Volume 0 (1981), Number 0 pp. 1–11 COMPUTER GRAPHICS *forum*

Inverse Dynamics Filtering for Sampling-based Motion Control

Kaixiang Xie† and Paul G. Kry‡

McGill University, Canada

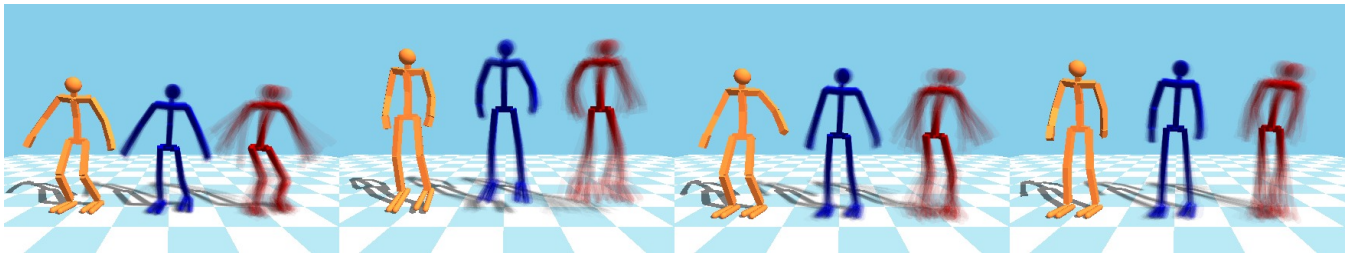

Figure 1: Samples drawn for different frames of a jumping motion. The orange is the reference motion. The blue shows the samples drawn by our method (with sample covariance initialization) and the red shows the samples drawn by our implementation of the sampling-based motion control method of Liu et al. [LYG15]. With the use of inverse dynamics torques and sample covariance initialization, our sample simulations can have higher success rates, and we can choose to draw control samples from tighter distributions to produce smoother results.

Abstract

We improve the sampling-based motion control method proposed by Liu et al. [LYG15] using inverse dynamics. To deal with noise in the motion capture we filter the motion data using a Butterworth filter where we choose the cutoff frequency such that the zero-moment point falls within the support polygon for the greatest number of frames. We discuss how to detect foot contact for foot and ground optimization and inverse dynamics, and we optimize to increase the area of supporting polygon. Sample simulations receive filtered inverse dynamics torques at frames where the ZMP is sufficiently close to the support polygon, which simplifies the problem of finding the PD targets that produce physically valid control matching the target motion. We test our method on different motions and we demonstrate that our method has lower error, higher success rates, and generally produces smoother results.

CCS Concepts

•***Computing methodologies*** → ***Animation;***

Keywords: *physics-based animation, motion control, character animation*

1. Introduction

Physics-based character motion plays an important role in computer animation and computer games. While kinematic methods are widely used in the industry to adapt motion capture to new characters or new scenarios, such methods can fail to produce physically valid motions. The great promise of physics-based simulation for character animation is that the resulting motion will be physically valid, but this comes at the cost of solving a control problem. That is, the motion must satisfy other important constraints, such as staying balanced, or reproducing an example trajectory recorded with motion capture.

In this paper, we focus on the problem of producing a physically valid control trajectory for a given motion capture clip. Furthermore, we follow the steps of Liu et al. [LYvdP*10,LYG15], who introduce and improve a sampling-based method for motion reconstruction which they call SAMCON. Their method works well on contact-rich motions and can be easily parallelized. They demonstrate that their method is suitable for tasks like motion transformation and motion retargeting. Note that this motion reconstruction problem is an important component necessary to solve other larger and important problems. For instance, it can be used in the creation of feedback controllers that can run in real-time (e.g., see work on control fragments [LH17]).

† Email: kaixiang.xie@mail.mgill.ca

‡ Email: kry@cs.mcgill.ca

Building on the improved SAMCON method [LYG15], we provide the sample simulations feed forward torques computed by inverse dynamics of the target motion clip. We also shape the initial covariance matrices to correspond to the variation of torques that we observe in the target motion clip. The intuition is that the sample simulations are more likely to be in the ballpark of a correct trajectory with the feed forward inverse dynamic torques. As such we can achieve a good success rate in simulations with control offsets that are drawn from tighter distribution, which has the added benefit of producing smoother result trajectories. Figure 1 shows that our method draws tighter distributed samples (blue) compared with SAMCON (red). Note that when we refer to SAMCON in this paper, we refer to the improved SAMCON algorithm [LYG15], and not the original version [LYvdP*10].

The challenge of using inverse dynamics to improve the improved SAMCON method is that noise in the motion capture leads to very noisy torques due to the finite differences that are necessary to estimate joint velocities and accelerations. We apply a Butterworth filter to the motion data in order to reduce the noise. We choose the cutoff frequency based on the signed distance between the zero-moment point and the support polygon. Accurate foot ground contact is also important for successful inverse dynamics. We propose a method for foot and ground optimization that produces better support polygons by ensuring the bottom of the foot is better aligned with the ground. While filtering and foot optimization do not lead to inverse dynamics control torques that will produce the desired target motion, they will simplify the process of finding PD control targets within the sampling-based optimization framework. However, when inverse dynamics torques place the ZMP far outside the support polygon, they may cause foot rotation and may work against the objective of keeping the character balanced and following the target trajectory. In these cases, we fall back to the case of the improved SAMCON sampling without feed forward torques.

We evaluate our method on a variety of different motions. We observe that our method can generate lower error trajectories for most motions. We also observe a higher success rate for reconstructing the control within each sliding window (i.e., the SAMCON algorithm works only on a small part of the control problem at any given time). While this has a minimal effect on reducing the computation time to produce a physically valid trajectory, it does lead to smoother solution trajectories than the improved SAMCON method. That is, a smaller sampling distribution can be used, set by the covariance matrix adaptation evolution strategy (CMA-ES) step size [Han06]. Control samples drawn from a tighter distribution naturally lead to a smoother simulated trajectory. Smoothing of the control trajectory in the improved SAMCON method, in contrast, comes later in the process from a repeated refinement of a solution.

Key features of our contribution include the following: an increase of the success rate and reduction of the error of the SAMCON method by introducing a base control force using inverse dynamics; an efficient method for setting foot geometry configuration and floor configuration to ensure better quality foot-ground contact; a straightforward and automatic method for selecting the cutoff frequency for the filters we use in computing inverse dynamics; and a method for setting the initial covariance matrix of the sample distribution.

2. Related Work

Control design can require the creation of control trajectories that imitate motions from a motion capture database. Sok et al. [SKL07] develop an optimization method which transforms biped motion into physically-feasible, balance-maintaining motion. Sharon and van de Panne [SvdP05] use a simple deterministic search algorithm to optimize the control for a target motion. Inspired by their work, Liu et al. [LYvdP*10] use a Monte-Carlo-like sampling-based method to reconstruct the control given a target motion. Furthermore, they improve the sampling-based method by using the CMA-ES method of Hansen [Han06] and averaging of elite samples [LYG15]. Geijtenbeek et al. [GPvdS12] use PD control to track reference motions and Jacobian transpose control to deal with balance.

In early work, Faloutsos et al. [FvdPT01] introduce the idea of composable controllers and propose a framework that manages transitions between controllers based on SVM. The idea of having a graph of controllers is powerful, and the analogous kinematic solution for reusing motion capture by blending compatible clips has been very successful [AF02, KGP08]. Indeed, Liu et al. [LvdPY16] can synthesize long motions from short motion capture clips by building a control graph whose control fragments use linear feedback controllers estimated from many examples of physically valid trajectories produced by the improved SAMCON method [LYG15].

Reinforcement learning is probably the more popular approach for motion control construction. Liu et al. [LH17] use a deep Q-learning method to schedule control fragments for interactive human characters. In contrast, Peng et al. [PALvdP18] address more closely the motion reconstruction problem using RL, while also learning how to modify the control to satisfy different task objectives. Other notable results use RL to learn juggling control in the presence of disturbances such as wind [CL18], RL to control both torques and gains to recover from large disturbances while tracking reference motion, and RL to provide robust control to track a trajectory generated from motion matching [BCHF19]. In contrast, our work does not address the problem of a feedback control in the way that RL policies accommodate disturbances. Instead, we focus on the space-time optimization problem for a given target motion with the understanding that solutions permit the construction control fragments that can be scheduled to robustly address varying tasks. We note, however, that exploration in RL may likewise benefit from the use of the inverse dynamics feed forward torques and sample covariance shaping that we present here.

Inverse dynamics is an important problem in human character simulation. Ko and Badler [HB96] use inverse dynamics with other controllers which maintain the balance and the natural joint torque limits. The inverse dynamics problem is indeterminate in the double-stance phase, which they solve by distributing the force according to the percentage of the support on each leg. Inverse dynamics approaches can benefit from foot ground force data

to produce much less noisy results, but this force data is rarely available. Lv et al. [LCX16] use a data-driven approach to solve this problem. They reduce the dimension of the solution space of the inverse dynamics problem by constructing a prior from kinematics and dynamics database constructed with force plates and pressure pads, and then maximize the posterior to reconstruct the control. Liu et al. [LYWG13] use inverse dynamics to improve the initial PD targets in their soft body character control framework. Andrews et al. [AHK*16] develop a real-time tracking framework using inverse dynamics with orientation and position constraints from sensors and optical markers, as well as joints angle constraints from a motion prior. In contrast, we assume that the captured motion did not involve a joint limit or maximum torque and we simplify our inverse dynamics solve by dropping these constraints. Instead we rely on filtering to compute plausible approximate torques with inverse dynamics.

Finally, identifying foot-ground contact is a critical problem in many character animation problems. In particular, to correctly solve the inverse dynamics problem, we must know which bodies are in contact with the ground and providing forces on the character. Previous work has identified a variety of heuristics to identify foot-ground contact. Liu and Popović [LP02] find all the fixed points in space. Ikemoto et al. [IAF06] learn a k-nearest neighbors classifier from human annotated motion database to label the contact body in the motion frames. A better foot model can also improve the contact quality and produce appropriate contact forces to actuate the character. Park et al. [PYL18] propose a multi-segment foot model to produce realistic foot motions in a physics-based simulation. In our work we employ a relatively simple heuristic, but likewise make adjustments to the feet of our character to improve the final result.

3. System Overview

Our system reconstructs motion control from raw motion data, and a system overview diagram can be seen in Figure 2. Motion capture data does not typically contain body geometry information. Starting from a template model that fits the size of the captured subject, we optimize the foot and ground to make sure that each foot geometry creates good quality contacts with the ground (Section 5.4). We filter the raw motion data to reduce the noise from the motion capture system (Section 5.2). The filtered motion data is used as the input to the inverse dynamics computation, while the original reference motion is provided to the SAMCON algorithm. Inverse dynamics provides an approximation of control torques (Section 5), and we evaluate the quality of these torques and clamp them as necessary according to the distance between the zero-moment point and the support polygon (Section 5.5). Contact detection is required for inverse dynamics, quality evaluation, and the optimization of ground height and foot orientation (Section 5.3). We compute the initial covariance matrix for SAMCON based on the variance of the inverse dynamics torques (Section 5.6). In the SAMCON method, the sum of the PD torques and the inverse dynamics torques are used to actuate the character.

4. Sampling-based Control Optimization

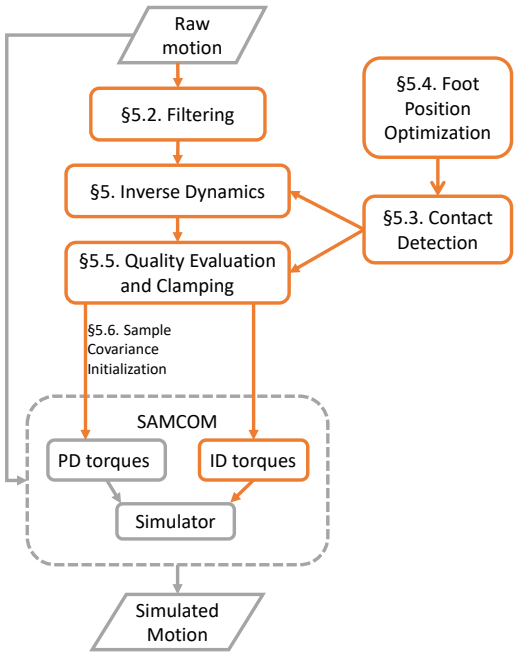

Figure 2: System Overview. The orange highlights our improvement. The solid arrows show the data flow, starting with motion capture in the lozenge shaped box at the top, and ending with the simulated motion and its control torques.

In this section, we review sampling-based control optimization proposed by Liu et al. [LYvdP*10, LYG15]. The character is an articulated rigid body system. We refer to the rigid bodies as segments, and we use a minimal set of generalized coordinates (i.e., joint angles) to represent the pose, $\mathbf{q} = (q_1, q_2, \ldots, q_n)$. The first six degrees of freedom in $\mathbf{q}$ give the root pose (i.e., the position and orientation of the hips). We use the PD control to drive all the joint degrees of freedom except the root,

$$\tau_i = k_{pi}(\bar{q}_i - q_i) - k_{di}\dot{q}_i, \tag{1}$$

where $\bar{q}_i$ and q_i are the target and current positions, $\dot{q}_i$ is the velocity, and k_{pi} and k_{di} are the gains. Given a reference trajectory $\tilde{\mathbf{m}} = \{\tilde{\mathbf{q}}_k\}$, if we naively use the reference trajectory $\tilde{\mathbf{m}}$ as the target trajectory for the PD control, it is unlikely to succeed due to external forces and modeling errors. At the core of the SAMCON method [LYvdP*10] is a stochastic algorithm that tries to solve this problem. Given a reference trajectory, the algorithm samples pose displacements $\Delta\mathbf{q}_k$ in order to find a revised target trajectory $\hat{\mathbf{m}} = \{\tilde{\mathbf{q}}_k + \Delta\mathbf{q}_k\}$ in the vicinity of the original motion where the PD control achieves a successful simulation.

In detail, at time t, the algorithm draws N_s samples $\{\Delta\mathbf{q}^i_{t+1}\}$ from the normal distribution $\pi_{t+1} = \mathcal{N}(\mathbf{0}, \Sigma_{t+1})$, where i is the sample index. The algorithm then uses $\tilde{\mathbf{q}}_{t+1} + \Delta\mathbf{q}^i_{t+1}$ as the target for the PD control. After advancing the simulation for a short interval of time Δt, we get N_s result states at time $t + \Delta t$. A greedy strategy would just select the state which is closest to the reference pose $\tilde{\mathbf{q}}_{t+1}$. However, the best choice at time $t + \Delta t$ can unfortunately make it difficult to match the reference pose at future times in

the simulation. Thus, the algorithm instead saves n_s states with the least tracking error. In the next short time interval (SAMCON refers to these as iterations), starting at time $t+\Delta t$, the simulation is initialized with the previous saved states and repeats this process until the control is fully constructed. The tracking error at a given time step is a weighted sum of four terms,

$$E = w_p E_p + w_r E_r + w_e E_e + w_b E_b. \tag{2}$$

These four terms measure error in pose control E_p, root control E_r, end-effector control E_e and balance control E_b, and are computed as follows:

$$E_p = \frac{1}{n_b}\sum_{i=1}^{n_b}(d_r(\mathbf{R}_i,\tilde{\mathbf{R}}_i)+0.1d_v(\omega_i,\tilde{\omega}_i)) \tag{3}$$

$$E_r = d_r(\mathbf{R}_{root},\tilde{\mathbf{R}}_{root})+0.1d_v(\omega_{root},\tilde{\omega}_{root}) \tag{4}$$

$$E_e = \frac{1}{k}\sum_{i=0}^{k}||\mathbf{p}_{iy}-\tilde{\mathbf{p}}_{iy}|| \tag{5}$$

$$E_b = \frac{1}{hk}\sum_{i=0}^{k}||\mathbf{r}_{ci}-\tilde{\mathbf{r}}_{ci}||+0.1||\mathbf{v}_{CoM}-\tilde{\mathbf{v}}_{CoM}||, \tag{6}$$

where n_b is the number of body segments, $d_r(\mathbf{R}_1,\mathbf{R}_2)$ measures the rotation difference, $d_v(\omega_1,\omega_2)$ measures the angular velocity difference, $\mathbf{R}_i$ is the ith joint rotation, ω_i is the ith joint velocity, k is the number of end-effectors (i.e., hands and feet), $\mathbf{p}_{iy}$ is the y component of the position of the ith end-effector, h is the height of the character, $\mathbf{r}_{ci}=(\mathbf{p}_{CoM}-\mathbf{p}_i)|_{y=0}$ is the position of end-effectors relative to the center of mass and projected onto the ground, and $\mathbf{v}_{CoM}$ is the linear velocity of the center of mass. Quantities with a tilde denote the corresponding target quantities. For a Δt time interval, or likewise a larger window of time, the tracking error is summed over the simulation time steps.

A single run of this algorithm usually fails to give us a successful trajectory. A failure occurs when the tracking error within a Δt time interval rises above a set threshold (at which point it is not necessary to simulate any of the following time intervals). We can simply rerun the algorithm until we get a successful one but this is obviously not a good solution: it does not learn anything from the past trials. An improvement is to adapt the mean and covariance of the sample distribution from the past [LYG15]. In SAMCON, the CMA-ES method is used to update the sample distribution $\pi_t=\mathcal{N}(\mathbf{m}_t,\sigma_t\Sigma_t)$ according to the quality of the samples. The step size σ_t controls add a uniform scaling of the sample distribution. A sliding window mechanism is introduced to accelerate the sample distribution adaptation process. The idea is to only reconstruct the control and adapt the sample distributions over a number of trials in a fixed time window consisting of a small number of Δt time intervals, and then slide the window forward when the control is good enough.

Thus, a full run of the algorithm consists of several trials. In each trial, the algorithm tries to reconstruct the control and update the sample distributions used in each Δt time interval. The trial is a success if the tracking error remains below the threshold for each Δt time interval in the sliding window. The algorithm gradually advances the window until it completely reconstructs the control for the whole motion. In our implementation, we use a slightly different strategy for the sliding window compared with the original paper. We update the samples only if the current trial reconstructs a better trajectory. We advance the window if the samples do not improve for T_{min} times or the sample distribution has been updated for T_{max} times.

Although SAMCON can successfully reconstruct controls for many motions, it has some disadvantages:

- It does not utilize the dynamics information of the motion. Therefore the success rate for each trial is not very high, especially for highly dynamic motion. The first trial is very likely to fail.
- Sometimes the character cannot make correct contact to actuate itself because of the wrong foot geometry configuration. This is worse for the first time step since the character has no chance to correct its foot position.

5. Inverse Dynamics

Liu et al. recognize the value of feed forward torques and do not initialize the PD control pose-displacement distributions with zero mean. Assuming some ground contact, it is straightforward to select a pose-displacement that will generate feed forward torques to compensate gravity, for instance, in the upper body to help keep an arm held horizontally. Liu et al. also identify the potential benefit of inverse dynamics, but instead allow sampling to compensate for dynamics because of the challenge of computing inverse dynamics torques in contact-rich tasks. We note that inverse dynamics is also challenging because velocity and acceleration information must be estimated from captured position data, which can be noisy. In our work, we focus on motion with important dynamics and fewer contacts, and take steps to deal with noise and obtain good estimates of foot ground contact. Although the direct application of inverse dynamics torques usually fails to generate correct control due to an approximate character model, capture error, and incorrect contact labels, we observe that the solution can still sometimes be a good appropriate initial guess for SAMCON.

Using the generalized coordinates, assuming that the first six degrees of freedom are the root's, the dynamics of the character can be formulated by the following equation:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}}+\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})+\mathbf{G}(\mathbf{q}) = \mathbf{Q}+\sum_i \mathbf{J}_i^T\mathbf{w}_i, \tag{7}$$

where $\mathbf{M}(\mathbf{q})$ is the mass matrix, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ is the Coriolis and centrifugal term, $\mathbf{G}(\mathbf{q})$ is the gravity term, $\mathbf{Q}$ is the joint torques, $\mathbf{w}_i$ is the wrench on the ith segment, and $\mathbf{J}_i$ is the Jacobian matrix of the ith segment. The velocity $\dot{\mathbf{q}}$ and the acceleration $\ddot{\mathbf{q}}$ are computed using the forward finite difference. The inverse dynamics algorithm tries to solve for the joint torques $\mathbf{Q}$ and the contact wrenches $\mathbf{w}_i$ given $\mathbf{M}(\mathbf{q})$, $\ddot{\mathbf{q}}$, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$) and $\mathbf{G}(\mathbf{q})$.

Because the character is underactuated, which means there is no force on the root, the first 6 components of the joint torque vector $\mathbf{Q}$ must be zero:

$$\mathbf{M}_{1:6,1:n}(\mathbf{q})\ddot{\mathbf{q}}+\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})_{1:6}+\mathbf{G}(\mathbf{q})_{1:6} = \sum_i \mathbf{J}_{i1:6,1:6}^T\mathbf{w}_i. \tag{8}$$

When there is only contact on one body segment (for example, one foot), the equation becomes a linear equation with 6 unknown

variables:

$$\mathbf{J}_{i1:6,1:6}^T \mathbf{w}_i = \mathbf{M}_{1:6,1:n}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})_{1:6} + \mathbf{G}(\mathbf{q})_{1:6}, \tag{9}$$

which we can easily solve for the contact wrench $\mathbf{w}_i$. We can get the joint torques $\mathbf{Q}$ by substitution of $\mathbf{w}_i$ in Equation 7.

When there are contacts on more than one segment (for instance, both feet), the inverse dynamics problem becomes indeterminate. We eliminate the indeterminacy by solving an optimization problem similar to the method proposed by Courtemanche [Cou14]. We minimize the total wrenches subject to the constraint given by Equation 8. This is a quadratic programming problem:

$$\min \sum_i \mathbf{w}_i^T \mathbf{w}_i, \tag{10}$$

subject to

$$\sum_i \mathbf{J}_{i1:6,1:6}^T \mathbf{w}_i = \mathbf{M}_{1:6,1:n}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})_{1:6} + \mathbf{G}(\mathbf{q})_{1:6}. \tag{11}$$

In both cases above, we need to identify which segments have contact forces. We discuss this in Section 5.3. After computing inverse dynamics on the trajectory with contact labels, we must filter the output to reduce the noise due to motion capture error. We discard the result if the contact forces are outside of the fiction cone. For each motion to be reconstructed, we use the method above to predict the control force. During the simulation, we add the predicted force and the PD force together and use this force to drive the character.

5.1. Zero-Moment Point and Support Polygon

The zero-moment point (ZMP) and the support polygon are two important concepts in legged locomotion control. We use them to measure the quality of filtered inverse dynamics torques in Section 5.2, so we briefly introduce these two concepts in this section. The ZMP can be interpreted as the center of pressure, and is defined as the point where the ground reaction force produces no torque in the horizontal plane [VB04]. The ZMP can be computed using the following equations:

$$\mathbf{F}^{gi} = m(\mathbf{g} - \mathbf{a}_G) \tag{12}$$

$$\mathbf{M}_P^{gi} = \overrightarrow{PG} \times m\mathbf{g} - \overrightarrow{PG} \times m\mathbf{a} - \dot{\mathbf{L}}_G \tag{13}$$

$$\overrightarrow{PZ} = \frac{\mathbf{n} \times \mathbf{M}_P^{gi}}{\mathbf{F}^{gi} \cdot \mathbf{n}}, \tag{14}$$

where m is the total mass of the character, $\mathbf{g}$ is the gravitational acceleration, $\mathbf{a}_G$ is the linear acceleration of the center of mass, $\dot{\mathbf{L}}_G$ is the rate of angular momentum at the center of mass, $\mathbf{n}$ is the normal of the ground plane, $\mathbf{P}$ is a point on the ground plane, $\mathbf{Z}$ is the ZMP and $\mathbf{G}$ is the center of mass.

The support polygon is the convex hull which encloses all the contact points. Assuming that there is no sliding and the ground is flat, the ZMP should always fall inside the support polygon [VB04].

5.2. Motion Filter

The raw motion data is usually noisy due to the capture error. Directly applying the inverse dynamics to the noisy data cannot generate correct control forces, which is not helpful and can even mislead SAMCON, because the noisy motion often leads to large joint torques. Andrews et al. [AHK*16] iteratively adjust the damping parameter according to the torque violation vector until the joint torques are within a reasonable limit. In contrast, we use a Butterworth filter, which alleviates the requirement of selecting the torque limit. We apply the Butterworth filter twice, first in the forward and then the backward direction, in order to avoid phase shift. We filter all degrees of freedom of the character in the raw motion data. The filtered motion is used as the input of inverse dynamics.

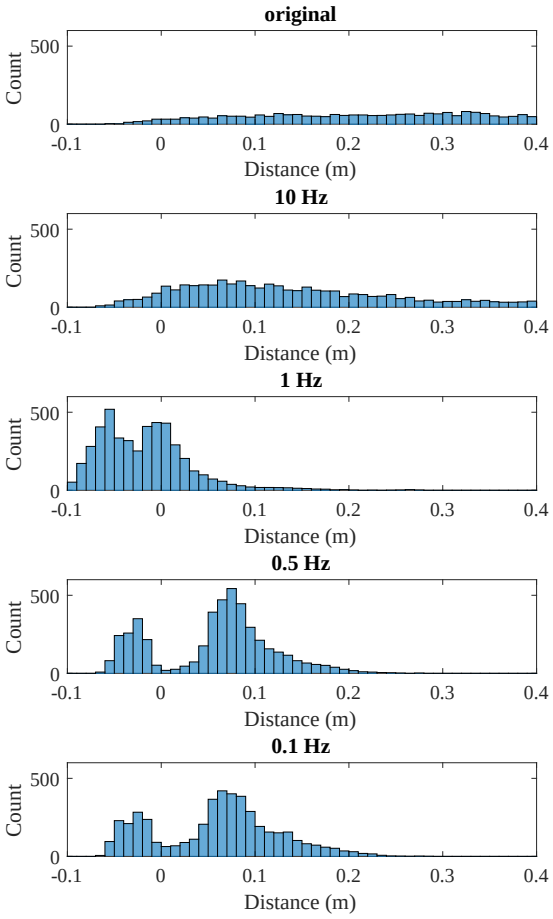

Figure 3: Histograms of signed distance between the ZMP and the support polygon, with different cutoff frequencies, for a walking motion. Moderate filtering moves the ZMP into the support polygon but over-filtering, here 0.5 Hz and lower, unfortunately moves the ZMP of many frames to a larger distance outside the polygon.

One question we must answer is how to choose an appropriate cutoff frequency. We know that the ZMP should be inside the support polygon. Therefore, we test different cutoff frequencies

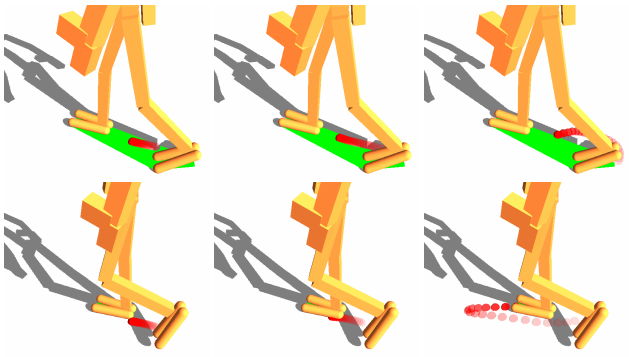

Figure 4: The ZMP trajectory and the support polygon at two frames in a walking motion, double stance (top) and single stance (bottom). From left to right, filter cutoff set to 0.1, 1.2, and 5 Hz.

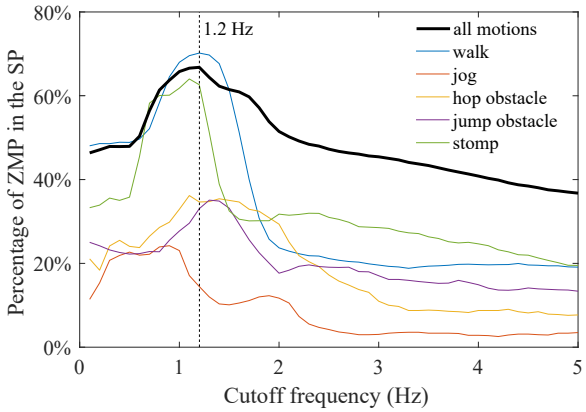

Figure 5: The percentage of frames where the ZMP falls in the support polygon for different filter cutoff frequencies, both for individual motions and the set of all motions, from which we identify an optimal cutoff frequency of 1.2 Hz.

for different motions and compute the signed distance between the ZMP and the support polygon as a measurement. The signed distance is defined as the distance from the ZMP to the closest edges of the support polygon. The sign is positive if the ZMP is outside of the support polygon and it is negative if inside.

Figure 3 shows the histograms of the signed distance between the ZMP and the support polygon with different cutoff frequencies for a walking motion. With a high cutoff frequency, the ZMP often falls outside of the support polygon. We can move the ZMP distribution into the support polygon by reducing the cutoff frequency. However, if the cutoff frequency is too low, we lose too much velocity information and many ZMPs fall outside. In this walking case, we can see two "peaks" in the histogram with a low cutoff frequency: one inside the support polygon and one outside. This is because the ZMP deteriorates to the projection of the center of mass onto the ground. It is inside the support polygon during the double stance phase but it is outside during much of the single stance phase. This is also visualized in Figure 4 left. In general, if the cutoff frequency is too low, the motion is over-filtered to the point that the joint velocities tend toward zero, and therefore the ZMP becomes the projection of the center of mass onto the ground (Figure 4 left). If the cutoff frequency is too high, the motion is still under-filtered and the ZMP can stray far from the support polygon due to noisy velocity and acceleration estimates (Figure 4 right). With an appropriate cutoff frequency, the ZMP should fall inside or be close to the support polygon (Figure 4 middle).

Figure 5 shows how the cutoff frequency affects the percentage of the ZMP in the support polygon for different motions (walk, jog, hop, jump, etc.) and for all motions. From this, we choose the cutoff frequency to be 1.2 Hz because this is where we observe the resulting ZMP inside the support polygon for the largest percentage of frames across all motions. Our intuition is that this seems aggressively low, but we also note that our primary goal is to provide feed forward torques that can help the sampling-based optimization and need not be perfect. It is very difficult to identify a level of filtering that produces a motion closer to a correct (i.e., successful) control trajectory, while this method serves as an inexpensive heuristic that is likely to benefit in the maximum number of frames.

5.3. Identifying Foot Ground Contact

We must identify the foot ground contact for three reasons. First, we must know the frames where we believe there is contact to optimize the ground position and foot rotation. This is discussed in Section 5.4. Second, for the inverse dynamics problem, we must know which body segments have contact to solve for the joint torques.

We use different methods to detect the foot-ground contact. For the foot and ground optimization, because we initially know nothing about the floor height and the foot collision-geometry rotation, we use a very simple but fast approach: we consider the foot is on the ground if the linear velocity of the foot is below a threshold and the y component of foot position is smaller than a threshold. This method is not very accurate. For example, it will miss cases where a foot is sliding on the ground. But missing some contacts is not a problem since we only need a collection of these frames to do the foot and ground optimization.

Once we know the floor position and foot collision-geometry, we simply use collision detection at each frame to identify contacts for the inverse dynamics computations. Again, this may not always be accurate, but we will discard frames that have a poor inverse dynamics solution, and thus, occasional errors are not problematic because we can fall back to relying on sampling to compensate for the dynamics.

5.4. Floor Position and Foot Orientation

Appropriate contact is essential for motion control. For example, having feet aligning with the ground can increase the contact area, and hence provides a larger support polygon, which helps the character maintain balance. But it can happen that the default foot collision-geometry that one would attach to the ankle of the skeleton is not good enough for making quality contacts with the ground. Setting appropriate foot geometry can be a very tedious job. We propose an automatic way to deal with it. Our solution for this is to optimize the orientation of collision-geometry attached

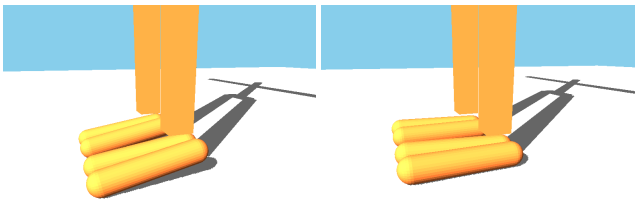

Figure 6: Before (left) and after (right) the foot position optimization. The latter is better aligned with the floor plane.

to the foot segment, while simultaneously identifying the ground height.

For simplicity, we model the feet geometry as n_f spheres (which we draw as capsules connecting pairs of spheres). As a heuristic, we minimize the average distance between the spheres and the ground. For all frames of a motion capture sequence, we can use forward kinematics to compute the rigid transform from the foot to the world at frame i, that is, homogeneous transformation matrix $\mathbf{g}_{wfi}$, consisting of rotation $\mathbf{R}_{wfi}$ and foot frame origin ${}^{w}\mathbf{p}_{fi}$ in world coordinates. As described in the previous section, we try to identify all the frames when the foot contact occurs, and we denote these frames as set S.

Let $\mathbf{c}_k$ and r_k for $k = 1 \cdots n_f$ be the centers and radiuses of spheres in the foot's local frame. We use a rotation $\mathbf{R}$ to orient the collision-geometry with the floor. The center position of each sphere in the world frame is $\mathbf{g}_{wfk}\mathbf{R}\mathbf{c}_k$. We denote the floor plane equation in vector form as $(0,1,0,-h)$, where h is the height. We then minimize the average distance between the spheres and the ground in all frames with contact,

$$\min_{h,\mathbf{R}} \sum_{i \in S} \sum_{j=1}^{n_f} \left| (0,1,0,-h)\, \mathbf{g}_{wfi}\, \mathbf{R}\, \mathbf{c}_j - r_j \right|. \tag{15}$$

We use the Levenberg–Marquardt algorithm provided by ALGLIB to solve this optimization problem. Figure 6 shows an example of the optimization of a T-pose motion. Both the floor plane and the foot rotation are optimized to be better aligned with each other.

5.5. Quality of Inverse Dynamics

While good feed forward inverse dynamics torques can help SAMCON find the solution faster, poor torque estimates can degrade performance. Therefore, we evaluate the quality of the inverse dynamics control torques τ and only use it if the quality is good enough. We achieve this by computing the signed distance d between the ZMP and the support polygon for each frame. If the signed distance is smaller than zero, we consider the quality is good and keep the corresponding inverse dynamics control torques. If the signed distance is beyond the threshold γ, we set the corresponding inverse dynamics control torques to zero. This can be a hard threshold, or likewise, if the signed distance is between zero and γ, we can also use a smoothstep function to smoothly drive the feed forward inverse dynamics torques to zero. That is,

$$\hat{\tau} = \text{smoothstep}(d) \cdot \tau, \tag{16}$$

ALGORITHM 1: SAMCON with inverse dynamics

input : reference trajectory $\tilde{\mathbf{m}} = \{\tilde{\mathbf{q}}_k\}$
output: reconstructed trajectory $\mathbf{m}$

```
detect foot contacts and divide m̃ into m̃_c where the character has foot
  contacts and m̃_a where the character is in the air
optimize foot position using m̃_c
compute inverse dynamics torques {τ_k}
clamp inverse dynamics torques according to the signed distance
  between the ZMP and the support polygon d using the smoothstep
  function τ̂_k = smoothstep(d) · τ_k
compute the initial covariance matrix Σ = diag(σ_s) using the inverse
  dynamics torques
initialize default sample distributions {π_k}
set the sliding window at the beginning of m̃
repeat
    for each frame k in sliding window do
        draw N_s samples {Δq_k^i} ~ π_k
        simulate using τ_k = K(q̃_k + Δq_k^i − q_k) − Dq̇_k + τ̂_k
        save n_s best samples in s_k
    end
    for each frame k in sliding window do
        rank s_k according to the sample quality
        update π_k with s_k using CMA-ES
    end
    while the first interval in the sliding window is good enough do
        slide the window forward
    end
until completely reconstruct control
```

where smoothstep(d) is defined as

$$\text{smoothstep}(d) = \begin{cases} 1 & d \le 0 \\ \frac{2d^3}{\gamma^3} - \frac{3d^2}{\gamma^2} + 1 & 0 < d < \gamma \\ 0 & d \ge \gamma \end{cases}. \tag{17}$$

The intuition for using a smoothstep rather than a hard cutoff is that the PD control optimization will not need large jumps to account for the sudden disappearance of feed forward torques, and thus the algorithm will reconstruct a smoother motion.

5.6. Sample Covariance Initialization

SAMCON uses an identity matrix as the initial covariance matrix of the sample distribution (note that the CMA step-size parameter provides uniform scaling of this distribution). While CMA-ES evolves the covariance during the sample adaptation process, setting a better initial covariance matrix can improve the quality of the reconstructed motions. We compute the initial covariance matrix based on the variance of the inverse dynamics torques. Specifically, we compute the variance of the inverse dynamics torques for each degree of freedom except for the root degrees of freedom, and use this for the initial distribution. That is,

$$\sigma_d = \text{Var}(\mathbf{T}), \tag{18}$$

where $\mathbf{T} = (\tau_1 \tau_2 \cdots \tau_t)$ is a matrix whose columns are the inverse dynamics torques, and $\text{Var}(\mathbf{T})$ computes the variance of each row of $\mathbf{T}$. We scale the vector σ_d so that its largest component is equal

Table 1: We limit the maximum computation time for each motion and compare the performance of our method and SAMCON. The success rate is the proportion of the successful trials in all the trials. The tracking error is the lowest error among all the trials. The parameters are $\sigma = 0.1$, $\gamma = 20$, $T_{min} = 5$ and $T_{max} = 20$.

	success rate (%)			tracking error			clip length (s)	compute (h)
motion	SAMCON	ours (unfiltered)	ours (filtered)	SAMCON	ours (unfiltered)	ours (filtered)		
hop[1]	51	**63**		6.68	**6.22**		7.9	24
obstacle hop	15	49	**51**	8.21	8.80	**5.81**	2.3	3
obstacle jump[1]	0	0	2	12.45	15.03	**9.62**	4.6	12
stomp	93	97	**100**	4.24	4.63	**3.85**	2.3	3
lift leg	90	**96**	93	6.03	5.84	**5.33**	2.5	3
swing leg	91	77	**100**	**4.56**	5.56	4.67	2.3	3
kick	**100**	**100**	**100**	5.21	**5.15**	5.56	2.2	3
Wushu kick	11	37	**41**	10.80	10.50	**8.19**	2.4	3
Cha Cha	91	**97**	94	6.41	7.19	**6.11**	2.9	3

1. We use $N_s = 800$, $n_s = 200$ instead of $N_s = 280$, $n_s = 70$ for these motions.

Table 2: Stiffness and damping parameters.

joint	stiffness	damping
hip	800	40
knee	800	40
foot	300	25
spine	600	30
neck	100	10
shoulder	300	15
elbow	200	10
wrist	30	1.5

to one,

$$\sigma_s = \sigma_d / \max_i \sigma_{di}. \tag{19}$$

Thus, $\mathrm{diag}(\sigma_s)$ serves as the initial covariance matrix of the sample distribution in the CMA-ES algorithm. Our intuition is that degrees of freedom which have a larger variance of torques should allow larger PD offsets to produce the necessary torques, while degrees of freedom that have almost constant torques can have a smaller exploration of PD offsets in the sample simulation, and this will lead to smoother final results.

Algorithm 1 shows the SAMCON method modified to include inverse dynamics. First, we detect foot contacts for the frames in the target motion. Next, we set up appropriate foot position using the method in Section 5.4. Then we apply inverse dynamics and get the predicted control torques. We evaluate the quality of the inverse dynamics control torques and clamp them using the smoothstep function. We compute the initial covariance matrix $\Sigma = \mathrm{diag}(\sigma_s)$ using the inverse dynamics torques. The rest is similar to the SAMCON, except that we use the sum of the inverse dynamics control torques and the PD control torques (which uses the PD offsets plus the reference trajectory) to drive the character. For the initial sample distribution, we set a zero mean PD offset, $\mathbf{m} = \mathbf{0}$, step size $\sigma = 0.1$, and covariance $\Sigma = \mathrm{diag}(\sigma_s)$ as discussed above.

6. Results and Discussion

We use DART for the rigid body simulation [LGH*18], with a simulation time step of 10^{-4} s. We have tested motions from the SFU motion capture database and other motions we captured, with lengths varying between 2 and 8 seconds (details are listed in Tables 1 and 4). The characters have different skeletons and between 52 and 78 degrees of freedom. The stiffness and damping parameters we use for these characters are shown in Table 2. For all experiments (except where specified), we set the number of samples to $N_s = 280$ and the number of saved samples to $n_s = 70$. All tests were run on Compute Canada with Intel E5-2683 v4 Broadwell @ 2.1 Ghz.

Please see our supplemental video for examples of physically valid motions optimized with our method. The video results qualitatively demonstrate the smoothness of our results. Specifically, we show results that use filtered inverse dynamics torques and sample covariance initialization in comparison to our implementation of the improved SAMCON method. In the following sections we also provide a collection of quantitative experiments that give an indication of how our approach influences sample simulation success rates, error, and motion quality as measured by torque squared.

6.1. Filtering and Trial Success Rate

In our first experiment, we evaluate the influence of filtering on the success of trials. We limit the maximum computation time for each motion and compare the success rate and the tracking error of our method and SAMCON. A trial is a single motion construction process for the motion frames in the sliding window, as discussed in Section 4. The success rate is defined as the proportion of successful trials in all the trials. A trial is considered successful if it reconstructs the control in the sliding window. The tracking error is the lowest error among all the trials during this run. Note that *hop*, *lift leg*, *swing leg*, and *kick* have been preprocessed and hence are cleaner than the rest. Table 1 shows the result. Given that

Table 3: We use different threshold γ and increase the foot size compared with Table 1. Other parameters are the same as Table 1.

	success rate (%)			tracking error		
motion	SAMCON	γ=0	γ=20	SAMCON	γ=0	γ=20
obstacle hop	12	21	**69**	7.16	6.78	**5.65**
obstacle jump	0	2	**3**	11.44	9.46	**8.23**
stomp	99	**100**	**100**	4.57	**3.75**	**3.75**
Wushu kick	31	26	**38**	9.77	9.92	**8.89**
Chacha	**96**	91	95	6.20	**5.94**	6.39

the algorithm uses random sampling, each test is run 5 times and the result is the average.

For most motions, our method performs better in terms of both the success rate and the tracking error, especially for highly dynamic motions like *obstacle hop*, *obstacle jump*, and *two feet jump*. The filter helps improve the quality of the inverse dynamics when the motions contain noise.

6.2. Inverse Dynamics Quality Threshold Evaluation

As mentioned in Section 5.5, we use a threshold γ to determine whether the quality of the inverse dynamics is good. In the second experiment, we evaluate different thresholds of $\gamma = 0$ and $\gamma = 20$ cm with a hard cutoff (rather than smoothstep) to observe the effect of the threshold. We also limit the computation time for this experiment. The result is shown in Table 3.

Setting threshold $\gamma = 0$ will only keep the inverse dynamics force whose corresponding ZMP is strictly inside the support polygon. The table suggests that this may not always be the best idea since it can discard a lot of the inverse dynamics results. Because the contact detection is not very accurate, especially when the foot is just contacting or leaving the ground, the support polygon we compute might not reflect the current situation, and some valid results might be discarded. Using a larger threshold can avoid a *false alarm* in this situation. The table shows that a threshold $\gamma = 20$ cm can increase the success rate of sample simulations and lower the error. For motions where double-support frequently occurs, such as *Cha Cha*, we speculate that the larger threshold does not have an obvious benefit because of frequently larger support polygons with the two feet on the ground.

6.3. Computation Time and Motion Quality

In the third experiment, we do not limit the computation time to understand the natural variation of termination times under different conditions. We do note that there is certainly some amount of luck involved given the results depend on random sampling.

We remove fingers and toes from the motion data and our simulated character in order to speed up the computation. We compare three cases: the improved SAMCON method, our method with the default covariance matrix, and our method with sample covariance initialization (SCI). We compare the success rate, tracking error, and compute time.

Notice that with the default covariance, our method performs better in terms of the success rate and tracking error as shown in Table 4. The improvement is more significant for highly dynamic motion like *obstacle jump* and *two feet jump*. In terms of the time performance, our method does not have an improvement compared with SAMCON, but this is not unexpected. Our method performs better in some motions but SAMCON can terminate earlier in others because the scheme for advancing the sliding window cannot benefit from the higher success rate in each trial. That is, although our method may find a better solution at the first attempt, it might still vainly try another T_{min} times to update the sample distribution in the sliding window, making little or no improvement.

Despite this, the higher success rate in each trial is still generally a good property we desire. With the same step size σ, our method usually finds an acceptable solution earlier than SAMCON, especially for short clips where the advancing of the sliding window is not important. For example, if we use the same step size $\sigma = 0.8$ for the hopping motion, because of the higher success rate, our method can find an acceptable solution at the 7^{th} trial, while SAMCON cannot find one until the 41^{st} trial. With a larger step size $\sigma = 1.5$, SAMCON can find an acceptable solution at the 6^{th} trial, but the simulated trajectory looks noisier due to a larger step size.

In terms of the motion quality, Table 4 shows that our method tends to lead to lower error. With sample covariance initialization, our method can further reduce the error. In general low variance will lead to smooth motions, and low error, just as the CMA step size will accomplish the same thing by reducing the variance of samples. The drop in success rate is likewise expected, but we note there is a much larger drop in success rate for long difficult motions. The hop motion is both long and involves somewhat delicate balancing on one foot during dynamic arm and leg motion.

Figure 7 shows the average torque squared for different motions, which can be considered as an evaluation of the motion quality. Our method has a lower average torque square, which indicates that our method has a lower variance of joint torques, and thus the reconstructed motion has less noise. Our supplemental video also shows that this represents a significant improvement in the motion quality. Note that in the video, the characters have different morphology for different motions because we use motion data from different databases. Furthermore, for some motions, SAMCON fails to reconstruct the control while our method succeeds, but it does not mean SAMCON cannot generate the control for these motions. If we increase the number of samples and the computation time, SAMCON can probably generate control for these motions.

6.4. Control Fragments

While the bulk of our investigation has been on the problem of producing a physically valid control trajectory from motion capture, the real application of this result is in the creation of control fragments that can be used for feedback control during trajectory following in a real-time interactive simulation. We created control fragments for a walk cycle by running our method on a one-minute motion trajectory. Following Liu et al. [LvdPY16] we estimate a feedback controller and we show that the resulting

Table 4: We do not limit the maximum computation time. The parameters are $\sigma = 0.1$, $\gamma = 20$, $T_{min} = 3$ and $T_{max} = 20$. SCI is the abbreviation for Sample Covariance Initialization. We remove thumbs and toes in order to reduce computation time.

	success rate (%)			tracking error			compute time (h)			clip length (s)	N_s	n_s
motion	SAMCON	ours	ours (SCI)	SAMCON	ours	ours (SCI)	SAMCON	ours	ours (SCI)			
walk	**99.6**	99.3	96.7	4.99	4.61	**3.63**	4.57	**4.38**	4.52	4.8	280	70
hop	64.8	**80.8**	21.5	5.95	**5.82**	7.81	**20.21**	22.68	48.56	7.9	800	200
obstacle hop[1]	93.6	**98.4**	96.4	4.58	4.50	**4.12**	3.77	3.57	**3.33**	2.3	280	70
obstacle jump	57.2	**65.2**	44.0	7.28	5.44	**5.04**	**19.44**	21.62	21.64	4.6	280	70
stomp[1]	**100.0**	**100.0**	99.1	3.29	3.28	**2.66**	3.80	3.61	**3.33**	2.3	280	70
two feet jump[1]	95.1	**99.0**	92.0	4.31	3.24	**2.36**	**6.38**	7.97	6.68	5.9	280	70
Wushu kick[1]	87.1	**95.3**	92.9	5.79	4.27	**4.02**	**3.47**	4.18	4.21	2.4	280	70
Cha Cha[1]	99.7	**99.8**	99.2	4.15	3.84	**3.20**	6.25	**6.05**	6.42	2.9	280	70

1. We use larger feet for these motions.

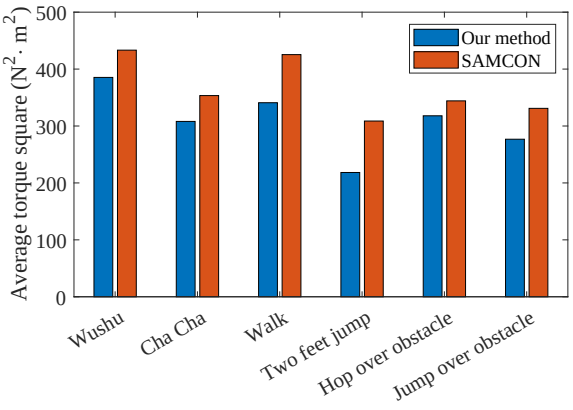

Figure 7: Average torque square of several motions.

controller can react to external disturbances and resist up to 80 N of force applied for 0.1 s on the chest.

7. Limitations, and Future Work

Although our method has a higher success rate, our current approach cannot use this advantage in order to reduce computation time. In future work, we want to find a better sliding window strategy which can benefit from a higher success rate. In order to achieve this, we need to explore a better way to decide whether the reconstructed control in the sliding window is good enough, and a more aggressive sliding window scheme which can advance the window faster. A naive approach is to advance the window as soon as the error in the window is below some threshold. But this threshold can vary between different motions.

Another limitation is that our selection of the ZMP distance threshold γ is empirical. It would be beneficial to have a method to choose the threshold for different motions. Alternatively, we could devise a better approach to evaluate the quality of the inverse dynamics rather than using a hard threshold.

8. Conclusions

In this paper, we present techniques to improve the improved SAMCON method. We propose filtered inverse dynamics as feed forward control torques, and sample covariance initialization based on the variance of inverse dynamics torques. Overall, we observe that our method leads to higher sample success rates, lower error, and most importantly qualitatively smoother results. We show that our method produces control trajectories that can be used to produce control fragments for real-time feedback control.

Filtering of the motion capture data is necessary to reduce the noise of the motion data to improve the quality of the inverse dynamics. We present a simple approach to decide the filter cutoff by picking the cutoff that leads to the largest number of motion frames with the ZMP inside the support polygon. We also use a simple method to set up the foot orientation and floor position, which ensures better foot-ground contact. Filtering and foot adjustment can generate good quality inverse dynamics control torques, which are useful for improving sampling-based control optimization. Our method has the potential to reduce the computation time of SAMCON if we have a better sliding window strategy to use the higher success rate of our method.

References

[AF02] ARIKAN O., FORSYTH D. A.: Interactive motion generation from examples. *ACM Trans. Graph. 21*, 3 (July 2002), 483–490. 2

[AHK*16] ANDREWS S., HUERTA I., KOMURA T., SIGAL L., MITCHELL K.: Real-time physics-based motion capture with sparse sensors. In *Proceedings of the 13th European Conference on Visual Media Production (CVMP 2016)* (New York, NY, USA, 2016), CVMP 2016, Association for Computing Machinery. 3, 5

[BCHF19] BERGAMIN K., CLAVET S., HOLDEN D., FORBES J. R.: Drecon: Data-driven responsive control of physics-based characters. *ACM Trans. Graph. 38*, 6 (Nov. 2019). 2

[CL18] CHEMIN J., LEE J.: A physics-based juggling simulation using reinforcement learning. In *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games* (New York, NY, USA, 2018), MIG '18, ACM, pp. 3:1–3:7. 2

[Cou14] COURTEMANCHE S.: *Analysis and Simulation of Optimal Motions in Rock Climbing*. Theses, Université de Grenoble, Oct. 2014. 5

[FvdPT01] FALOUTSOS P., VAN DE PANNE M., TERZOPOULOS D.: Composable controllers for physics-based character animation. In *Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques* (New York, NY, USA, 2001), SIGGRAPH '01, ACM, pp. 251–260. 2

[GPvdS12] GEIJTENBEEK T., PRONOST N., VAN DER STAPPEN A. F.: Simple data-driven control for simulated bipeds. In *Proceedings of the 11th ACM SIGGRAPH/Eurographics conference on Computer Animation* (2012), pp. 211–219. 2

[Han06] HANSEN N.: The CMA evolution strategy: a comparing review. In *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, Lozano J., Larranaga P., Inza I., Bengoetxea E., (Eds.). Springer, 2006, pp. 75–102. 2

[HB96] HYEONGSEOK KO, BADLER N. I.: Animating human locomotion with inverse dynamics. *IEEE Computer Graphics and Applications 16*, 2 (1996), 50–59. 2

[IAF06] IKEMOTO L., ARIKAN O., FORSYTH D.: Knowing when to put your foot down. In *Proceedings of the 2006 symposium on Interactive 3D graphics and games* (2006), pp. 49–53. 3

[KGP08] KOVAR L., GLEICHER M., PIGHIN F.: Motion graphs. In *ACM SIGGRAPH 2008 Classes* (New York, NY, USA, 2008), SIGGRAPH '08, ACM, pp. 51:1–51:10. 2

[LCX16] LV X., CHAI J., XIA S.: Data-driven inverse dynamics for human motion. *ACM Trans. Graph. 35*, 6 (Nov. 2016). 3

[LGH*18] LEE J., GREY M., HA S., KUNZ T., JAIN S., YE Y., SRINIVASA S., STILMAN M., LIU C.: Dart: Dynamic animation and robotics toolkit. *Journal of Open Source Software 3*, 22 (2018), 500. 8

[LH17] LIU L., HODGINS J.: Learning to schedule control fragments for physics-based characters using deep q-learning. *ACM Trans. Graph. 36*, 3 (June 2017). 1, 2

[LP02] LIU C. K., POPOVIĆ Z.: Synthesis of complex dynamic character motion from simple animations. *ACM Trans. Graph. 21*, 3 (July 2002), 408–416. 3

[LvdPY16] LIU L., VAN DE PANNE M., YIN K.: Guided learning of control graphs for physics-based characters. *ACM Transactions on Graphics 35*, 3 (2016). 2, 9

[LYG15] LIU L., YIN K., GUO B.: Improving sampling-based motion control. *Comput. Graph. Forum 34*, 2 (May 2015), 415–423. 1, 2, 3, 4

[LYvdP*10] LIU L., YIN K., VAN DE PANNE M., SHAO T., XU W.: Sampling-based contact-rich motion control. *ACM Trans. Graph. 29*, 4 (July 2010), 128:1–128:10. 1, 2, 3

[LYWG13] LIU L., YIN K., WANG B., GUO B.: Simulation and control of skeleton-driven soft body characters. *ACM Trans. Graph. 32*, 6 (Nov. 2013). 3

[PALvdP18] PENG X. B., ABBEEL P., LEVINE S., VAN DE PANNE M.: Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Transactions on Graphics (Proc. SIGGRAPH 2018) 37*, 4 (2018). 2

[PYL18] PARK H., YU R., LEE J.: Multi-segment foot modeling for human animation. In *Proceedings of the 11th Annual International Conference on Motion, Interaction, and Games* (New York, NY, USA, 2018), MIG '18, Association for Computing Machinery. 3

[SKL07] SOK K. W., KIM M., LEE J.: Simulating biped behaviors from human motion data. *ACM Trans. Graph. 26*, 3 (July 2007). 2

[SvdP05] SHARON D., VAN DE PANNE M.: Synthesis of controllers for stylized planar bipedal walking. In *Proc. of IEEE International Conference on Robotics and Animation* (2005). 2

[VB04] VUKOBRATOVIC M., BOROVAC B.: Zero-moment point - thirty five years of its life. *I. J. Humanoid Robotics 1* (03 2004), 157–173. 5