Physical Simulation of Balance Recovery after a Push

Alexis Jensen, Thomas Chatagnon, Niloofar Khoshsiyar, Daniele Reda,
Michiel van de Panne, Charles Pontonnier, Julien Pettré

Physical Simulation of Balance Recovery after a Push

Alexis Jensen
Univ Rennes, CNRS, INRIA,
IRISA-UMR 6074
Rennes, 35000, France
alexis.jensen@irisa.fr

Thomas Chatagnon
Univ Rennes, CNRS, INRIA,
IRISA-UMR 6074
Rennes, 35000, France
thomas.chatagnon@irisa.fr

Niloofar Khoshsiyar
University of British Columbia
Vancouver, Canada
nkhosh@cs.ubc.ca

Daniele Reda
University of British Columbia
Vancouver, Canada
dreda@cs.ubc.ca

Michiel van de Panne
University of British Columbia
Vancouver, Canada
van@cs.ubc.ca

Charles Pontonnier
Univ Rennes, CNRS, INRIA,
IRISA-UMR 6074
Rennes, 35000, France
charles.pontonnier@irisa.fr

Julien Pettré
Univ Rennes, CNRS, INRIA,
IRISA-UMR 6074
Rennes, 35000, France
julien.pettre@irisa.fr

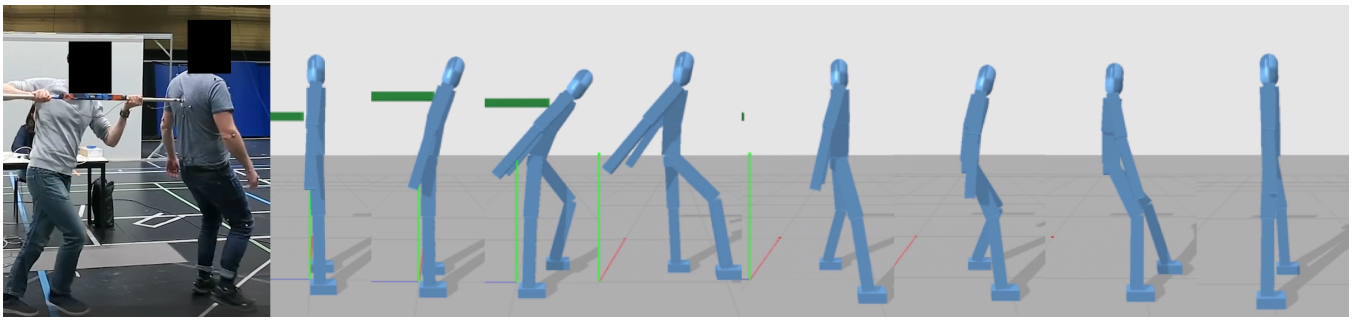

Figure 1: Experiment and simulation of balance recovery after a push with a pole

ABSTRACT

Our goal is to simulate how humans recover balance after external perturbation, e.g., being pushed. While different strategies can be adopted to achieve balance recovery, we particularly aim at replicating how humans combine the control of their support area with the control of their body movement to regain balance when it is necessary. We develop a physics-based approach to simulate balance recovery, with two main contributions to achieve our goal: a foot control technique to adjust the shape of a character's support zone to the motion of its center of mass (*CoM*), and the dynamic control of the *CoM* to maintain its vertical projection in this same zone. We also calibrate the simulation by optimisation, before validating our results against experimental data.

CCS CONCEPTS

• **Computing methodologies** → **Real-time simulation**.

KEYWORDS

physics-based simulation, biomechanics, animation, controller-based, experimental data,balance recovery

ACM Reference Format:
Alexis Jensen, Thomas Chatagnon, Niloofar Khoshsiyar, Daniele Reda, Michiel van de Panne, Charles Pontonnier, and Julien Pettré. 2023. Physical Simulation of Balance Recovery after a Push. In *ACM SIGGRAPH Conference on Motion, Interaction and Games (MIG '23), November 15–17, 2023, Rennes, France*. ACM, New York, NY, USA, 11 pages. https://doi.org/10.1145/3623264.3624448

MIG '23, November 15–17, 2023, Rennes, France

ACM ISBN 979-8-4007-0393-5/23/11…$15.00
https://doi.org/10.1145/3623264.3624448

1 INTRODUCTION

Simulating the behaviour of humans during physical interactions with their environment has been a long-standing goal in computer animation. In this work, we are interested in simulating how humans respond to moderate to strong local pushes and realistically estimating their global motion after such a perturbation. This is of interest, for example, in understanding how motion can propagate from one person to another and generate large scale motion in crowds, but for now we consider the case of a single human subject to an external force applied to his or her body.

When standing still, humans rest on their two feet in an upright position. On a flat surface, static balance is achieved by keeping the projection of the centre of mass (*CoM*) on the ground within the limits of the Base of Support *BoS* (the convex hull of the foot-floor contact points). After an external push, several movement strategies are known in humans to achieve balance recovery [Cheng and Yeh 2015]. We focus here on stepping strategies that occur after moderate to strong push forces. Stepping strategy is to extend the *BoS*, and therefore combines the control of the position of the feet together with the one of the *CoM* to satisfy the balance conditions.

Thus, to understand better the global motion resulting from external perturbations, our goal is to dynamically control the motion of a character to simulate steeping strategy for balance recovery. Our approach is to control the joint angles of the character to drive the projected *CoM* towards the center of the *BoS*, while simultaneously repositioning the feet as necessary to adjust the configuration of the *BoS* according to the instantaneous motion of the *CoM*. The control scheme is derived from an experimental study observing stepping strategy for balance recovery [Chatagnon et al. 2023]. The results of this study were used to provide generalized equations of the feet behavior during stepping and to optimize the model parameters to enhance the overall response with regard to the experimental data.

The current work proposes two main contributions. The first contribution introduces a balance assessment method controlling the stepping policy of the character with regard to the time remaining before the *CoM* reaches the boundaries of the *BoS* (Time to Base of Support *TtBoS*). The second contribution is a novel foot stepping strategy leading to a steady state after a balance loss. These two contributions are combined with state-of-the-art physics-based human simulation, then tuned and validated using the same experimental data that inspired the control of the character. The end result is a simulator that can accurately reproduce the behavior of a human being pushed, for any push force or duration.

2 STATE OF THE ART

The field of physics-based human simulation is a long standing subject involved with the domains of biomechanics, robotics and dynamic simulation. This section covers the main approaches that emerged over the years, with an additional focus on the literature concerning balance recovery and the use of data in the field.

Optimisation based models. Optimisation methods are based on defining a set of costs to be minimized, which converges towards the simulated human performing a given task. Balance in these methods is part of the optimisation cost. These methods compute the best future trajectory, and as such are qualified as feedforward approaches. An example is the use of the zero moment point (*ZMP*) to follow [Kajita et al. 2003] along a trajectory. Predictive optimisation models [Shen et al. 2020] have addressed the simulation of standing balance, leading to works studying stepping after large pushes [Kudoh et al. 2006]. Optimisation based models have been often used in robotics, providing robust results by translating the problem into solvable equations [De Lasa et al. 2010], sometimes mixed with motion capture [Muico et al. 2009]. The quality of the results comes at the cost of long computation times and a lack of flexibility or reactivity in the character's responses, since the optimisation problem must be defined in advance.

Control based models. Control based methods aim to adjust the posture of the body to achieve a defined goal, often by controlling the position of the *CoM*. Unlike optimisation methods, they rely on the analysis of current and past states to achieve their goal. Controller-based methods rely on a set of controllers, that act typically on joints to move the character. While Proportional-Derivative (*PD*) joint controllers are mainly used, some works have use muscle-based control approaches [Cruz Ruiz et al. 2017; Geijtenbeek et al. 2013]. This type of control-based approach can track target motions, whether they are authored [Yin et al. 2007] or computed [Coros et al. 2010]. The strength of controller-based methods is their ability to respond and adapt to external perturbations.

To control the *CoM*, various techniques have been studied in the field. While previous works have used real-time inverse dynamics [Herzog et al. 2014], many rely on the inverse pendulum model [Tsai et al. 2009] (*IPM*), a simplified model of lower body equilibrium that can easily simulate the trajectory of the center of gravity. Previous studies have extended the *IPM* with intelligent foot placement using capture points [Pratt et al. 2006], or with inverse kinematics and state machines [Stephens and Atkeson 2010]. However, the simplicity of such models struggles against sudden perturbations, such as a hard push.

Since controlling the position of the *CoM* is closely related to balance, control methods must implement strategies for balance management for the simulation to work. In the literature, a three step balance recovery method has emerged [Aftab et al. 2012]. The strategy is rooted in human behavior, with low-intensity impulse recovery that does not require stepping and involves only the ankles and hips motion when possible. It relies directly on previous step triggering works [Stephens 2007] on the basis of the *CoM* movement. The core of the balance recovery simulation is balance assessment, the relationship between which and feet placement has been studied in several papers, with concepts such as the *Extrapolated CoM* (*XCoM*) [Schulz et al. 2006] or the *Base of Support* (*BoS*) [Hof et al. 2005]. The *BoS* has already been used to compute a threshold for step triggering [Pai et al. 2000], and the *XCoM* has been used in a simple control balance model [Hof 2008].

Learning based models. In recent years, traditional controllers have given way to deep learning, reinforcement learning, and other neural network based methods of character control. Balance in learning model is part of the reward system, without being the main focus. These works are able to reproduce tasks by imitating motion capture data [Peng et al. 2018], and are sometimes able to adapt the choice of task to the environment [Peng et al. 2021]. Some work focus on continuous adaptation of the learned movement [Xie

et al. 2020]. Another approach relies on reinforcement learning to generate control without the need for motion capture, but often falls short in terms of adaptability [Howell et al. 2022]. In general, learning-based approaches are the best at seamlessly reproducing captured motions, which may include complex interactions [Lee et al. 2018; Starke et al. 2020].

Data parameter optimisation. In the aforementioned models, motion capture is often used directly as a standard to mimic. This paper takes a different approach, with a Proportional Derivative (*PD*) joint control strategy whose parameters are optimized using experimental data. *PD* control optimisation is based on gain tuning and has been widely studied in the field of robotics, both with [Azmi et al. 2019; Gaing 2004; Zamani et al. 2009] and without [Jagodnik and van den Bogert 2010] using the particle swarm optimisation (PSO) method, which is used in this paper.

Optimisation of bipedal walking simulations has also been studied in the domain, with similarly structured work [Wang et al. 2009] based on the Simbicon paper. Metrics to compute the effort of human walking have been established [Wang et al. 2012], and along other metrics have been used in an optimisation scheme [Geijtenbeek et al. 2013] for gait simulation. Such metrics allow the evaluation of the produced motion and torques for this paper's output simulation.

Our work. This paper studies the stepping strategy required for balance recovery after pushes, with a focus on accurately representing reality. For these reasons, we opt for a control based approach. The lack of responsiveness of optimisation based methods clashes with our intention of responding to perturbations, while learning based approaches involve black boxes that naturally do not allow an interpretation of the model to validate its internal accuracy. Our approach is based on the 'Generalized Biped Walking Control' paper [Coros et al. 2010], which based on the Simbicon controller [Yin et al. 2007]. We extend the system with inverse kinematics similar to previous works [Stephens and Atkeson 2010]. The inverse kinematics are based on the goal feet position, a strategy that has been used in works related to trajectory step planning (the aforementioned *ZMP* [Kajita et al. 2003] and capture points [Pratt et al. 2006]).

Previous models lack an adequate response to high intensity push, which is the core of our contribution. We introduce a novel stepping strategy to handle strong perturbations, based on feet positioning. In contrast to other works where target motions are generated from predefined state machines or motion capture, our method is able to assess balance and compute proper feet placement in a human-like manner. Our general approach makes it possible to experiment with variations in the push intensity, duration, position, and in the morphology of the subject without changing the model. Our novel feet based balance assessment and recovery are both anchored in observations of recent experimental data dedicated to balance recovery. Previous works have established a link between step characteristics and COM velocity, through experimental analysis of pushing scenarios [Li et al. 2020; Zhang and Fu 2018]. Here we used a similar approach based on our experimental data [Chatagnon et al. 2023] .

This paper presents how the same data are used to tune the parameters of our model and validate its output motions. By using the data for optimisation, we are able to not only reproduce the human balancing process, but also to better explain its mechanism.

3 OVERVIEW

Figure 2 depicts the method used to simulate balance recovery, with a visual representation of the character being pushed and a flowchart of the simulation iteration. We follow the SI metric system for forces, distances and velocities. The legend of the figure shows the correspondence between the colors of the flowchart boxes and the main components of the proposed method: the **Locomotion System** that triggers steps to recover balance, the control of the **Foot Trajectory** after steps are triggered, and finally the **Full-Body Control** of the *Center of Mass* (*CoM*) position. The following paragraphs provide details about each of these components, starting with the controlled character itself.

Character. The character of the simulation is a poly-articulated kinematic chain of cuboid limbs, except for a capsule head. A total of 14 limbs are attached at 11 joints. Of all the joints, 4 have 1 Degree of Freedom (DoF) while the others have 3, for to a total of 25 DoFs. The size and mass of the limbs in the body follow Winter's [Winter 2009] table, scaled by the character's total height and weight.

Joint PD controller. Actuation of the motion for the character is mainly done with traditional Proportional Derivative (PD) controllers. For every joint, a unique set of *Kp* and *Kd* gains is provided handling all the DoFs of the joint. As seen in Figure 2 close-up 1, the desired and actual angles θ_d and θ are compared on a local, limb-aligned basis. Considering also the angle velocity $\dot{\theta}_d$ and $\dot{\theta}$, the PD controller outputs a torque τ that actuates the joint to the target angle, according to the classic PD controller equation (1).

$$\tau = K_p.(\theta_d - \theta) + K_d.(\dot{\theta}_d - \dot{\theta}) \tag{1}$$

Simulation iteration. At the beginning of the simulation iteration, the current state of the character is evaluated by the **Locomotion System** (Section 4.1) in red. This process is based on the analysis of the *Center of Mass* (*CoM*), the *Expected Center of Mass* (*XCoM*) and the *Base of Support* (*BoS*). If both feet are planted (i.e., the character is not walking), balance is assessed. If necessary, a step is triggered which puts the character in the walking state.

Depending on this analysis, a step may be necessary, which is represented by the **Foot Trajectory** component (Section 4.2), shown in blue in Figure 2. This involves first computing a goal position coordinates for the swinging foot on the ground XZ plane, here called $X_{desired}(X_D)$ and $Z_{desired}(Z_D)$, which will lead the character to balance. To reach this goal, a trajectory is computed to be followed during the next simulation iterations. An overall motion is generated for the legs using inverse kinematics, which follows the trajectory of the swinging foot. Each joint in the body has default target angles, some of which are overwritten by the inverse kinematics process during walking.

These goal angles are provided to the PD controllers, producing a torque τ at each joint of the body. All the forces applied in our method by the **Full-body Control** (Section 5) are shown in green. To support the motion of the body, the Jacobian transpose method is used to compute the torques for the three type of forces, F_{CoM}, F_{VEL} and F_{GRAV}. An example is shown in Figure 2 close-up 2, where the

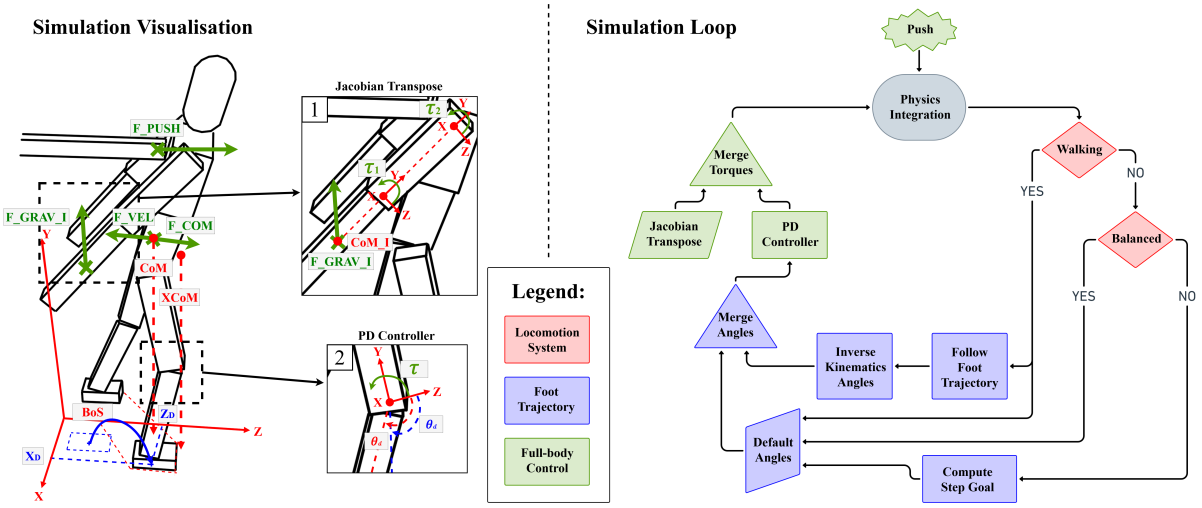

Figure 2: Character being pushed, with two close-up of internal systems (Left). Classification of the internal concepts and flowchart steps (Middle). Flowchart of the simulation iteration (Right).

force of the limb i, F_{GRAV_i}, is applied through two torques τ_1 and τ_2 using the elbow and shoulder joints, according to their Jacobian matrix with respect to the *CoM* of each limb.

All of these torques are then merged and integrated in the physics engine (using the Dantzig-Wolfe [Dantzig and Wolfe 1960] decomposition solver applied to Mixed Linear Complementary Problems), for a new iteration to begin. This process can be perturbed by an external force F_{PUSH}, directly applied to the body during the integration.

4 FEET MOTION STRATEGY

The core of this work is based on the simple premise that humans, after receiving a push, may need to take steps in order to recover balance. At the core of this recovery is the contact between the character and the ground: their feet. This section describes the two parts of the model that deal with the placement of the feet, respectively the **Locomotion System** and the **Foot Trajectory**.

4.1 Locomotion System

The first objective of the simulation is to accurately determine whether it is necessary or not for the character to take a step. This section introduces how the current state of the simulation is analyzed, leading to a walking state if necessary. It also explains the state machine that the model follows, to alternate stepping between legs to recover balance.

Balance assessment using Time to Base of Support. The purpose of this work is for the character to recover and maintain balance, with or without locomotion. As such, the first concept to be clarified is the definition of balance : what factual observation can be made to determine whether a character is balanced or not. For this purpose, this work is based on an experimental dataset of participants undergoing external perturbations [Chatagnon et al. 2023].

More specifically, the step triggering condition described in the following work [Chatagnon et al. 2023], is implemented in our model. This method to estimate step initiation could ether relies on on of the following concept: the *Margin of Stability* (MoS)[Hof et al. 2005], or the *Time to BoS boundary*[Schulz et al. 2006] (here referred as TtBoS). On one hand, the concept of MoS relies on the analysis of an augmented representation of the CoM relying its velocity called *XCoM*, computed with the following equations :

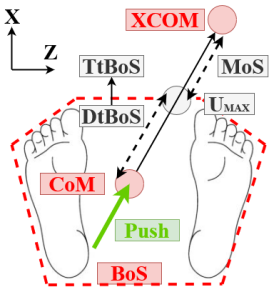

Figure 3: Step detection process when pushed

$$XCoM = \mathbf{x_{CoM}} + \frac{\dot{\mathbf{x}}_{\mathbf{CoM}}}{\omega_0} \qquad \text{with } \omega_0 = \sqrt{g/l} \tag{2}$$

$$MoS = (u_{max} - XCoM).\frac{\dot{\mathbf{x}}_{\mathbf{CoM}}}{||\dot{\mathbf{x}}_{\mathbf{CoM}}||} \tag{3}$$

Here, x_{CoM} represents the *CoM* position, g is the gravity constant, l is the length of the legs and u_{max} is the closest intersection with the *BoS* in the direction of the *XCoM* from the *CoM*. Figure 3 illustrates these concepts and the relationships between them.

On the other hand, the concept of TtBoS uses the *Distance to BoS (DtB)* together with the instantaneous velocity of the CoM, as written in the following equations :

$$TtBoS = \frac{DtB}{||\dot{\mathbf{x}}_{\mathbf{CoM}}||} \tag{4}$$

$$DtB = (u_{max} - \mathbf{x}_{\mathbf{CoM}}).\frac{\dot{\mathbf{x}}_{\mathbf{CoM}}}{||\dot{\mathbf{x}}_{\mathbf{CoM}}||} \quad (5)$$

The *DtB* represents the distance between the *CoM* and the nearest boundary of the *BoS* in the direction of *CoM* velocity, while the *TtBoS* is the time it would take for the *CoM* to reach the BoS at its current velocity. Based on experimental data, a threshold value T for the *Time to BoS* is defined, below which the character is considered to be unbalanced in the future, without the possibility to recover without taking a step.

Leg switch state machine. Using the previously defined threshold T, the simulation is now able to detect whether the character should step or not. This allows us to define a simple state machine which is the locomotion model of the character, as shown in Figure 4.

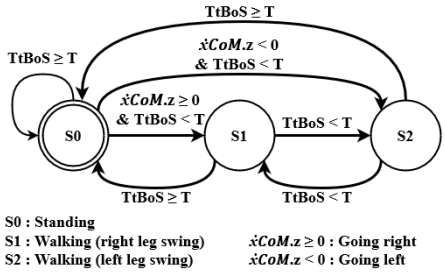

Figure 4: Locomotion state machine

The base state of the character is the *Standing* (*S0*) state. In this state, the character' rigidity is sufficient to maintain its *CoM* in the BoS, and steps are not required. When the threshold T is crossed, a step is initiated. This changes the state of the character to *Walking* with the right (*S1*) or left leg (*S2*) depending on the direction the *CoM* is moving toward.

While stepping, the characters converge toward a balanced state. However, if at the end of the step when both feet are in their intended position, the character is still considered as unbalanced (the *TtBoS* is still below T), then another step is initiated. The swinging leg is switched, and the process is repeated until the character is considered as balanced at the end of a step. Thus, it simply returns to the *Standing* state, where the internal forces are sufficient to keep the balance.

4.2 Foot trajectory

Computation of the foot trajectory. When the character enters a *Walking* state, first are computed the end goal position's coordinates, X_D and Z_D, for the foot of the current swinging leg. Not only must the final position be computed, but the *StepTime* the step takes to be completed is also critical for a quick yet cohesive movement.

To calculate these key factors, equations derived from experimental data are used. Both the foot end position and stepping time have been found to be closely related to the value of the *CoM* velocity when a step is triggered, the scalar denoted $\dot{\mathbf{x}}_{\mathbf{CoM}_{\mathbf{projected}}}$. Similar to the threshold for the *time to BoS*, those parts of the model are anchored in the observation of experimental data that supports the simulation, leading to the following equations and parameters:

$$(X_d, Z_d) = x_{Foot_{projected}} + a_1.height.\dot{\mathbf{x}}_{\mathbf{CoM}_{\mathbf{projected}}} \quad (6)$$

$$StepTime = a_2 + b_2.\dot{\mathbf{x}}_{\mathbf{CoM}_{\mathbf{projected}}} \quad (7)$$

In order to reach the target position (X_D, Z_D) in *StepTime* seconds, a trajectory must be established that will move the foot to the next position, as shown in Figure 5. The problem can be divided into two parts : the projected trajectory of the foot on the ground over time, and the height of the foot over time. The projected position can be easily computed by linear interpolation between the initial and final position, as seen on the left of Figure 5. For the height, the model approximates it using a spline proportional to the step distance, resulting in a 3D trajectory like the right part of Figure 5.

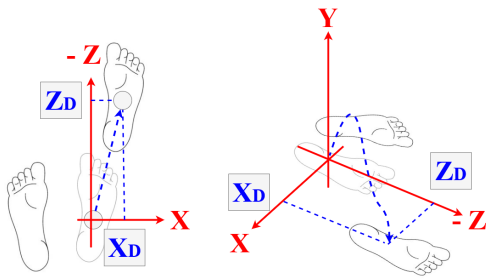

Figure 5: Step Computation in 2D and 3D

Taking into account the planned *StepTime*, this results in a trajectory that can be followed over time for the stepping foot. At each iteration of the simulation, the goal position of the stepping foot follows the trajectory according to the current elapsed time since the start of the step, leading to the goal (X_D, Z_D) position in *StepTime* seconds.

Swinging leg inverse kinematics. By default, each joint in the character's body has a target angle of 0 for each *DoF* while being subjected to gravity. When a step is taken, those goal angles are overwritten for the lower body to be in the correct position. These angles are computed at each simulation iteration when in the *Walking* state, following the previously defined trajectory. The angles are computed using the *Cycling Coordinate Descent (CCD)* algorithm [Song and Hu 2011], as shown in Figure 6. The first step is in the coronal plane, to compute the rotation around the X axis θ_{stride} which relies on the single *DoF* of the knee to simplify the inverse kinematics to a plane. The goal in blue follows along the trajectory, represented by the dashed line. This is shown in the second part, where by knowing the length of the leg's limbs and the position of the goal, the position of the knee can be deduced and the angles θ_{hip} and θ_{knee} are calculated.

Standing leg inverse kinematics. To support the stepping motion, the standing leg must contribute to the balance. The goal position of the *CoM* is estimated to be between the final positions of the two feet when they are projected onto the ground. The standing leg is given goal angles with the foot not moving and the *CoM* in the correct position, using the same *CCD* method but in reverse (where the end effector is now the hips).

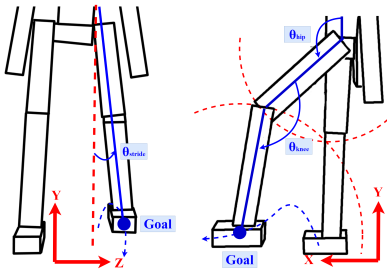

Figure 6: Two-step Inverse Kinematics

5 FULL-BODY CONTROL

Throughout the body, motion at this point is generated by PD controllers of the joints provided with goal angles. These goal angles are set to 0 by default. Thus, the angles of the whole body result only from the weight of gravity on the limbs, which is counteracted by the PD controllers. Below the waist, the goal angles are redefined when stepping. However, this leaves the rest of the body in its default state, which is both inefficient and incorrect. When pushed, every part of the body is involved in the balance recovery, albeit to different degrees. This section deals with the calculation of the other applied torques throughout the whole body.

Set of applied torques. In total, four different types of torques can be applied in the simulation. The first is from the *PD* controllers, which generate torques from goal angles. The other three are gravity compensation, velocity compensation and *CoM* driving, which are computed using jacobian transpose. Gravity compensation compensates for the constant effort required to resist gravity, to reduce the load on the *PD* controllers for finer control. The goal of velocity compensation is to involve the whole body in counteracting the push by converging towards zero *CoM* velocity. Finally, *CoM* driving assists the legs' PD controller in their stepping motion by pushing the ground-projected *CoM* toward the center of the feet.

Torque computation. The method used in order to compute the torques is the jacobian transpose, a classic control method in robotics [Buss 2004]. Figure 7 illustrates the application to limb i of a force F_{GRAV_i} at the position CoM_i through two torques, one for the lower arm (τ_1 in the figure) and one for the upper arm (τ_2 in the figure). Each torque is computed for a joint k, with the goal of acting on a limb i and as such with respect to its CoM_i position with:

$$J_k = \frac{\delta P_{CoM_i}}{\delta \theta_k} \tag{8}$$

By transposing the Jacobian of the joint k, a torque can be computed that will apply the equivalent of a force F (F_{GRAV_i} in the figure) at the position CoM_i using :

$$\tau_k = J_k^T F_{GRAV_i} \tag{9}$$

The implementation of this process and the relevant force values to add to the system (namely the following gravity and velocity compensation) are described in detail in the paper 'Generalized Biped Walking Control' [Coros et al. 2010], which is the main inspiration for this paper.

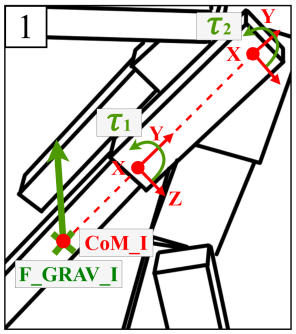

Figure 7: The Jacobian forces applied

Figure 8: Unbalanced participant about to step following external perturbation from the experimenter.

Gravity compensation. The first torques applied are gravity compensation. A set of forces is computed, one for each limb of the upper body. The position of force application is the *CoM* of the limb i, with the force following the equation:

$$F_{GRAV_i} = -g.m_i \tag{10}$$

The joints between the corresponding limb and the hips each have a torque computed. In the end, every limb i of the upper body have gravity compensation applied, with each limb i involving a different set of joints k using jacobian transpose to do so. For example, the shoulder joint not only compensates for the gravity of the upper arm limb, but also the lower arm limb, since this joint affects both.

Velocity compensation. The next force applied is velocity compensation, at the *CoM* of the character, with the corresponding equation :

$$F_{VEL} = -a_{VEL}.\dot{\mathbf{x}}_{\mathbf{CoM}} \tag{11}$$

a_{VEL} is a dimensionless tuning parameter to control the intensity of the resulting torques. This force is applied only once at the *CoM* of the character, but the torques are computed for each joint between the standing foot and the head.

CoM driving. In addition to the previous set of forces, another force called the *CoM* driving contributes to balance. It is applied in the same way as velocity compensation, from the standing foot to the head, and only once at the *CoM* of the character using:

$$F_{CoM} = a_{CoM}.(\mathbf{x}_{\mathbf{CoM}_{\mathbf{desired}}} - \mathbf{x}_{\mathbf{CoM}}) \tag{12}$$

a_{CoM} is another dimensionless tuning parameter, once again regulating the final torques values. It is a mixture of the two previous forces: it supports the PD controllers like the gravity compensation and involves the whole body like the velocity compensation.

Force purpose and equilibrium. The three forces applied to the system are expressed in the same way as the PD controllers : by torques at the joints of the body. This means that at each joint, several torques can be summed to get the final applied torque. In order for this summation to work, each torque value must be carefully tuned in order to ensure a balanced contribution from the controllers, through gains (K_p, K_d) or tuning parameters (a_{VEL},a_{CoM}).

6 PARAMETER OPTIMISATION

The goal of the simulation is to mimic the reaction of a human to physical perturbations, more specifically a push. As seen in the previous sections, this simulation relies heavily on observations made from the experiment data [Chatagnon et al. 2023]. The experiment can be summarized as follows : A person is pushed with a pole with a certain intensity for a certain time, and in response takes steps or not at a certain distance. The experimental data consists of markers positions, which are processed with CusToM [Muller et al. 2019] to compute reference joint angles and CoM trajectories using inverse kinematics.

As shown in Figure 8, the trigger for the experiment is the force applied to the body. Pushes of varying angle, force and time were explored.

Several parameters introduced into the simulation can be adjusted to improve the realism of the results. Therefore we have developed an optimisation procedure to adjust these parameters in order to make the generated simulations comparable to the experimental data.

Particle Swarm Optimisation. The optimisation process uses the *PSO* (*Particle Swarm Optimisation*) algorithm [Kennedy and Eberhart 1995]. This process involves exploring a set of parameters in a bounded multi-dimensional domain. It uses multiple particles that influence each other, by defining a personal and global best and gravitating towards an optimal multi-dimensional position.

More specifically, PySwarm [pys [n. d.]] is used along the simulation. After many iterations, the particles naturally gravitate towards a global minimum result according to the defined cost function.

Joint optimisation. The first optimisation step adjusts the joint *PD* controller gains. The goal is to find the best gain values that allow an accurate tracking of the goal angles without introducing unnecessary or unrealistic forces. Total cost is computed by summing the results over a set of scenarios, covering weak to small pushes, with the C_{joint} cost function of a single push with a total duration of t_{max} is :

$$\begin{aligned} C_{joint} = W_{angle}\left\{\int_0^{t_{max}} E_{angle}(t)\delta t\right\}_{H_{angle}} \\ +W_{effort}\left\{\int_0^{t_{max}} E_{effort}(t)\delta t\right\}_{H_{effort}} \end{aligned} \tag{13}$$

Where for each error, a dimensionless weight W and a threshold value H are defined. Below the threshold, the integrated value is replaced by 0. This means that if the integrated value of the angle error is low enough it is ignored. With a heavy weight, this means that simulation that do not follow the goal angles are heavily penalized, while all simulation below the threshold compete to minimize the effort.

The cost function is based on two main error functions, inspired by works on human effort [Geijtenbeek et al. 2013], integrated over the duration of the simulation. For effort minimization $E_{effort}(t)$, we use the current rate of metabolic expenditure [Wang et al. 2012] where i is for each joint, and j is for each DoF of each joint .

$$E_{angle}(t) = \sum_{i \in Joints} \sum_{j \in DoF} ||\theta_{i,j_{desired}}(t) - \theta_{i,j}(t)||^2 \tag{14}$$

$$E_{effort}(t) = \sum_{i \in Joints} \sum_{j \in DoF} ||\tau_{i,j}(t)||^2 \tag{15}$$

Model optimisation. The first optimisation concerned the ability of the body to follow joint angle goals without too much effort. This second optimisation aims at tuning the generation of these angle goals, along with tuning the full-body control forces. The goal is for the trajectory of the *CoM* generated by the simulation to resemble the general human response to the same push. To do this, the corresponding C_{model} cost function is :

$$C_{model} = \int_0^{t_{max}} E_{traj}(t)dt \tag{16}$$

It relies on a trajectory error function, that directly compares the experimental data and the simulation side by side. The experimental data, while very useful, also show a great deal of variability in individual response and between characters. For the same push and character, the final position shows high inconsistencies. Comparing the velocity over time, rather than the absolute positions, eliminates the differences at the starting and final positions between the simulation and the data, focusing on the stepping motion.

$$E_{traj}(t) = ||v_{CoM_{desired}}(t) - v_{CoM}(t)||^2 \tag{17}$$

For this optimisation, the data compared to the simulation is limited to perpendicular pushes to the back. For each iteration, the same set of pushes is analyzed for consistency. This set of pushes pans across the entire range of push intensity and duration, from small pushes with no step to multi-step responses.

A total of 6 parameters are optimized in this second step. The first two parameters are those from the Jacobian-based forces a_{VEL} and a_{CoM}. Closely related to the force is a a_{PUSH} tuning parameter, a conversion gain of the push for the simulation.

The remaining 3 parameters are related to the feet. The *stickiness* S of the feet is the first : the simulation alternates each foot between total complete freedom and a grounded state with restricted motions to avoid the physics engine's foot sliding. The grounded state is activated when the foot does not belong to the swinging leg and touching the ground. The stickiness when grounded ranges from 0 to 1, where 0 means no possible movement and 1 complete freedom of movement. It is followed by the *Height of the Spline* H_s, a dimensionless tuning parameter of step height over time. Finally, a filter is applied to avoid micro-step, tuned by the *threshold* T_f in our optimisation. Steps with a *stepTime* under T_f are removed.

7 RESULTS AND DISCUSSION

In order to validate the direction of our approach, four aspects are scrutinized. The first evaluation aims to check whether the simulation is capable of recovering balance in the boundaries of our data, with a supplementary look beyond the experimental pushes values. The second metric is the torques produced at the joints of the model. The goal is for the model's produced torque to be under the human maximum in the same condition. By comparing the

values with real human standards, the physical soundness of the control model can be tested.

The last two analyses are based on reproducing each experimental push in the simulation, allowing direct data comparison. This is done by normalizing the real trajectories in accordance to height and weight, creating pairs of real and simulated results for a push force and duration. The first comparison performed is the step triggering, to verify that the simulation triggers step consistently with the data. To do this, we use a precision test, where 'Positive' and 'Negatives' correspond to the simulation requiring a step, while 'True' and 'False' come from the experimental data needing to step. The second criterion is the CoM trajectories over time, being compared visually and statistically between experimental and simulated data. The difference between every pair of simulated and real trajectories were analyzed using an SPM [Pataky 2011] t-test, to determine if there are notable differences between them.

7.1 Validation

The first qualitative metric explored in Figure 9 is the ability of the character to recover balance after receiving pushes of varying strength and duration. By default, we set our character to an average height and weight (1.70m and 70kg), pushed from a perpendicular angle to the back. A push is applied at every 20N (between 0 and 300 Newtons) and 0.2s (between 0 and 2 seconds), only once due to the deterministic nature of the simulation. Note that this range of impulse goes beyond the experimental study on which we based our work, as shown in Figure 9. The blue and red regions correspond to complete success or failure. Mixed responses indicate the area in which the character fell for pushes of lower intensity than pushes with successful balance recovery.

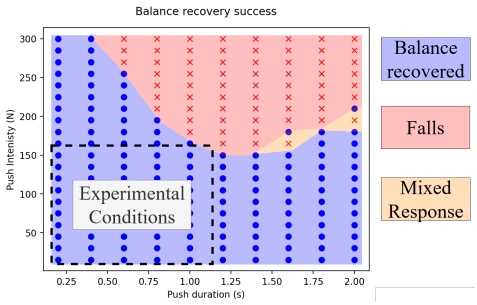

Figure 9: Balance recovery for varied impulse, with the range of intensity and duration covered experimentally delimited

The model is able to recover balance after a physical perturbation as seen in Figure 9, within the boundaries of the experimental conditions but also beyond. Falling naturally occurs for strong pushes, where balance is naturally not recoverable. While the lack of data does not allow to appreciate the experimental balance recovery limits, the response of the model in experimental conditions is in accordance to the data, thus validating the model for this range of pushes. An interesting phenomena also appears, as strong but very long pushes yield more balance recovery : the push is so long that it begins to serve as support for the character, that simply steps along the push.

7.2 Physical Plausibility

The following section aims at verifying whether the generated torques values are within the range of human torques. The measurements are made for the lower body joints, and compared with an empirical study [Anderson et al. 2007]. Figure 10 illustrates the results. For each type of torque, the maximum torque (up axis) observed in humans for a given angle (right axis) and an angular velocity value (bottom axis) is represented as a surface. Each color corresponds to the torque measured over the duration of a low (green), medium (blue) and high (red) impulse push. The surface represents the torque generation capabilities for the corresponding angle and angular velocity values reported from Anderson *et al* [Anderson et al. 2007]. For every lower body joint, the measured torques are below the human maximum. The only exception is plantar flexion, with values well above the recorded human limits.

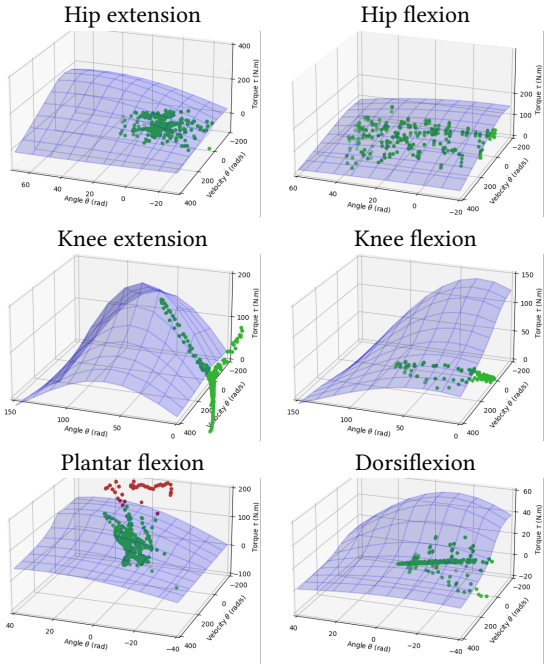

Figure 10: Measured torques of the simulation in respect (green for valid, red for over the maximum) to recorded human maximum torques (surface)

The computation of torques in the simulator is done without prior knowledge, but still manages to produce cohesive values, underlying the physical cohesiveness of the model. The only exception is the plantar flexion, which suffers from the foot contact model of the simulator that produces unrealistic contact forces and high torques. This is not a problem in our case because the interactions between the feet and the ground are simplified, and the generated torque is mostly bypassed by our contact model. However, this means that the dispersion of energy into the ground is inaccurate, which could be a clue for further progress of this work, which will be discussed in the conclusion section.

7.3 Experimental vs simulated data

In this subsection, the simulation results are confronted and compared with the experimental data computed from measured human pushes.

Step triggering. For each pair of real and simulated pushes, the correspondence of step triggering is tested. The resulting values are a sensitivity of 0.88, a precision of 0.81, and an overall accuracy of 0.74.

The step triggering condition based on the use of the *XCoM* and *BoS* is able to detect the same balance as the experimental data with an accuracy of 0.74. The relatively low accuracy is due to the high variation in the original data. However the high sensitivity of 0.88 shows that the necessity of stepping is well detected, rather taking a step than risking a fall.

CoM trajectory. Figures 11 and 12 showcase a study of the *CoM* trajectory over time in the direction of the push. Only stepping motions are examined in these figures. The color gradient corresponds to the intensity of the push for each of the simulated (red) and real (blue) motions. Once the step is triggered, Figure 11 shows the similarity in the position over the duration of the step, which is directly related to both the simulated *StepTime* and feet goal position.

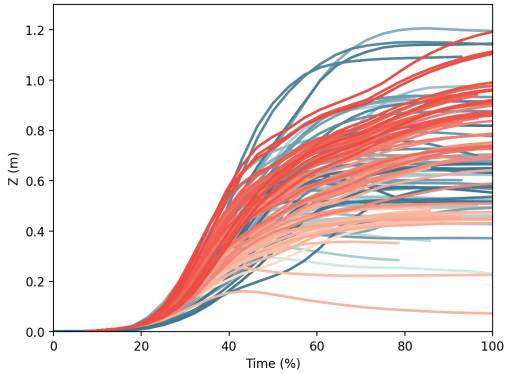

Figure 11: Trajectory over time of the normalized experimental data (blue) and simulation (red)

Figure 11 serves as a first validation of the motions generated by our simulation. Both time-wise and space-wise, the trajectories of the *CoM* are similar. It also shows the lack of variability in our model's responses, since two final positions seems to be favored, corresponding to one and two steps. The distance and number of steps correlate accurately with the experimental data. The mean values and variance in Figure 12 have been studied to detect discrepancies. From a statistical standpoint, the hypothesis of the SPM t-test is never rejected, supporting that our model is able to reproduce accurately the real motions of human balance recovery.

As a complementary analysis, various cases of different push angles, subject height and subject weight with corresponding morphology were tested in our simulation. A supplementary video is provided to demonstrate those configurations, and while no data is used to validate the results, this serves as an example of the potential uses of this simulation.

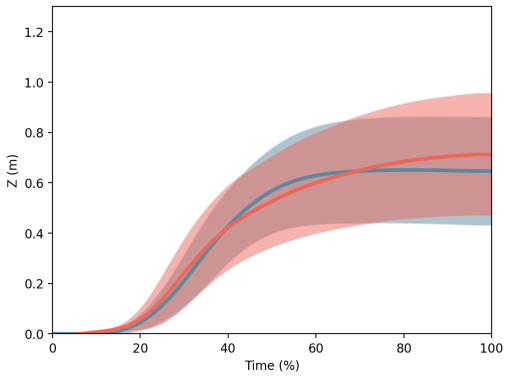

Figure 12: Mean trajectory and variance over time of the normalized experimental data (blue) and simulation (red)

7.4 Baseline comparisons

We provide a qualitative comparison with two baseline methods: (a) RL: a reinforcement learning based neural network policy, trained using proximal policy optimization (PPO) [Reda et al. 2023]; and (b) MPC: an online model predictive control method [Howell et al. 2022]. The RL policy is conditioned on the current state as well as a target state that consists of the position and velocity of the head, root, hands, and feet. The policy is trained to imitate walking and standing clips, consisting of about 10 minutes of data. At inference time, the policy is tracking a reference clip that consists of a fixed standing pose. The character is simulated using the GPU-based ISAAC simulator. The MPC policy uses a receeding horizon optimization of 0.35 s, as simulated using MuJoCo. We experiment with sampling-based method and iLQG methods, which both optimize the torque-based action trajectory over the horizon. A multithreaded implementation allows for interactive experimentation with the MPC method, opeerating at 10 to 50% of real-time. While the default MuJoCo-MPC implementation [Howell et al. 2022] comes with both a standing and walking controller, we find that the walking controller, which also stands when at rest, is most robust to pushes and thus we use this as for our baseline comparison.

The RL baseline produces results that are more qualitatively similar to the real push results than the MPC results. However the results from our proposed controller are qualitatively more similar to the real world results, taking into account the number, length, and timing of the resulting steps. We note that one possibly way to improve on the RL results would be to use an adversarial motion prior (AMP), which can be shown to be a feasible way to guide RL to solve a motion task while respecting the available motion styles. In principle, a set of the captured push responses clips could then be used as the data to train an AMP policy. However, this type of approach remains tricky to train, given that the discriminator capability and the RL state distributions are both changing over time.

8 CONCLUSION

Contributions. This paper introduces a new control-based model which, by taking direct inspiration from real push data, aims to accurately recreate the human response to a physical perturbation. With relevant feet positioning and forces in the body, real life stepping behaviors and trajectories can be observed in the simulator.

As new contributions to the domain this paper not only introduces a new balance assessment method but also a new adaptive foot trajectory computation strategy, and validated their robustness against novel experimental data in the context of physics-based human simulation.

Limitations. Our method shows its limitations mainly through the use of the default collision system for the contact foot model. The behavior of the character upon hitting the ground is far from reality, as shown by the torque analysis of the physical plausibility study. Contact with the ground simply nullifies the foot's acceleration and velocity, and then locks the foot in position to serve as support. This is something we could aim to improve, as this is an important interaction where the energy from the push is dissipated into the ground.

Another limitation concerning the foot trajectory is the leg-crossing issue. When stepping to the side, the other leg becomes an obstacle to the step and must be taken into account in the foot trajectory. While the issue does not appear in perpendicular pushes, pushes from the side often results in our simulation with the legs colliding and a fall. While the current solution is to disable solid collisions, this does not represent reality and should be replaced by a better foot trajectory computation, aware of the space available.

Finally, while the produced trajectories are comparable to the experimental values, it lacks the inherent variability of human behavior. Figure 11 is a prime example of how the simulation's results is representative of reality, but too rigid, as the trajectories corresponding to one or two steps create two sets of end positions.

Perspectives. Adding variability using stochastic approaches to step goals and time could lead to better trajectories capable of recreating the natural variability of the experimental data, using similar distributions to those observed in the data. Variations in push angle and morphology of the character could be studied in more detail. The video linked to this paper showcases the simulation results with different push angles, height and weight. Entailing a new foot trajectory computation system and perhaps more adaptive torque computation, this could be a way of generating high variability data that could not otherwise be captured.

The variability could also lie in the surrounding environment of the simulation. A spatially aware character might be able to generate feet target position adapted to the surrounding, thus avoiding obstacles or supporting itself with other objects. This opens the way for complex balance resolution, where all parts of the body especially the hands can be used to recover balance.

With characters of different morphologies and a model that is spatially aware of the surroundings, a new type of behavior could be observed in physics-based human simulation. Multi-character physics-based interactions could be simulated, taking the push simulation to the level of a crowd simulation. Any crowd configuration could then be simulated by adding complex environments and objectives for each person in the simulation.

The simulation proposed in this paper is based on a character modeled from realistic anthropometric data and produces biomechanically plausible results, which makes it possible to consider its future use for fall simulations, or the instantiation of large scale physics-based crowd simulations.

ACKNOWLEDGMENTS

This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No [899739] (CrowdDNA), as well as French government ANR CONTINUUM project under grant agreement No [ANR-21-ESRE-0030]

REFERENCES

[n. d.]. PySwarms documentation page. https://pyswarms.readthedocs.io/en/latest/.

Zohaib Aftab, Thomas Robert, and Pierre-Brice Wieber. 2012. Ankle, hip and stepping strategies for humanoid balance recovery with a single Model Predictive Control scheme. In *2012 12th IEEE-RAS International Conference on Humanoid Robots (Humanoids 2012)*. IEEE, 159–164.

Dennis E Anderson, Michael L Madigan, and Maury A Nussbaum. 2007. Maximum voluntary joint torque as a function of joint angle and angular velocity: model development and application to the lower limb. *Journal of biomechanics* 40, 14 (2007), 3105–3113.

Nur Iffah Mohamed Azmi, Nafrizuan Mat Yahya, Ho Jun Fu, and Wan Azhar Wan Yusoff. 2019. Optimization of the PID-PD parameters of the overhead crane control system by using PSO algorithm. In *MATEC Web of Conferences*, Vol. 255. EDP Sciences, 04001.

Samuel R Buss. 2004. Introduction to inverse kinematics with jacobian transpose, pseudoinverse and damped least squares methods. *IEEE Journal of Robotics and Automation* 17, 1-19 (2004), 16.

Thomas Chatagnon, Anne-Hélène Olivier, Ludovic Hoyet, Julien Pettré, and Charles Pontonnier. 2023. Stepping strategies of young adults undergoing sudden external perturbation from different directions. *Journal of Biomechanics* (2023), 111703.

Kuangyou B. Cheng and Chih-Kuo Yeh. 2015. A unified approach for revealing multiple balance recovery strategies. *Human Movement Science* 44 (2015), 307–316. https://doi.org/10.1016/j.humov.2015.10.001

Stelian Coros, Philippe Beaudoin, and Michiel Van de Panne. 2010. Generalized biped walking control. *ACM Transactions On Graphics (TOG)* 29, 4 (2010), 1–9.

Ana Lucia Cruz Ruiz, Charles Pontonnier, Nicolas Pronost, and Georges Dumont. 2017. Muscle-based control for character animation. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 122–147.

George B Dantzig and Philip Wolfe. 1960. Decomposition principle for linear programs. *Operations research* 8, 1 (1960), 101–111.

Martin De Lasa, Igor Mordatch, and Aaron Hertzmann. 2010. Feature-based locomotion controllers. *ACM Transactions on Graphics (TOG)* 29, 4 (2010), 1–10.

Zwe-Lee Gaing. 2004. A particle swarm optimization approach for optimum design of PID controller in AVR system. *IEEE transactions on energy conversion* 19, 2 (2004), 384–391.

Thomas Geijtenbeek, Michiel Van De Panne, and A Frank Van Der Stappen. 2013. Flexible muscle-based locomotion for bipedal creatures. *ACM Transactions on Graphics (TOG)* 32, 6 (2013), 1–11.

Alexander Herzog, Ludovic Righetti, Felix Grimminger, Peter Pastor, and Stefan Schaal. 2014. Balancing experiments on a torque-controlled humanoid with hierarchical inverse dynamics. In *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 981–988.

AL Hof, MGJ Gazendam, and WE Sinke. 2005. The condition for dynamic stability. *Journal of biomechanics* 38, 1 (2005), 1–8.

At L Hof. 2008. The 'extrapolated center of mass' concept suggests a simple control of balance in walking. *Human movement science* 27, 1 (2008), 112–125.

Taylor Howell, Nimrod Gileadi, Saran Tunyasuvunakool, Kevin Zakka, Tom Erez, and Yuval Tassa. 2022. Predictive sampling: Real-time behaviour synthesis with mujoco. *arXiv preprint arXiv:2212.00541* (2022).

Kathleen M Jagodnik and Antonie J van den Bogert. 2010. Optimization and evaluation of a proportional derivative controller for planar arm movement. *Journal of Biomechanics* 43, 6 (2010), 1086–1091.

Shuuji Kajita, Fumio Kanehiro, Kenji Kaneko, Kiyoshi Fujiwara, Kensuke Harada, Kazuhito Yokoi, and Hirohisa Hirukawa. 2003. Biped walking pattern generation by

using preview control of zero-moment point. In *2003 IEEE international conference on robotics and automation (Cat. No. 03CH37422)*, Vol. 2. IEEE, 1620–1626.

James Kennedy and Russell Eberhart. 1995. Particle swarm optimization. In *Proceedings of ICNN'95-international conference on neural networks*, Vol. 4. IEEE, 1942–1948.

Shunsuke Kudoh, Taku Komura, and Katsushi Ikeuchi. 2006. Stepping motion for a human-like character to maintain balance against large perturbations. In *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.* IEEE, 2661–2666.

Kyungho Lee, Seyoung Lee, and Jehee Lee. 2018. Interactive character animation by learning multi-objective control. *ACM Transactions on Graphics (TOG)* 37, 6 (2018), 1–10.

Xudong Li, Weiguo Song, Xuan Xu, Jun Zhang, Long Xia, and Congling Shi. 2020. Experimental study on pedestrian contact force under different degrees of crowding. *Safety Science* 127 (2020), 104713. https://doi.org/10.1016/j.ssci.2020.104713

Uldarico Muico, Yongjoon Lee, Jovan Popović, and Zoran Popović. 2009. Contact-aware nonlinear control of dynamic characters. In *ACM SIGGRAPH 2009 papers*. 1–9.

Antoine Muller, Charles Pontonnier, Pierre Puchaud, and Georges Dumont. 2019. CusToM: a Matlab toolbox for musculoskeletal simulation. *Journal of Open Source Software* 4, 33 (2019), 1–3.

Y-C Pai, BE Maki, K Iqbal, WE McIlroy, and SD Perry. 2000. Thresholds for step initiation induced by support-surface translation: a dynamic center-of-mass model provides much better prediction than a static model. *Journal of biomechanics* 33, 3 (2000), 387–392.

Todd Pataky. 2011. One-dimensional statistical parametric mapping in Python. *Computer methods in biomechanics and biomedical engineering* 15 (07 2011), 295–301. https://doi.org/10.1080/10255842.2010.527837

Xue Bin Peng, Pieter Abbeel, Sergey Levine, and Michiel Van de Panne. 2018. Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions On Graphics (TOG)* 37, 4 (2018), 1–14.

Xue Bin Peng, Ze Ma, Pieter Abbeel, Sergey Levine, and Angjoo Kanazawa. 2021. Amp: Adversarial motion priors for stylized physics-based character control. *ACM Transactions on Graphics (ToG)* 40, 4 (2021), 1–20.

Jerry Pratt, John Carff, Sergey Drakunov, and Ambarish Goswami. 2006. Capture point: A step toward humanoid push recovery. In *2006 6th IEEE-RAS international conference on humanoid robots*. IEEE, 200–207.

Daniele Reda, Jungdam Won, Yuting Ye, Michiel van de Panne, and Alexander Winkler. 2023. Physics-based Motion Retargeting from Sparse Inputs. *arXiv preprint arXiv:2307.01938* (2023).

Brian W Schulz, James A Ashton-Miller, and Neil B Alexander. 2006. Can initial and additional compensatory steps be predicted in young, older, and balance-impaired older females in response to anterior and posterior waist pulls while standing? *Journal of biomechanics* 39, 8 (2006), 1444–1453.

Keli Shen, Ahmed Chemori, and Mitsuhiro Hayashibe. 2020. Human-like balance recovery based on numerical model predictive control strategy. *IEEE Access* 8 (2020), 92050–92060.

Wei Song and Guang Hu. 2011. A fast inverse kinematics algorithm for joint animation. *Procedia Engineering* 24 (2011), 350–354.

Sebastian Starke, Yiwei Zhao, Taku Komura, and Kazi Zaman. 2020. Local Motion Phases for Learning Multi-Contact Character Movements. *ACM Trans. Graph.* 39, 4, Article 54 (aug 2020), 14 pages. https://doi.org/10.1145/3386569.3392450

Benjamin Stephens. 2007. Humanoid push recovery. In *2007 7th IEEE-RAS International Conference on Humanoid Robots*. IEEE, 589–595.

Benjamin J Stephens and Christopher G Atkeson. 2010. Push recovery by stepping for humanoid robots with force controlled joints. In *2010 10th IEEE-RAS International conference on humanoid robots*. IEEE, 52–59.

Yao-Yang Tsai, Wen-Chieh Lin, Kuangyou B Cheng, Jehee Lee, and Tong-Yee Lee. 2009. 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.

Jack M Wang, David J Fleet, and Aaron Hertzmann. 2009. Optimizing walking controllers. In *ACM SIGGRAPH Asia 2009 papers*. 1–8.

Jack M Wang, Samuel R Hamner, Scott L Delp, and Vladlen Koltun. 2012. Optimizing locomotion controllers using biologically-based actuators and objectives. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–11.

David A Winter. 2009. *Biomechanics and motor control of human movement*. John Wiley & Sons.

Zhaoming Xie, Hung Yu Ling, Nam Hee Kim, and Michiel van de Panne. 2020. Allsteps: curriculum-driven learning of stepping stone skills. In *Computer Graphics Forum*, Vol. 39. Wiley Online Library, 213–224.

KangKang Yin, Kevin Loken, and Michiel Van de Panne. 2007. Simbicon: Simple biped locomotion control. *ACM Transactions on Graphics (TOG)* 26, 3 (2007), 105–es.

Majid Zamani, Masoud Karimi-Ghartemani, Nasser Sadati, and Mostafa Parniani. 2009. Design of a fractional order PID controller for an AVR using particle swarm optimization. *Control Engineering Practice* 17, 12 (2009), 1380–1387. https://doi.org/10.1016/j.conengprac.2009.07.005 Special Section: The 2007 IFAC Symposium on Advances in Automotive Control.

Lei Zhang and Chenglong Fu. 2018. Predicting foot placement for balance through a simple model with swing leg dynamics. *Journal of Biomechanics* 77 (2018), 155–162. https://doi.org/10.1016/j.jbiomech.2018.07.006