2.1. *Immediate optimization*

The most straightforward approach to optimization-based motion control is to use a control strategy based on objectives that can be optimized for without looking ahead. Pioneering examples of this approach can be found in the work of Stewart and Cremer [SC92b, SC92a], who define state-driven constraints to construct controllers for bipedal walking, as well as climbing and descending stairs. Later examples include the work of Abe *et al*. [AdSP07], who use a weighted objective that drives the projected COM position towards the centre of the base-of-support, while tracking a reference pose or motion. Optimal joint torques are found trough *quadratic programming*. The result is an in-place balance controller that robustly copes with change in friction or character morphology, supporting uneven foot placement and moving support surfaces. Macchietto *et al*. [MZS09] demonstrate natural-looking and robust balance controllers that use objectives for angular momentum control. They use inverse dynamics for tracking the optimized target trajectories. Wu and Zordan [WZ10] use a similar approach, but add trajectory objectives for COM and swing foot to produce stepping behaviours for balance correction. Their trajectories are based on motion capture data and analysis of the character's momentum.

Other methods use finite state machines to construct more complex behaviours. Jain *et al*. [JYL09] demonstrate controllers for side-stepping, object dodging and balancing through use of external support from walls, railings or other characters. They use a weighted set of objectives, including objectives that drive hands and feet towards specific locations. De Lasa *et al*. [dLMH10] construct robust balancing, walking and jumping controllers by combining several coordinated and state-driven low-level objectives, using prioritized optimization. They demonstrate how to tune objectives to suggest different moods and exhibit properties of natural human motion. In addition, their controllers are robust against major on-the-fly changes of character morphology.

Finally, da Silva *et al*. [dSDP09] demonstrate a framework in which different controllers with immediate optimization objectives can be combined, by weighting the contribution of different controllers based on current state and goal.

5.2.2. *Pre-optimized prediction models*

In this strategy, full motion trajectories are computed through off-line constrained optimization and adapted by an on-line controller based on the current state.

Da Silva *et al*. [dSAP08a] use a reference motion to optimize a three-link biped model for balanced locomotion. They then use a linear feedback policy to combine the model predictions with a tracking objective to generate real-time locomotion tracking. Muico *et al*. [MLPP09] use a reference motion to optimize a model of both a full-body character and ground reaction forces, and use a non-linear feedback policy for on-line tracking. Their controller can track of agile walking and running motions with sharp turns. In later research [MPP11], they increase robustness against perturbations and inclines by simultaneously tracking multiple trajectories and using a graph that describes possible blends and transitions between trajectories. Wu and Popović [WP10] use off-line optimization to compute optimal end-effector trajectories and tracking gains, which are selected by an on-line controller at the beginning of each step, based on the current state and a goal task. On-line joint torques are found through *quadratic*

programming. Their controller demonstrates robust locomotion on uneven terrain using sharp turns and backwards walking, without requiring reference motion trajectories, and can be used with common physics engines.

5.2.3. *Model predictive control*

This control strategy maintains a low-dimensional model of the character, which is used in on-line optimization to find a set of motion trajectories covering a short period ahead in time. These predictive target trajectories are constantly updated and used as input to control the high-dimensional character. The use of predictive motion trajectories enables real-time look-ahead capabilities, which are important for actions involving complex root motions, such as jumping and rolling, but also for discontinuous dynamic behaviours, such as walking and running.

This approach is related to humanoid robotics research that performs *preview control* of the *zero-moment point* (ZMP) of a character model [KKK*03b, KKK*03a, KKI06, WC06, Che07]. The ZMP is defined as the point from which a character's ground reaction force would result in a zero net moment [VB04]. A character is statically balanced if its ZMP coincides with its COP [Wie02].

Da Silva *et al*. [dSAP08b] demonstrate a motion capture tracking controller that uses a linearized model of linked rigid bodies and contact forces for predictive control. Kwon and Hodgins [KH10] demonstrate a controller for running motions, using an inverted pendulum model optimized using motion data to generate reference footstep coordinates on-line. Mordatch *et al*. [MdLH10] demonstrate robust locomotion on highly constrained and uneven terrains, using a spring-loaded inverted pendulum model (an inverted pendulum model that uses a spring to model variable length of the stance leg) for on-line planning. Their resulting COM and foot trajectories are then tracked using low-level feature objectives described by De Lasa *et al*. [dLMH10].

Ye and Liu [YL10] demonstrate a controller that can track a reference motion with on-line modification of long-term goals and completion time. They use off-line optimization to construct an abstract model that describes a non-linear relation between global motion and external contact forces. Their controller robustly responds to external perturbations and changes in step height, as long as the footstep positions in the response strategy do not deviate from the original motion. Jain and Liu [JL11b] demonstrate a model that allows on-line long-horizon planning at every time step. Their model is based on modal analysis of reference motion data and is used to construct separate control strategies for dynamically decoupled modes (an approach explored earlier by Kry *et al*. [KRFC09]). To improve performance, they use a linearized dynamics system and disregard higher-frequency modes. Their method allows robust tracking and on-line editing of a reference trajectory, but the linearized dynamics system prevents the control method to be used for high-energy motions such as running.

5.3. Summary

The recent renewed interest in constrained dynamics optimization techniques to control physics-based characters has lead to several impressive results. Some of these results have not been demonstrated using joint-space or stimulus–response methods, including tracking of captured running motions, balance control on moving surfaces, and navigation over uneven terrain. The main challenge of these methods is in the fact that optimization must be performed in real-time, which makes full trajectory planning infeasible. We have described three approaches to overcome this hurdle:

1. Describe optimization goals in such a way that they don't require looking ahead. This approach has lead to various robust balance controllers [AdSP07, MZS09, WZ10], as well as a locomotion and jumping control framework that uses prioritized optimization [dLMH10].
2. Perform off-line trajectory optimization and use the results to guide the on-line optimization process. This has resulted in robust motion tracking controllers [MLPP09, MPP11] as well as flexible navigation over uneven terrain [WP10].
3. Use a simplified model to perform look-ahead trajectory optimization on-line. This approach has resulted in controllers for agile running [KH10], locomotion on highly constrained and uneven terrains [MdLH10] and robust motion tracking [YL10, JL11b].

Implementation. The implementation of an optimization-based motion control framework is significantly more challenging than joint-space or stimulus–response methods. It requires substantial knowledge of constrained dynamics modelling and on-line optimization techniques such as quadratic programming—a skill set uncommon for developers with a background in computer animation.

The amount of tuning or pre-processing required for new characters or behaviours varies per method. Some methods require pre-processing for different motions or character morphologies (e.g. [MLPP09, MPP11]), while methods that combine low-level objectives to achieve high-level behaviours require tuning to get the behaviour right (e.g. [JYL09, dLMH10]). On the other hand, some controllers are robust against large changes in character morphology, without the need for additional tuning (e.g. [AdSP07, dLMH10]).

The computational requirements for optimization-based controllers depend on the number of objectives, the degree in which they compete [JYL09], the complexity of the character model, and the number of active external contacts [MLPP09]. In general, computational requirements for optimization-based methods are substantially higher than most joint-space

control or stimulus–response network control methods; publications typically report real-time or near real-time performance for a single character.

6. Summary and Future Directions

It is an interesting time for physics-based character animation. After two decades of floundering, the field is rapidly maturing and commercial adaptation of interactive physics-based characters appears to be within reach. As a consequence, this review paper may require regular updates.

Approaches. We have organized research on physics-based character animation into three categories, based on the primary originating research area. The first is *joint-space control*, a method derived from classic control theory that uses local feedback controllers to track kinematic targets. While this method is intuitive, the local nature of the feedback controllers requires extra attention to coordinated full-body control. The SIMBICON controller [YLvdP07] tackles several balance-related issues in an intuitive and generic way; its introduction has lead to a wide range of follow-up publications using joint-space control methods. Most notable are recent extensions that demonstrate high flexibility and robustness by incorporating global feedback control methods, such as virtual forces [CBvdP10, CKJ*11] and inverse dynamics [LKL10]. These methods allow easy authoring of new styles without the need for laborious manual tuning. Other interesting additions use off-line parameter optimization for developing new behaviours [YCBvdP08] and style control [WFH09, WFH10].

The second approach is *stimulus–response network control*, which is derived from artificial intelligence and inspired by evolutionary biology. It attempts to optimize a generic network-like control framework based on high-level fitness criteria. Even though early publications have successfully used this approach to generate striking and agile animations for low-dimensional creatures (most notably the work of Sims [Sim94]), recent years have shown little to no progress. As of today, we have found no example of successful full-body humanoid locomotion using stimulus–response networks.

The final category is *constrained dynamics optimization control*, which is derived from optimal control theory and space–time optimization. Even though the approach is relatively new in computer animation, several researches have used it to produce compelling results. Key examples are efficient balance controllers [AdSP07, MZS09, WZ10], robust motion capture tracking controllers [MLPP09, MPP11, YL10, JL11b], controllers for navigating on uneven terrain [WP10, MdLH10], and flexible feature-based controllers with parametrized style control, which automatically adapt to large changes in character morphology [dLMH10]. The main strength of this approach is that control is tightly linked with dynamics simulation, because the *equations of motion* are part of the optimization process. However, the approach is computationally more expensive than joint-space or stimulus–response methods, and successful implementation requires thorough understanding of constrained dynamics and optimization theory—topics with which many computer animation researchers may be unfamiliar with.

Applications. Physics-based characters thrive during unexpected interactions. If within limits of what a character can handle, responses to perturbations will be fully original and physically correct. Compared to kinematics-based methods that use motion capture data, physics-based controllers are not limited to animations that have equivalent real-life performances; they can just as easily be used for the animation of dangerous stunts (such as jumping head-first from a staircase), for the control of animals unsuitable for motion capture, or for creatures that do not even exist. Physics-based characters also have the potential to generalize better than kinematics-based methods. Major changes in character morphology or style can in some cases [CBvdP10, dLMH10] automatically be contained in physically accurate motion.

The main limitation of physics-based characters is the lack of control. This is perhaps most evident in applications that require responsive user-control or demand tight control over style. Despite great advances, we do not feel physics-based animation has the potential to replace kinematics-based approaches in such applications.

Future directions. Even though joint-space motion control is the approach with the longest history in physics-based character animation, it is only recently that this approach has resulted in stable, generic motion control frameworks (e.g. [CBvdP10, CKJ*11, LKL10]). This is largely due to incorporation of global control methods, such as virtual forces; we expect a similar approach can lead to robust controllers for various other types of behaviours. In addition, it could be interesting to see these recent frameworks be subject to parameter optimization using high-level objectives, similar to the work of Wang *et al*. [WFH09, WFH10].

Although stimulus–response network control appears to be unsuitable for control of anything beyond the most simple 3D biped character, recent publications show that the idea of relying on high-level optimization for motion control still has merit [WFH09, WFH10, TGTL11, WHD12]—as long as the control framework somehow fits the task in mind and the parameter space is limited.

Optimization-based control frameworks have demonstrated impressive results for various types of behaviour, even though the application of this approach in interactive physics-based character animation is relatively new. We expect to see several new optimization-based strategies to arise in the near future, especially those based on predictive control

models. We consider the more interesting direction to be the development of controllers that do not rely on motion capture data (e.g. [dLMH10, MdLH10, WP10]), as such a reliance limits applicability and poses less of an immediate advantage over kinematics-based methods. A possibly interesting direction could be the use of off-line parameter optimization based on high-level goals (similar to [WFH09, WFH10]) applied to low-level parametrized control frameworks such as [dLMH10].

One type of behaviour that we believe deserves more attention is the act of gracefully falling down and robustly getting back up (a theme briefly explored in [FvdPT01]). Even the most robust controller will fail to maintain balance in some conditions, often leading to animations that are perhaps humorous, but undesirable in production games or movies. We imagine that commercial animation frameworks require a physics-based character to be able to fall and get back up, both robustly and naturally (i.e. without the use of external forces). The use of a natural injury metric as an optimization criterion could be helpful for the development of such controllers [GVE11].

Another more generic trend that we witness is the use of more advanced character models. If designed correctly, such models can take away much of the burden of control from a motion controller, for example by incorporating mechanisms for passive energy storage-and-release [KH10] or soft contact modelling [JL11a]. We expect such model improvements will also benefit efforts to improve controllers through off-line parameter optimization based on high-level objectives—a direction already hinted at by Liu *et al.* [LHP05].

We finally wish to address the need for proper benchmarking, to allow quantitative comparison between controllers. Van de Panne [vdP00] proposes the idea of a *Virtual Olympics* event, in which physics-based characters compete with each other in various contests, with restrictions on body morphology and energy consumption. The field of physics-based character animation would greatly benefit if such a competition would become a recurring event, similar to the *RoboCup* [KAK*97] initiative.

Acknowledgements

We would like to thank the anonymous reviewers, whose insightful comments greatly contributed to improving the quality of this review.

REFERENCES

[AdSP07] Abe Y., Da silva M., Popović J.: Multiobjective control with frictional contacts. In *Proceedings of the ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Diego, CA, USA, August 2007), pp. 249–258.

[AF09] Allen B., Faloutsos P.: Evolved controllers for simulated locomotion. In *Motion in Games*. A. Egges, R. Geraerts and M. Overmars (Eds.). Springer, Berlin/Heidelberg (2009), pp. 219–230.

[AFP*95] Auslander J., Fukunaga A., Partovi H., Christensen J., Hsu L., Reiss P., Shuman A., Marks J., Ngo J. T.: Further experience with controller-based automatic motion synthesis for articulated figures. *ACM Transactions on Graphics 14*, 4 (October 1995), 311–336.

[AG85] Armstrong W., Green M.: The dynamics of articulated rigid bodies for purposes of animation. *The Visual Computer 1*, 4 (1985), 231–240.

[Ale01] Alexander R. M.: Design by numbers. *Nature 412*, 6847 (August 2001), 591.

[AP93] Anderson F. C., Pandy M. G.: Storage and utilization of elastic strain energy during jumping. *Journal of biomechanics 26*, 12 (December 1993), 1413–1427.

[AP06] Abe Y., Popović J.: Interactive animation of dynamic manipulation. In *Proceedings of the 2006 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (Vienna, Austria, September 2006), pp. 195–204.

[BB07] Boeing A., Bräunl T.: Evaluation of real-time physics simulation systems. In *Proceedings of the Int. Conf. on Computer Graphics and Interactive Techniques in Australia and Southeast Asia* (Perth, Australia, December 2007), pp. 281–288.

[Bis94] Bishop C.: Neural networks and their applications. *Review of Scientific Instruments 65*, 6 (1994), 1803–1832.

[BSH99] Bodenheimer B., Shleyfman A., Hodgins J.: The effects of noise on the perception of animated human running. In *Proceedings of the Computer Animation and Simulation, Eurographics Animation Workshop* (Milano, Italy, September 1999), pp. 53–63.

[BT08] Byl K., Tedrake R.: Approximate optimal control of the compass gait on rough terrain. In *Proceedings of the Int. Conf. on Robotics and Automation* (Pasadena, CA, USA, May 2008), 1258–1263.

[CBvdP09] Coros S., Beaudoin P., van de Panne M.: Robust Task-based Control Policies for Physics-based Characters. *ACM Transactions on Graphics 28*, 5 (2009), 170.

[CBvdP10] Coros S., Beaudoin P., van de Panne M.: Generalized biped walking control. *ACM Transactions on Graphics 29* (2010), 130.

[CBYvdP08] Coros S., Beaudoin P., Yin K. K., van de Panne M.: Synthesis of constrained walking skills. *ACM Transactions on Graphics 27*, 5 (December 2008), 113.

[Che07] CHESTNUTT J.: *Navigation Planning for Legged Robots*. PhD Thesis, Carnegie Mellon University, December 2007.

[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 (2011), 59.

[CO] COUMANS E.: Bullet physics library. 2006, www.bulletphysics.org

[Cou01] COUTINHO M.: *Dynamic Simulations of Multibody Systems*. Springer Verlag, New York, NY, USA, 2001.

[CRTW05] COLLINS S., RUINA A., TEDRAKE R., WISSE M.: Efficient bipedal robots based on passive-dynamic walkers. *Science (New York, N.Y.) 307*, 5712 (2005), 1082–1085.

[De 96] DE LEVA P.: Adjustments to Zatsiorsky-Seluyanov's segment inertia parameters. *Journal of Biomechanics 29*, 9 (1996), 1223–1230.

[dLH09] DE LASA M., HERTZMANN A.: Prioritized optimization for task-space control. In *proceedings of the 2009 IEEE/RSJ Int. Conf. on Intelligent Robots and Systems 3* (October2009), pp. 5755–5762.

[dLMH10] DE LASA M., MORDATCH I., HERTZMANN A.: Feature-Based Locomotion Controllers. *ACM Transactions on Graphics 29*, 3 (2010), 131.

[dSAP08a] DA SILVA M., ABE Y., POPOVIĆ J.: Interactive simulation of stylized human locomotion. *ACM Transactions on Graphics (SIGGRAPH) 27*, 3 (August 2008), 1–10.

[dSAP08b] DA SILVA M., ABE Y., POPOVIC J.: Simulation of human motion data using short-horizon model-predictive control. *Computer Graphics Forum 27*, 2 (2008), 371–380.

[dSDP09] DA SILVA M., DURAND F., POPOVIĆ J.: Linear Bellman combination for control of character animation. *ACM Transactions on Graphics (SIGGRAPH) 28*, 3 (July 2009), 1–10.

[EMHvdB07] ERDEMIR A., MCLEAN S., HERZOG W., VAN DEN BOGERT A.: Model-based estimation of muscle forces exerted during movements. *Clinical Biomechanics 22*, 2 (2007), 131–154.

[Fal01] FALOUTSOS P.: The virtual stuntman: Dynamic characters with a repertoire of autonomous motor skills. *Computers & Graphics 25*, 6 (December 2001), 933–953.

[FDCM97] FAURE F., DEBUNNE G., CANI M., MULTON F.: Dynamic analysis of human walking. *Computer Animation and Simulation* (1997), pp. 53–65.

[Fea08] FEATHERSTONE R.: *Rigid Body Dynamics Algorithms*. Springer-Verlag, New York Inc., 2008.

[FP03] FANG A. C., POLLARD N. S.: Efficient synthesis of physically valid human motion. *ACM Transactions on Graphics 22*, 3 (July 2003), 417–426.

[Fro79] FROHLICH C.: Do springboard divers violate angular momentum conservation? *American Journal of Physics 47*, 0002-9505 (1979), 583–592.

[FvdPT01] FALOUTSOS P., VAN DE PANNE M., TERZOPOULOS D.: Composable controllers for physics-based character animation. In *ACM SIGGRAPH Papers* (Los Angeles, CA, USA, 2001), pp. 251–260.

[Gar90] GARIS H. D.: Genetic programming: Building artificial nervous systems with genetically programmed neural network modules. In *Proceedings of the 7th Int. Conf. on Machine Learning* (Austin, TX, USA, July 1990), p. 207.

[GK04] GOSWAMI A., KALLEM V.: Rate of change of angular momentum and balance maintenance of biped robots. In *IEEE International Conference on Robotics and Automation* (New Orleans, LA, USA, April 2004), vol. 4, pp. 3785–3790.

[GT95] GRZESZCZUK R., TERZOPOULOS D.: Automated learning of muscle-actuated locomotion through control abstraction. In *ACM SIGGRAPH Papers Conference Proceedings* (Los Angeles, CA, USA, August 1995), pp. 63–70.

[GTH98] GRZESZCZUK R., TERZOPOULOS D., HINTON G.: Neuroanimator: Fast neural network emulation and control of physics-based models. In *ACM SIGGRAPH Papers* (1998), pp. 9–20.

[GvdBvBE10] GEIJTENBEEK T., VAN DEN BOGERT A., VAN BASTEN B., EGGES A.: Evaluating the physical realism of character animations using musculoskeletal models. In *Proceedings of the Int. Conf. on Motion in Games* (Zeist, The Netherlands, November 2010), pp. 11–22.

[GVE11] GEIJTENBEEK T., VASILESCU D., EGGES A.: Injury Assessment for Physics-Based Characters. In *Proceedings of the Int. Conf. on Motion in Games* (Edinburgh, UK, November 2011), pp. 74–85.

[Han06] HANSEN N.: The CMA evolution strategy: A comparing review. In *Towards a New Evolutionary Computation*. (J. Lozano, P. Larrañga, I. Inza and E. Bengoetxea (Eds.). Springer-Verlag, New York, NY, USA 2006), pp. 75–102.

[HMOA03] HASE K., MIYASHITA K., OK S., ARAKAWA Y.: Human gait simulation with a neuromusculoskeletal model

and evolutionary computation. *The Journal of Visualization and Computer Animation 14*, 2 (2003), 73–92.

[HMPH05] Hofmann a., Massaquoi S., Popovic M., Herr H.: A sliding controller for bipedal balancing using integrated movement of contact and non-contact limbs. In *Proceedings of the Int. Conf. on Intelligent Robots and Systems* (2005), vol. 2, IEEE, pp. 1952–1959.

[Hod91] Hodgins J.: Biped gait transitions. In *Proceedings of the IEEE International Conference on Robotics and Automation* (Sacramento, CA, USA, April 1991), pp. 2092–2097.

[Hog90] Hogan N.: Mechanical impedance of single-and multi-articular systems. In *Multiple Muscle Systems: Biomechanics and Movement Organization*. J. Winters and S. Woo (Eds.). Springer-Verlag, New York, NY, USA (1990), p. 149.

[HP97] Hodgins J. K., Pollard N. S.: Adapting simulated behaviors for new characters. In *ACM SIGGRAPH Papers* (Los Angeles, CA, USA, August 1997), pp. 153–162.

[HRS94] Hollars M., Rosenthal D., Sherman M.: *SD/Fast User's Manual*. Symbolic Dynamics Inc, Mountain View, CA, USA, 1994.

[HWBO95] Hodgins J. K., Wooten W. L., Brogan D. C., O'brien J. F.: Animating human athletics. In *ACM SIGGRAPH Papers* (Los Angeles, CA, USA, August 1995), pp. 71–78.

[HZ11] Hertzmann A., Zordan V.: Physics-Based Characters. *IEEE Computer Graphics and Applications 31*, 4 (2011), 20–21.

[IWZL09] Ishigaki S., White T., Zordan V. B., Liu C. K.: Performance-based control interface for character animation. *ACM Transactions on Graphics 28*, 3 (July 2009), 61.

[JL11a] Jain S., Liu C.: Controlling physics-based characters using soft contacts. *ACM Transactions on Graphics 30*, 6 (2011), 163.

[JL11b] Jain S., Liu C.: Modal-space control for articulated characters. *ACM Transactions on Graphics 30*, 5 (2011), 118.

[JYL09] Jain S., Ye Y., Liu C. K.: Optimization-based interactive motion synthesis. *ACM Transactions on Graphics 28*, 1 (2009), 10.

[KAK*97] Kitano H., Asada M., Kuniyoshi Y., Noda I., Osawa E., Matsubara H.: RoboCup: A challenge problem for AI. *AI Magazine 18*, 1 (1997), 73–85.

[KB96] Ko H., Badler N.: Animating human locomotion with inverse dynamics. *IEEE Computer Graphics and Applications 16* (1996), 50–59.

[KH10] Kwon T., Hodgins J.: Control systems for human running using an inverted pendulum model and a reference motion capture sequence. In *Proceedings of the ACM SIGGRAPH/Eurographics Symp. on Computer Animation* (Madrid, Spain, July 2010), pp. 129–138.

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

[KKI06] Kudoh S., Komura T., Ikeuchi K.: Stepping motion for a human-like character to maintain balance against large perturbations. *IEEE Int. Conf. Robotics and Automation* (Orlando, FL, USA, May 2006).

[KKK*03a] Kajita S., Kanehiro F., Kaneko K., Fujiwara K., Harada K., Yokoi K., Hirukawa H.: Biped walking pattern generation by using preview control of zero-moment point. In *IEEE Int. Conf. on Robotics and Automation* (Taipei, Taiwan, September 2003), vol. 2, pp. 1620–1626.

[KKK*03b] Kajita S., Kanehiro F., Kaneko K., Fujiwara K., Yokoi K., Hirukawa H.: Biped walking pattern generation by a simple three-dimensional inverted pendulum model. *Advanced Robotics 17*, 2 (2003), 131–147.

[KL96] Kane T., Levinson D.: *Dynamics Online: Theory and Implementation with AUTOLEV*. Online Dynamics, Inc., Sunnyvale, CA, USA, 1996.

[KLM96] Kaelbling L., Littman M., Moore A.: Reinforcement learning: A survey. *Journal of Artificial Intelligence Research 4* (1996), 237–285.

[KMB96] Kokkevis E., Metaxas D., Badler N.: User-controlled physics-based animation for articulated figures. In *Proceedings of the Computer Animation* (IEEE Computer Society, Geneva, Switzerland 1996), vol. 96, pp. 16–25.

[KP11] Kim J., Pollard N.: Direct Control of Simulated Nonhuman Characters. *Computer Graphics and Applications, IEEE 31*, 4 (2011), 56–65.

[KRFC09] Kry P., Reveret L., Faure F., Cani M.-P.: Modal Locomotion: Animating Virtual Characters with Natural Vibrations. *Computer Graphics Forum 28*, 2 (April 2009), 289–298.

[KSnL05] Kelly R., Santibáñez V., Loria A.: *Control of Robot Manipulators in Joint Space*. Springer-Verlag London, UK, 2005.

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

[LHP06] Liu C. K., Hertzmann A., Popović Z.: Composition of complex optimal multi-character motions. In *ACM SIGGRAPH/Eurographics Symp. on Computer Animation* (Vienna, Austria, September 2006), pp. 215–222.

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

[LKL10] Lee Y., Kim S., Lee J.: Data-driven biped control. *ACM Transactions on Graphics 29*, 4 (July 2010), 129.

[LST09] Lee S., Sifakis E., Terzopoulos D.: Comprehensive biomechanical modeling and simulation of the upper body. *ACM Transactions on Graphics 28*, 4 (August 2009), 99.

[LvdPF96] Laszlo J., van de Panne M., Fiume E.: Limit cycle control and its application to the animation of balancing and walking. In *ACM SIGGRAPH Papers* (New Orleans, LA, USA, August 1996), pp. 155–162.

[LvdPF00] Laszlo J., van de Panne M., Fiume E.: Interactive control for physically-based animation. In *ACM SIGGRAPH Papers* (New Orleans, LA, USA, August 2000), pp. 201–208.

[LWZB90] Lee P., Wei S., Zhao J., Badler N. I.: Strength guided motion. *ACM SIGGRAPH Computer Graphics 24*, 4 (September 1990), 253–262.

[LYvdP*10] Liu L., Yin K., van de Panne M., Shao T., Xu W.: Sampling-based contact-rich motion control. *ACM Transactions on Graphics 29* (July 2010), 128.

[MAEC07] Morimoto J., Atkeson C. G., Endo G., Cheng G.: Improving humanoid locomotive performance with learnt approximated dynamics via Gaussian processes for regression. In *Proceedings of the Int. Conf. on Intelligent Robots and Systems* (San Diego, CA, USA, October 2007), 4234–4240.

[McG90] Mcgeer T.: Passive dynamic walking. *The International Journal of Robotics Research 9*, 2 (January 1990), 62–82.

[MdLH10] Mordatch I., De Lasa M., Hertzmann A.: Robust Physics-Based Locomotion Using Low-Dimensional Planning. *ACM Transactions on Graphics 29*, 4 (2010), 71.

[MI97] Mussa Ivaldi F.: Nonlinear force fields: A distributed system of control primitives for representing and learning movements. In *Int. Symp. on Computational Intelligence in Robotics and Automation* (Monterey, CA, USA, 1997), IEEE,pp. 84–90.

[MLPP09] Muico U., Lee Y., Popović J., Popović Z.: Contact-aware nonlinear control of dynamic characters. *ACM Transactions on Graphics 28*, 3 (July 2009) 81:1–81:9.

[MPP11] Muico U., Popović J., Popović Z.: Composite control of physically simulated characters. *ACM Transactions on Graphics 30*, 3 (May 2011), 16.

[MTK98] Miyakoshi S., Taga G., Kuniyoshi Y.: Three dimensional bipedal stepping motion using neural oscillators-towards humanoid motion in the real world. *Int. Conf. on Intelligent Robots and Systems 18*, 1 (1998), 87–97.

[MWDM98] Mataric M., Williamson M., Demiris J., Mohan A.: Behavior-based primitives for articulated control. In *Proceedings of the 5th Int. Conf. on Simulation of Adaptive Behavior* (Zurich, Switzerland, 1998), pp. 165–170.

[MY11] Murai A., Yamane K.: A neuromuscular locomotion controller that realizes human-like responses to unexpected disturbances. *International Conference on Robotics and Automation (ICRA) 1*, 2-3 (2011), 1997–2002.

[MZS09] Macchietto A., Zordan V., Shelton C. R.: Momentum control for balance. *ACM Transactions on Graphics 28*, 3 (2009).

[MZW99] Matarić M., Zordan V., Williamson M.: Making complex articulated agents dance. *Autonomous Agents and Multi-Agent Systems 2*, 1 (1999), 23–43.

[NF02] Neff M., Fiume E.: Modeling tension and relaxation for computer animation. In *Proceedings of the 2002 ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (San Antonio, TX, USA, 2002), ACM, pp. 81–88.

[NHR99] Ng A. Y., Harada D., Russell S.: Policy invariance under reward transformations: Theory and application to reward shaping. In *Proc. of the 16th Int. Conf. on Machine Learning* (Bled, Slovenia, 1999), pp. 278–287.

[NM93] Ngo J., Marks J.: Physically realistic motion synthesis in animation. *Evolutionary Computation 1*, 3 (September 1993), 235–268.

[Nov98] Novacheck T.: The biomechanics of running. *Gait & Posture 7*, 1 (1998), 77–95.

[NVCNZ08] Nunes R. F., Vidal C. A., Cavalcante-neto J., Zordan V. B.: Simple feedforward control for

responsive motion capture-driven simulations. *Advances in Visual Computing*, LNCS Vol. 5358 (2008), 488–497.

[NWB*10] Nguyen N., Wheatland N., Brown D., Parise B., Liu C. K., Zordan V.: Performance capture with physical interaction. In *ACM SIGGRAPH/Eurographics Symp. on Computer Animation* (Madrid, Spain, July 2010), pp. 189–195.

[OM01] Oshita M., Makinouchi A.: A dynamic motion control technique for human-like articulated figures. *Computer Graphics Forum 20*, 3 (2001), 192–203.

[Ott03] Otten E.: Inverse and forward dynamics: Models of multi-body systems. *Philosophical Transactions of the Royal Society of London. Series B, Biological Sciences 358*, 1437 (September 2003), 1493–1500.

[PCTD01] Pratt J., Chew C., Torres A., Dilworth P.: Virtual model control: An intuitive approach for bipedal locomotion. *International Journal of Robotics Research 20*, 2 (February 2001), 129–143.

[PLF*00] Pollack J., Lipson H., Ficici S., Funes P., Hornby G., Watson R.: Evolutionary techniques in physical robotics. In *Evolvable Systems: From Biology to Hardware: Third Int. Conf.* (Berlin/Heidelberg, 2000), Springer Verlag, p. 175.

[PP10] Pejsa T., Pandzic I.: State of the Art in Example-Based Motion Synthesis for Virtual Characters in Interactive Applications. *Computer Graphics Forum 29*, 1 (2010), 202–226.

[PW99] Popović Z., Witkin A.: Physically based motion transformation. In *ACM SIGGRAPH Papers* (Los Angeles, CA, USA, 1999), pp. 11–20.

[PZ05] Pollard N. S., Zordan V. B.: Physically based grasping control from example. In *ACM SIGGRAPH/Eurographics Symp. on Computer Animation* (Los Angeles, CA, USA, 2005), ACM Press, p. 311.

[RH91] Raibert M. H., Hodgins J. K.: Animation of dynamic legged locomotion. *ACM SIGGRAPH Computer Graphics 25*, 4 (July 1991), 349–358.

[RH02] Reil T., Husbands P.: Evolution of central pattern generators for bipedal walking in a real-time physics environment. *IEEE Transactions on Evolutionary Computation 6*, 2 (April 2002), 159–168.

[RM01] Reil T., Massey C.: Biologically inspired control of physically simulated bipeds. *Theory in Biosciences 120*, 3-4 (December 2001), 327–339.

[SC92a] Stewart A., Cremer J.: Animation of 3d human locomotion: Climbing stairs and descending stairs. In *Proceedings of the 3rd Eurographics workshop on Animation* (Cambridge, UK, 1992), pp. 1–18.

[SC92b] Stewart A., Cremer J.: Beyond keyframing: An algorithmic approach to animation. *Graphics Interface Conference Proceedings* (Vancouver, Canada, 1992), 273–281.

[SCAF07] Shapiro A., Chu D., Allen B., Faloutsos P.: A dynamic controller toolkit. In *Proceedings of the 2007 ACM SIGGRAPH Symposium on Video Games* (San Diego, CA, USA, 2007), ACM, p. 20.

[SFNTH05] Shapiro A., Faloutsos P., Ng-thow-hing V.: Dynamic Animation and Control Environment. In *Proceedings of Graphics Interface 2005* (Victoria, Canada, May 2005), pp. 61–70.

[SH08] Shiratori T., Hodgins J.: Accelerometer-based user interfaces for the control of a physically simulated character. *ACM Transactions on Graphics 27*, 5 (2008), 123.

[SHP04] Safonova A., Hodgins J., Pollard N.: Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *ACM SIGGRAPH Papers* (Los Angeles, CA, USA, 2004), pp. 514–521.

[Sim94] Sims K.: Evolving virtual creatures. In *ACM SIGGRAPH Papers* (Orlando, FL, USA, 1994), pp. 15–22.

[SKL07] Sok K., Kim M., Lee J.: Simulating biped behaviors from human motion data. *ACM Transactions on Graphics 26*, 3 (2007), 107.

[SKP08] Sueda S., Kaufman A., Pai D. K.: Musculotendon simulation for hand animation. *ACM Transactions on Graphics 27*, 3 (2008), 83.

[Smi06] Smith R.: Open Dynamics Engine User Guide v0.5, 2006.

[SPF03] Shapiro A., Pighin F., Faloutsos P.: Hybrid control for interactive character animation. *Pacific Graphics Conference Proceedings* (Canmore, Alberta, Canada 2003), 455–461.

[Sta04] Stanley K.: *Efficient Evolution of Neural Networks Through Complexification*. The University of Texas at Austin, 2004.

[SvdP05] Sharon D., van de Panne M.: Synthesis of controllers for stylized planar bipedal walking. In *Proc. of the Int. Conf. on Robotics and Automation* (Barcelona, Spain, 2005), 2387–2392.

[Tag95] TAGA G.: A model of the neuro-musculo-skeletal system for human locomotion. *Biological Cybernetics 73*, 2 (1995), 97–111.

[Tag98] TAGA G.: A model of the neuro-musculo-skeletal system for anticipatory adjustment of human locomotion during obstacle avoidance. *Biological Cybernetics 78*, 1 (1998), 9–17.

[TGTL11] TAN J., GU Y., TURK G., LIU C.: Articulated swimming creatures. *ACM Transactions on Graphics 30*, 4 (2011), 58.

[TLC*09] TSAI Y., LIN W., CHENG K., LEE J., LEE T.: Real-Time Physics-Based 3D Biped Character Animation Using an Inverted Pendulum Model. *IEEE Transactions on Visualization and Computer Graphics 16*, 2 (2009), 325–337.

[TLT11] TAN J., LIU K., TURK G.: Stable proportional-derivative controllers. *Computer Graphics and Applications, IEEE*, 99 (2011), 34–44.

[TSR01] TAKAHASHI C. D., SCHEIDT R. A., REINKENSMEYER D. J.: Impedance control and internal model formation when reaching in a randomly varying dynamical environment. *Journal of Neurophysiology 86*, 2 (August 2001), 1047–1051.

[TYS91] TAGA G., YAMAGUCHI Y., SHIMIZU H.: Self-organized control of bipedal locomotion by neural oscillators in unpredictable environment. *Biological Cybernetics 65*, 3 (1991), 147–159.

[TZS04] TEDRAKE R., ZHANG T., SEUNG H.: Stochastic policy gradient reinforcement learning on a simple 3D biped. In *Proc. of the Int. Conf. on Intelligent Robots and Systems* (Sendai, Japan, 2004), pp. 2849–2854.

[van94] VAN DER HELM F. C. T.: A finite element musculoskeletal model of the shoulder mechanism. *Journal of Biomechanics 27*, 5 (1994), 551–553.

[VB04] VUKOBRATOVIC M., BOROVAC B.: Zero-moment point-thirty five years of its life. *International Journal of Humanoid Robotics 1*, 1 (2004), 157–173.

[VDJ99] VAUGHAN C., DAVIS B., JEREMY C.: *Dynamics of Human Gait*. Human Kinetics Publishers Champaign, Cape Town, South Africa, 1999.

[vdP96] VAN DE PANNE M.: Parameterized gait synthesis. *IEEE Computer Graphics and Applications 16*, 2 (1996), 40–49.

[vdP00] VAN DE PANNE M.: Control for simulated human and animal motion. *Annual Reviews in Control 24*, 1 (2000) 189–199.

[vdPF93] VAN DE PANNE M., FIUME E.: Sensor-actuator networks. In *ACM SIGGRAPH Papers* (Anaheim, CA, USA 1993), pp. 335–342.

[vdPKF94] VAN DE PANNE M., KIM R., FIUME E.: Virtual wind-up toys for animation. *Graphics Interface Conference Proceedings* (Banff, Alberta, Canada, 1994) 208–215.

[vdPL95] VAN DE PANNE M., LAMOURET A.: Guided optimization for balanced locomotion. *Computer animation and simulation 95* (1995), 165–177.

[VVE*10] VAN WELBERGEN H., VAN BASTEN B. J. H., EGGES A., RUTTKAY Z. M., OVERMARS M. H.: Real time animation of virtual humans: A trade off between naturalness and control. *Computer Graphics Forum 29*, 8 (December 2010), 2530–2554.

[WC06] WIEBER P., CHEVALLEREAU C.: Online adaptation of reference trajectories for the control of walking systems. *Robotics and Autonomous Systems 54*, 7 (2006), 559–566.

[WFH09] WANG J., FLEET D., HERTZMANN A.: Optimizing walking controllers. *ACM Transactions on Graphics 28*, 5 (2009), 168.

[WFH10] WANG J., FLEET D., HERTZMANN A.: Optimizing Walking Controllers for Uncertain Inputs and Environments. *ACM Transactions on Graphics 29*, 4 (2010), 73.

[WGF08] WEINSTEIN R., GUENDELMAN E., FEDKIW R.: Impulse-based control of joints and muscles. *IEEE transactions on visualization and computer graphics 14*, 1 (2008), 37–46.

[WH00] WOOTEN W., HODGINS J.: Simulating leaping, tumbling, landing and balancing humans. In *IEEE Int. Conf. on Robotics and Automation* (San Francisco, CA, USA, April 2000), vol. 1, pp. 656–662.

[WHD12] WANG J., HAMNER S., DELP S.: Optimizing Locomotion Controllers Using Biologically-Based Actuators and Objectives. *ACM Transactions on Graphics 31*, 4 (2012), 25.

[Wie02] WIEBER P.: On the stability of walking systems. In *Proceedings of the International Workshop on Humanoid and Human Friendly Robotics* (Tsukuba, Japan, December 2002), pp. 53–59.

[Wil87] WILHELMS J.: Using dynamic analysis for realistic animation of articulated bodies. *Computer Graphics and Applications, IEEE 7*, 6 (1987), 12–27.

[WJM06] WROTEK P., JENKINS O., MCGUIRE M.: Dynamo: Dynamic, data-driven character control with adjustable balance. *Proceedings of the 2006 ACM SIGGRAPH Symp. on Videogames 1*, 212 (2006), 61–70.

[WK88] WITKIN A., KASS M.: Spacetime constraints. In *ACM SIGGRAPH Computer Graphics* (August 1988), ACM, pp. 159–168.

[WL97] WOODS R., LAWRENCE K.: *Modeling and Simulation of Dynamic Systems*. Prentice Hall Upper Saddle River, New Jersey, 1997.

[WP10] WU J.-C., POPOVIC Z.: Terrain-Adaptive Bipedal Locomotion Control. *ACM Transactions on Graphics 29*, 4 (2010), 72.

[WZ10] WU C., ZORDAN V.: Goal-Directed Stepping with Momentum Control. In *ACM SIGGRAPH/Eurographics Symp. on Computer Animation* (Madrid, Spain, 2010), pp. 113–118.

[YCBvdP08] YIN K. K., COROS S., BEAUDOIN P., VAN DE PANNE M.: Continuation methods for adapting simulated skills. *ACM Transactions on Graphics 27*, 3(2008), 81.

[YCP03] YIN K. K., CLINE M. B., PAI D. K.: Motion perturbation based on simple neuromotor control models. In *Proceedings of the Pacific Conf. on Computer Graphics and Applications* (Canmore, Alberta, Canada, October 2003), pp. 445–449.

[YL10] YE Y., LIU C.: Optimal feedback control for character animation using an abstract model. *ACM Transactions on Graphics 29*, 4 (July 2010), 74.

[YLS04] YANG P., LASZLO J., SINGH K.: Layered dynamic control for interactive character swimming. In *ACM SIGGRAPH/Eurographics Symp. on Computer Animation* (Grenoble, France, August 2004), pp. 39–47.

[YLvdP07] YIN K. K., LOKEN K., VAN DE PANNE M.: Simbicon: Simple biped locomotion control. *ACM Transactions on Graphics 26*, 3 (2007), 105.

[ZH02] ZORDAN V. B., HODGINS J. K.: Motion capture-driven simulations that hit and react. In *ACM SIGGRAPH/Eurographics Symp. on Computer Animation* (New York, NY, USA, 2002), ACM Press, p. 89.

[ZMCF05] ZORDAN V., MAJKOWSKA A., CHIU B., FAST M.: Dynamic response for motion capture animation. *ACM Transactions on Graphics 24*, 3 (2005), 697–701.

[ZMM*07] ZORDAN V., MACCHIETTO A., MEDINA J., SORIANO M., WU C. C.: Interactive dynamic response for games. In *Proc. of the 2007 ACM SIGGRAPH Symp. on Video Games* (San Diego, CA, USA, 2007), ACM, p. 14.

[ZSC90] ZATSIORSKY V., SELUYANOV V., CHUGUNOVA L.: Methods of determining mass-inertial characteristics of human body segments. In *Contemporary Problems of Biomechanics* G.G. Chernyi and S.A. Regirer (Eds.). (CRC Press, Boca Raton, FL, USA, 1990), pp. 272–291.

[ZvdP05] ZHAO P., VAN DE PANNE M.: User interfaces for interactive control of physics-based 3D characters. In *Symposium on Interactive 3D Graphics and Games* (2005), pp. 87–94.